卒　業　研　究　報　告　書

令和二年度

研究題目

はらい検出による
古文書画像の切り出し方法の提案

指導教員　　松　尾　賢　一　　教授

氏　　名　　片　山　歩　希

令和 3 年 1 月 26 日　提出

奈　良　工　業　高　等　専　門　学　校　　情　報　工　学　科

はらい検出による古文書画像の切り出し方法の提案

松尾研究室 片山 歩希

日本には過去の歴史資料として，約10億冊の古文書・古記録が残っている．この古文書は歴史における重要な出来事や日常的な事務処理について理解するための歴史認識における重要な文献資料の一種である．よって，古文書を解読することでより深い歴史を知ることができる．しかし，古文書を解読できる人々の数は日本人口の0.01％にも満たない．そこで，古文書解読（以下，翻刻と称す．）の割合を増やすためにも，古文書解読の初学者に対して支援するシステムを作成することで，翻刻初学者の挫折を軽減や翻刻初学者人口の増幅が考えられる．

現在では，古文書の大規模なディジタル化とオープン化が急速に広まり，Web経由で日本の古文書画像にアクセスできるようになった．そこで，古文書画像を用いて初学者が熟練者の手を借りずに翻刻を行うことで熟練度が向上すると考える．

初学者が翻刻を行うには，2つの難しい段階が挙げられる．それは，古文書の書式や書体である．古文書の中でも，近世古文書の書式は文字同士が繋がっているために初学者では一文字の判断が難しい．また，書体では文字が草書のくずし字で書かれているため，書かれている文字が識別できずにくずし字辞書を引くのも難しい．

従来では，文字数を制限することにより，一文字を切り出してシステムが動作している．そこで，本研究では入力画像が何文字でも切り出しを行えるルーチンを提案する．提案する手法としては，従来の切り出し方法であるグラフ的処理ではなく，古文書の特徴である“はらい”に着目した処理である．古文書画像中から“はらい”を検出して，文字同士を繋がりを断つことで一文字が切り出せる仕組みとなる．

結果として，処理における“はらい”の定義を自らの経験則に基づいて決定したために，求めている切り出しができない出力も見られた．今後の課題としては，機械学習による“はらい”の定義やクラスタリングでの“はらい”の分類をすることで求めている出力が得れると考える．

目次

1　まえがき

日本には過去の歴史資料として，約10億冊の古文書･古記録が残っている[1]. この古文書は歴史における重要な出来事や日常的な事務処理について理解するための歴史認識における重要な文献資料の一種である.

また，古文書は時代によって文書の書体や体裁が変化する.

その中でも近世古文書は，書体が解読困難な**くずし字**と文字同士が繋がっている**つなげ字**で構成されている. この書体の解読困難性と一文字の識別がつきにくいことから近世古文書を解読できる人々の数は日本人口の0.01％にも満たない[2].

このような歴史資料（考古学的資料）の解読には段階があり，

> 「地球史解読を考古学的試料の解読にたとえると、
> （１）重要な試料のありそうなところをかぎつけて試料を確保する段階、
> （２）後の時代についた汚れや傷を区別し、初期の情報を読みとる段階、
> （３）試料に書かれている内容を読む段階、
> （４）試料の実体と試料が記録された歴史的背景を読む段階に分けられる。」
> (川上ら，1996)[3]

と定義づけられている.

これらの各段階を本論文では，以下のように解釈を行う.

(I) 古文書資料を見つける段階
(Ⅱ) 古文書に書かれている一文字を抽出する段階
(Ⅲ) 古文書に書かれている文字から現代文字に置き換える段階
(Ⅳ) 古文書の内容を理解する段階

(I) は古文書の大規模なディジタル化とオープン化が急速に広まり，Web経由で日本の古文書画像にアクセスできるようになった[1]. よって，(I) の段階は古文書を解読する（以下「翻刻」と称する）初学者でも容易に達成する. そこで，次なる課題として（(Ⅱ)）の段階を翻刻初学者でも容易に達成させる方法が必要となる.

（Ⅱ）の段階は，初学者にとって学習が難しい段階である. 理由としては，私が考える次の二点を挙げる. 一つ目は，前述にもあるように，文章が**くずし字**と**つなげ字**で構成されているために，初学者では，一文字の判断がつきにくい. 二つ目は，古文書の文章には，句読点や字下げなどがないために一文の判断がつかない. よって，初学者にとっては文同士の切れ目からの一文字の抽出も難

しいものとなる．

現在，(Ⅱ) の段階を翻刻初学者でも容易に達成させる方法に対して，世の中で初学者向けの古文書翻刻支援システムが普及し始めている．普及している初学者向けの古文書翻刻支援システムでは，語句サポートを目的とした，学習セッションを設けて語句を学ぶもの [4] や翻刻前の古文書に対して，複数人で翻刻を行うＳＮＳ [5][6] がある．画像処理を使った初学者向けの古文書翻刻支援システムでは，語句サポートを目的とした，くずし字認識 AI を用いて古文書画像から一文字を切り出し，対応する現代語の候補を出すもの [7] や文字切り出しサポートを目的とした，古文書画像から一文字切り出しを自動で処理するもの [8][9] がある．

しかし，これらのシステムは翻刻初学者が（Ⅲ）の段階 [3] である古文書に書かれている文字から現代文字に置き換える段階を達成してないため，翻刻初学者は完全に扱うことができない．理由として，以下の二点が挙げられる．

まず一点目は，翻刻初学者古文書画像から一文字切り出しができない．なぜなら前述にもあるように，文章が**くずし字**と**つなげ字**で構成されているために，初学者では，一文字の判断がつきにくい．また二点目は，一文字を認識したとしても対応する現代語が判断できない．なぜなら，古文書の内容理解をできない初学者が文章に沿った翻刻前の古文書に対して，対応する現代語を充てることはできないからである．そこで，入力する古文書画像の文字数を翻刻初学者が分からなくても，自動で一文字を切り出すルーチンを作成する必要がある．

本研究では，文字数が三文字と指定されている従来の一文字切り出し方法から文字の拡張を考え，新たな切り出し方法を提案する．

2　古文書について

2.1　古文書とは

古文書は歴史における重要な出来事や日常的な事務処理について理解するための歴史認識における重要な文献資料の一種であると定義づけられている．つまり，古文書を読むことでその時代の出来事や出来事の背景を理解することができる．また，時代によって古文書の書式や書体の変化が見られ，古文書の名前も時代によって変化する．本研究では，1603 年以降に記されている近世古文書について取り扱う．

2.2　近世古文書の書式と特徴

近世古文書では，文書の書式が図 1 のように，右詰め縦書きになっており，文書の内容によって異なるが題名，本文，筆者，日付，宛名の順になっている．

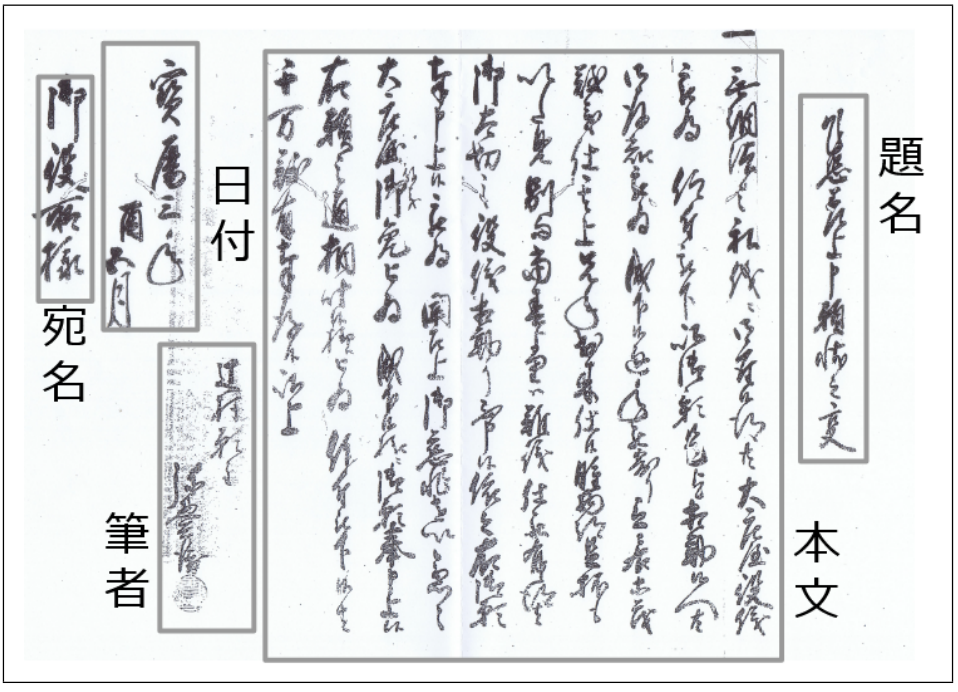

図 1　近世古文書の書式

特徴として，近世古文書は書流が青蓮院流（しょうれんいんりゅう）で書かれており．草書のように書かれているくずし字，一文字一文字が繋がっているつなげ字で記されている (図 2)．さらに，句読点などはなく，改行や改行後の一文字空けもないため，一文の区切りの判断が困難である．また，和漢混合文であるため，本文の内容を理解する際にも時間を要する．

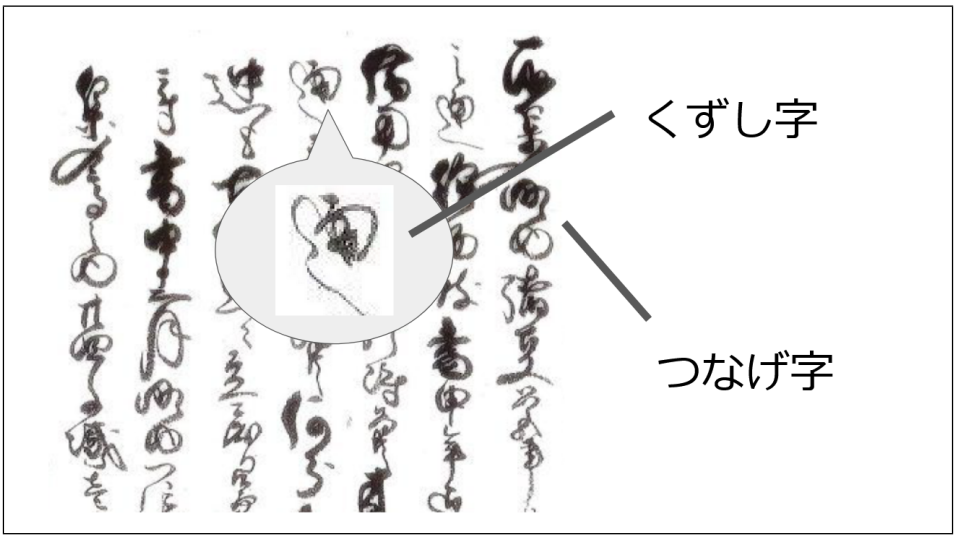

図 2　くずし字，つなげ字

3　従来システム

本稿では，従来システムとして桂尚輝氏が作成した三文字古文書画像から現代語の文字コードを出力するシステム（以降，桂氏のプログラムとする．）を挙げる．

桂氏のプログラムは，三つの過程で構成されており，三文字画像から一文字画像の抽出（切り出し），一文字画像から特徴量抽出（特徴量抽出），特徴量から文字の特定（文字特定）となっている (図 3).

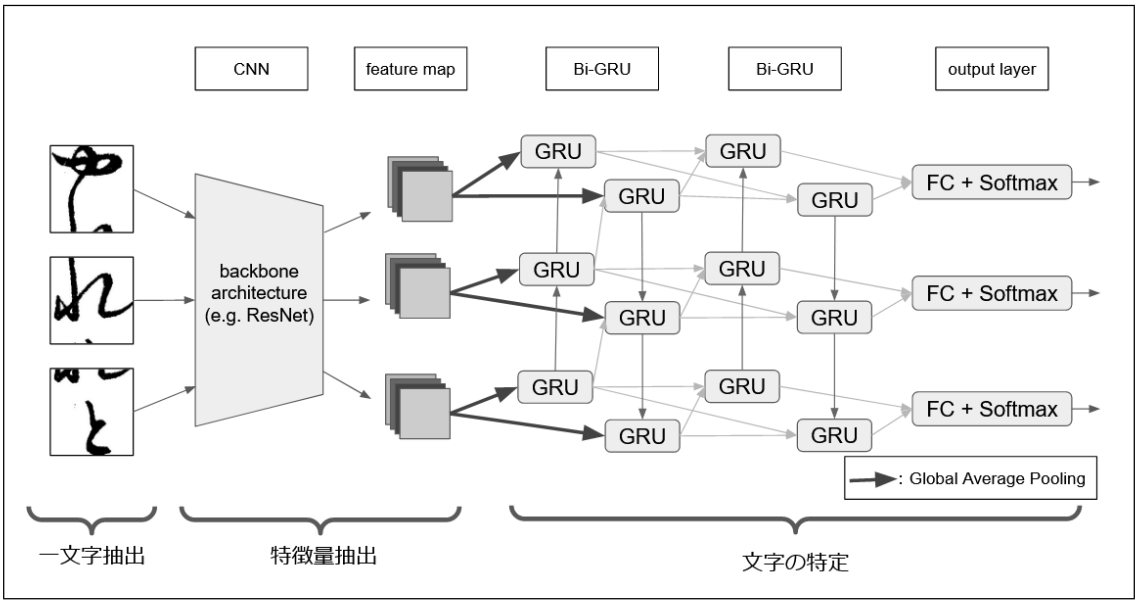

図 3　桂 尚輝氏のプログラム構造

学習器作成の過程では，CNN-BiGRU を用いる．CNN を用いて特徴量を抽出し，特徴量を Bi-GRU で学習させて文字の特定する．図 3 のように，CNN, feature map, Bi-GRU, output layer に分ける．また CNN のモデルは DenseNet-201, Inception-v4, SE-ResNeXt-101 の 4 つを学習させ，各学習器をアンサンブルすることによって，くずし字の識別度は 90.6 % を出す．

本稿では，本研究で使用する切り出しの部分のみ説明する．

3.1　使用データセット

桂氏のプログラムで使用されるデータセットは，第 23 回 PRMU アルゴリズムコンテストくずし字認識チャレンジ 2019[10] のデータセットを使用し，データセットの画像ファイル数は学習用に 119,997 枚, テスト用に 16,387 枚である．1 枚の古文書画像内に三文字が必ず含まれる．次の図 4 はその例の一つである．

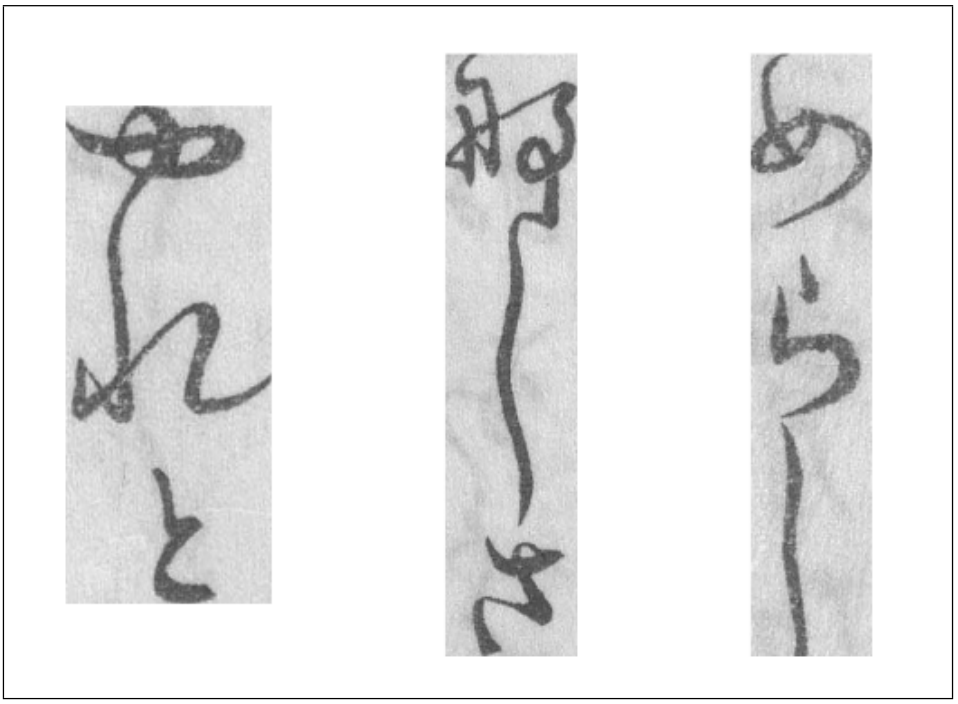

図 4　データセットの画像例

3.2　三文字画像から一文字画像の切り出し

桂氏のプログラムは，特徴量抽出を行うために三文字古文書から一文字ずつの画像に切り出して，出力する必要がある．そこで，桂氏の切り出しでは三文字から二文字，二文字から一文字の処理で，三文字画像からの切り出しを行っている．

3.2.1　三文字から二文字

図5のように前処理の過程は，三文字画像を二値化させる．次に二値化させた三文字画像を横に合計を取ったベクトルで表す．ベクトルが山状になっている部分は文字部分となっているため，平の部分が文字同士の切れ目となっている．

そこで一つ目のステップとして，文字同士の切れ目を利用して一文字目と二文字目を含んだ画像 1 と二文字目と三文字目を含んだ画像 2 に分解する（図6）．

3.2.2　二文字から一文字

画像 1 と画像 2 に対して，横に合計を取ったベクトルで表す．さらに，そのベクトルに対して標準正規分布の確率密度関数を掛ける．画像の真ん中になる

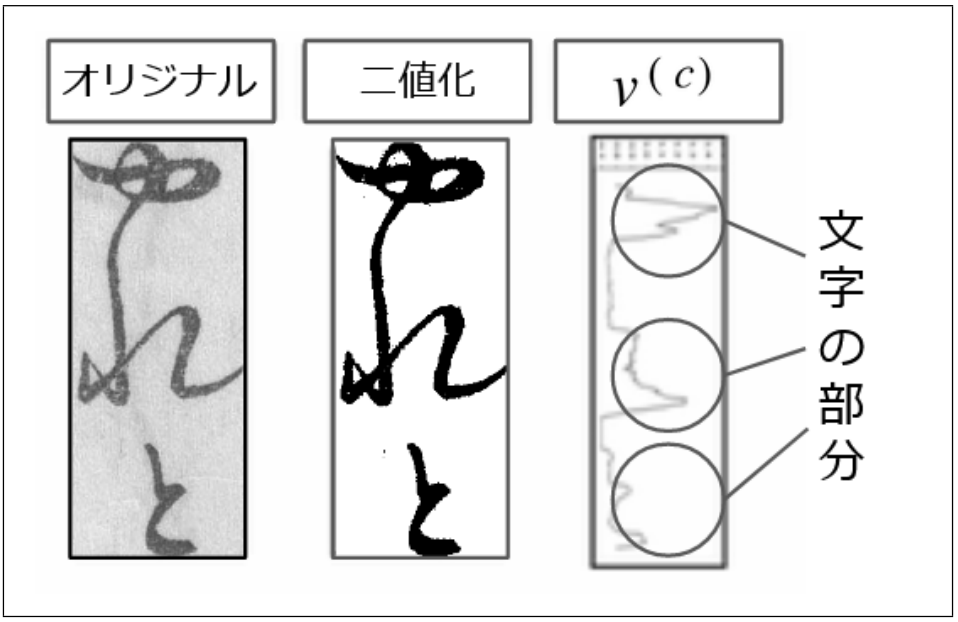

図 5　文字の切れ目

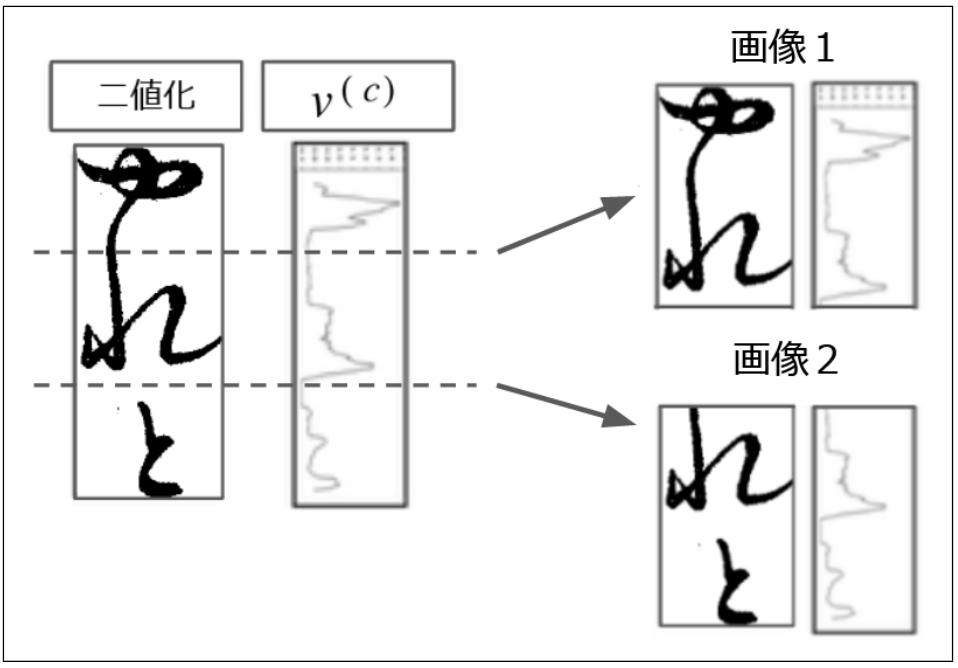

図 6　三文字から二文字への分解

ほど掛かる値が低いので文字同士の切れ目の部分が一番低い谷の部分になっている (図 7).

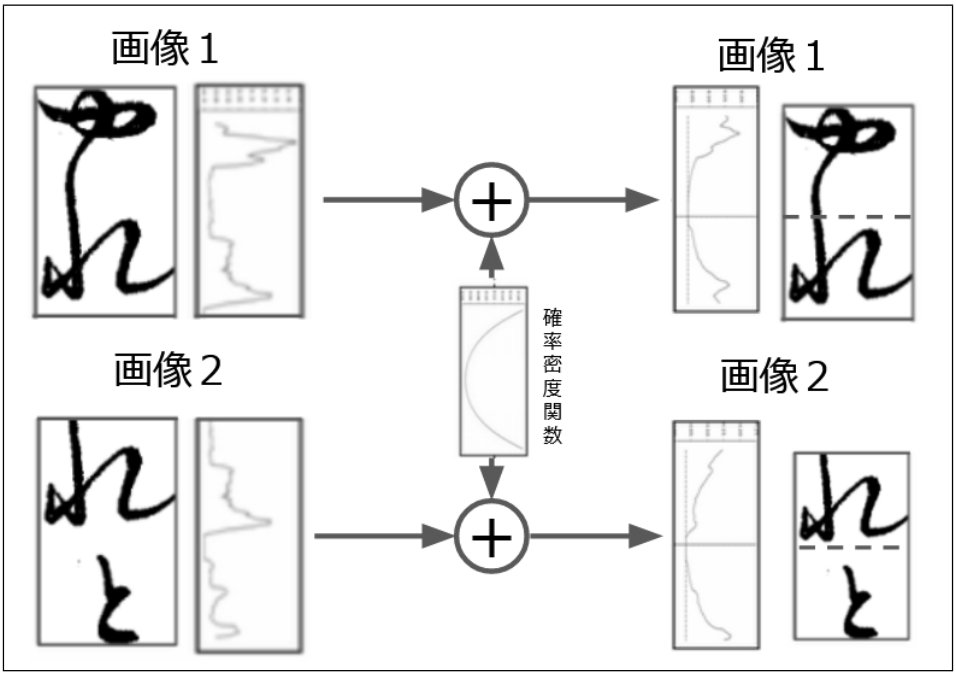

図７　二文字から一文字への分解

4　提案手法

従来システムの切り出し方法では，文字数制限をすることで動作していた．しかし，翻刻初学者にとって，一枚の古文書画像内に何文字含まれているか分からないため，翻刻初学者が従来システムを使用するのは手間がかかる．そこで，何文字（以降，N 文字とする．）でも対応する新たな切り出し方法を示すことで，入力画像の文字数制限を緩和する．本章では，従来の切り出し方法の欠点から課題を見つけ，新たな切り出し方法を提案する．

4.1　従来切り出し方法の欠点

従来の方法の文字数制限（N=3）の場合は，横軸の黒ピクセル数の総和グラフに対して，確率密度関数を掛けることで文字の境を強調し一文字の大きさ（高さ）を出していた．これにより，文字同士が繋がっているつなげ字でも，安定して切り出すことができた．

しかし，図 8, 図 9 のように N 文字の場合は，入力画像の大きさ（高さ）に対して，一文字がどれくらいの大きさ（高さ）であるかグラフから判断できないと入力画像に対して関数を掛けることが難しい．

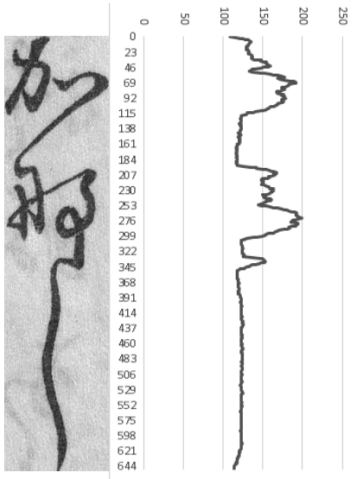

図 8　三文字のグラフ（N = 3）

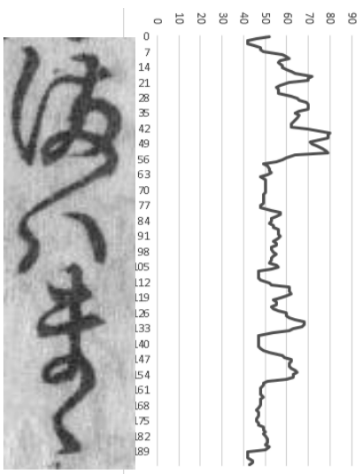

図 9　四文字のグラフ （N = 4）

そこで，提案する手法ではグラフ的処理にするのではなく，画像の特徴から一文字ずつに切り出す方法を提案する．

4.2　提案手法

提案手法では，つなげ字の特徴を利用して切り出す．つなげ字は，文字と文字を繋げるために，一文字目の右下から二文字目の左へ流れる特徴がある．本稿では，この流れを“はらい”とする．図10のようにはらいを検出するプログラム（以降，はらい検出プログラムとする．）を作成することで，文字同士を繋ぐ“つなげ線”を見つける．

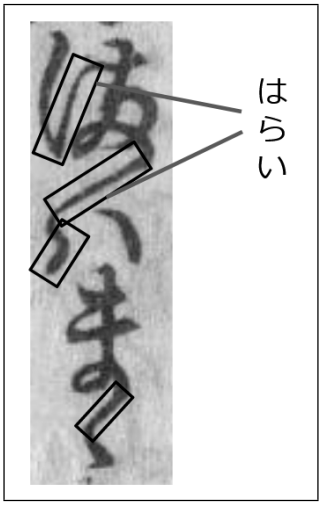

図10　はらい検出（イメージ図）

はらい検出プログラムでは，文字自体のはらい，文字同士のはらい（以降，つなげ線）を検出する．本研究では，つなげ線のみを取り扱うので，文字自体のはらいを候補から外すプログラムが必要である．

そこで，行ごとの黒の線を検出するプログラムを作成する．図11のように行ごとの黒の線を検出するプログラムにより，文字自体のはらいの部分は取り除くことができ，文字同士を繋げるためのつなげ線のみを取り出すことができる．

最後に，このつなげ線を消す処理を加えることで，一文字切り出しをすることができる．

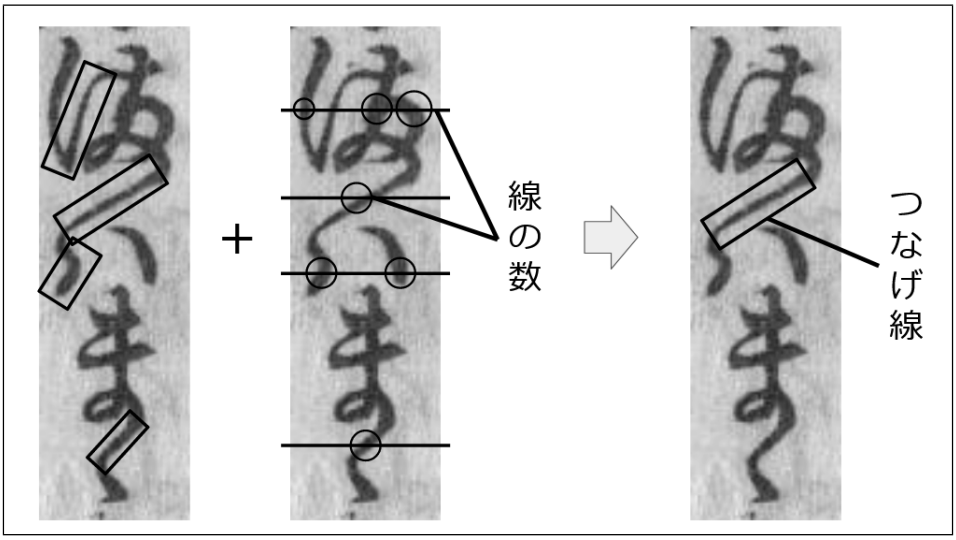

図 11　つなげ線のみの抽出

5　実現方法

本研究で提案する切り出しルーチンのために，はらい検出プログラムと行ごとの黒線検出プログラムを作成する．本章では，切り出しプログラムの全体像を示してから，各処理について詳しく説明する．

5.1　提案手法の全体像

提案する切り出しルーチンの過程として，図 12 より，前処理，はらい検出プログラム，行ごとの黒線検出プログラム，アンサンブル，出力がある．

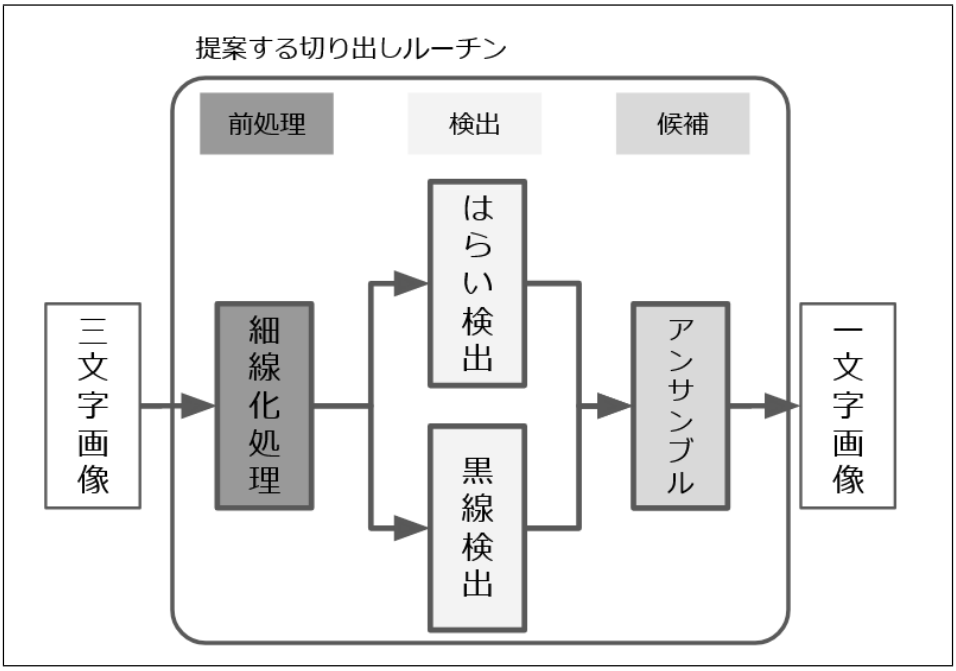

図 12　提案する切り出しルーチン（全体像）

入力の三文字画像に対し，細線化処理することで古文書の筆圧による文字の太さの変化をなくす．細線化後の出力画像に対し，はらい検出，黒線検出を行い，互いの候補をアンサンブルすることによって，一文字画像を出力する．これにより，古文書の特徴である“つなげ線”を使った切り出し方法を実現する．

5.2　前処理について

前処理では，入力画像に対して，細線化処理することで古文書の筆圧による文字の太さの変化をなくす．細線化のアルゴリズムとしては，Zhang-Suen のアルゴリズム [11][12] を使用する．

5.3　はらい検出プログラムについて

はらい検出プログラムでは，細線化した画像を使用する．はらいを検出する方法としては，画像の黒ピクセルの移動で判断する．移動では，次の図 13，図 14 である下と左下へ移動する二パターンが考えられる．

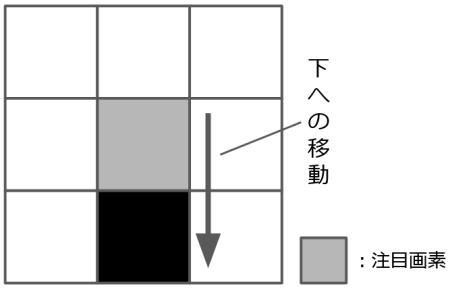

図 13　下への移動

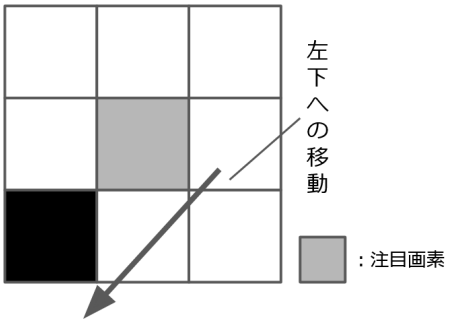

図 14　左下への移動

この二パターンのいずれかの移動を連続して，行った場合にその複数ピクセルをはらいとみなす．複数ピクセルの最後の行がはらいの最後の位置である候補とする．本研究では，この連続した移動回数の最適解を細線化した画像のピクセルから法則を目視で見つける．

5.4　行ごとの黒線検出プログラムについて

行ごとの黒線検出プログラムでは，細線化した画像を使用する．行ごとの黒線検出する方法としては，行ごとの黒ピクセルを数える．細線化により，文字の線は次のような図 15 になる．

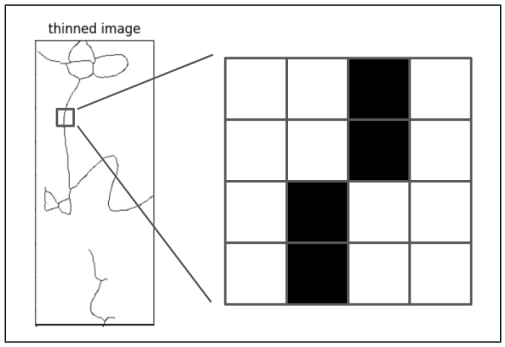

図 15　細線化によるピクセル

これにより，行ごとに見ていき，黒ピクセル数が一つであるところがつなげ

線である候補となる.

6　候補のアンサンブル

はらい検出プログラム，黒線検出プログラムにより出てきた各候補からつなげ線を見つけるために，候補同士をアンサンブルする．アンサンブル方法としては，各候補の共通する行をつなげ線とする．出力されたつなげ線の行の黒ピクセルを削除することにより，文字同士の繋がりを削除する．削除後，行ピクセルごとに見ていき，白ピクセルの行で切り出す．その際，行ごとの白ピクセルが連続している場合は，連続した行の中央を切り出す(図 16)．

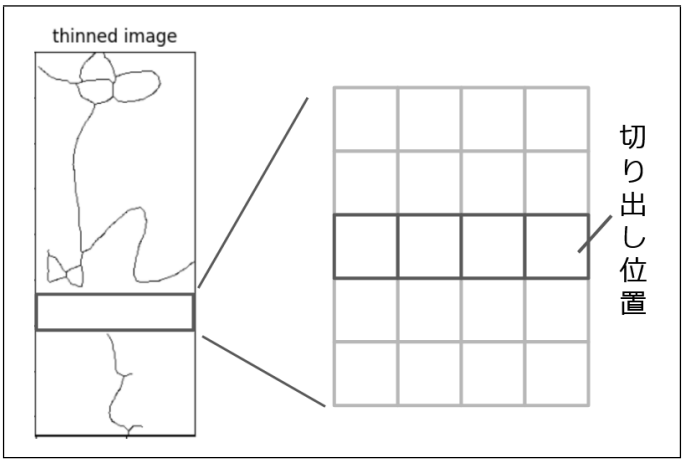

図 16　連続した場合の切り出し方法

7　研究結果

本章では，提案手法のプログラムを作成し，実行した結果を示す．

7.1　前処理について

入力画像に対して，細線化した画像は次の図17に示す．

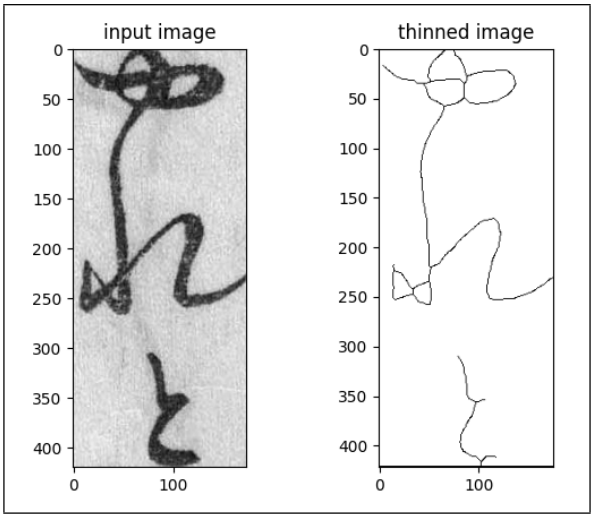

図17　細線化画像

図より，筆圧による文字の太さの変化をなくすことができた．

7.2　はらい検出プログラムの結果

はらい検出プログラムでは，連続した移動回数の最適解を目視により見つけた．本研究では，下への移動を5回，左下への移動を5回(順序は問わない)したものをはらいとする．検出したはらい候補を図18に示す．はらいが検出された場所の行を灰色の線で塗られている．

7.3　黒線検出プログラムの結果

前処理により，行ごとの黒ピクセル数を見ることで黒線の検出することができた．次の図19は，行ごとで黒ピクセル数が一つしかない行を灰色の線で塗られている．

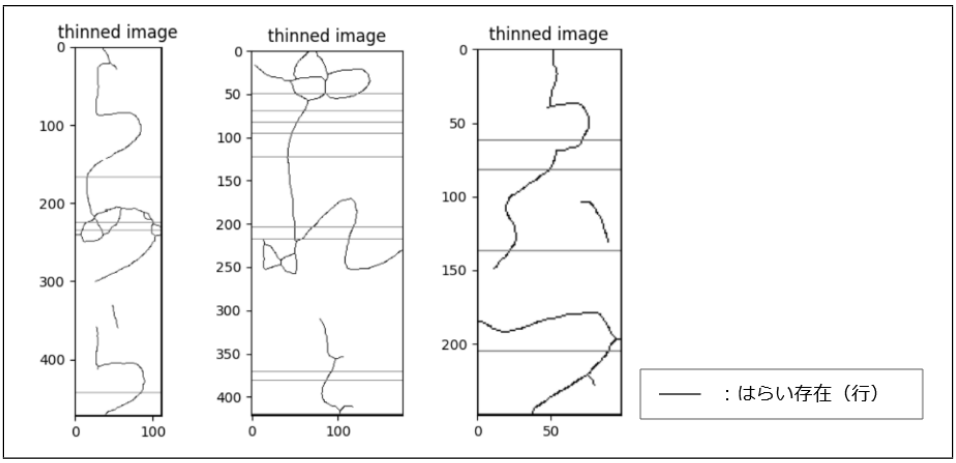

図 18　画像の中のはらい

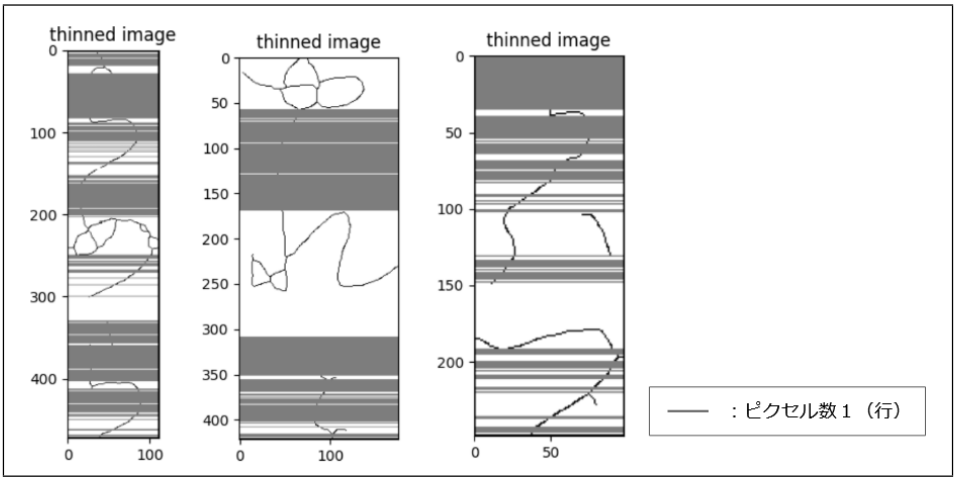

図 19　行における黒ピクセル数 1 の部分

7.4　アンサンブル結果

はらい検出プログラムの候補と黒線検出プログラムの候補をアンサンブルした結果の切り出し候補を次の図 20 に示す．切り出し候補が一つのはらいにつき，一つであると切り出しが正しく動作する．

7.4.1　切り出し画像の成功

三文字古文書画像に対して，一文字ずつの古文書画像を得ることができた．切り出し画像の成功例として，次の図 21 を示す．

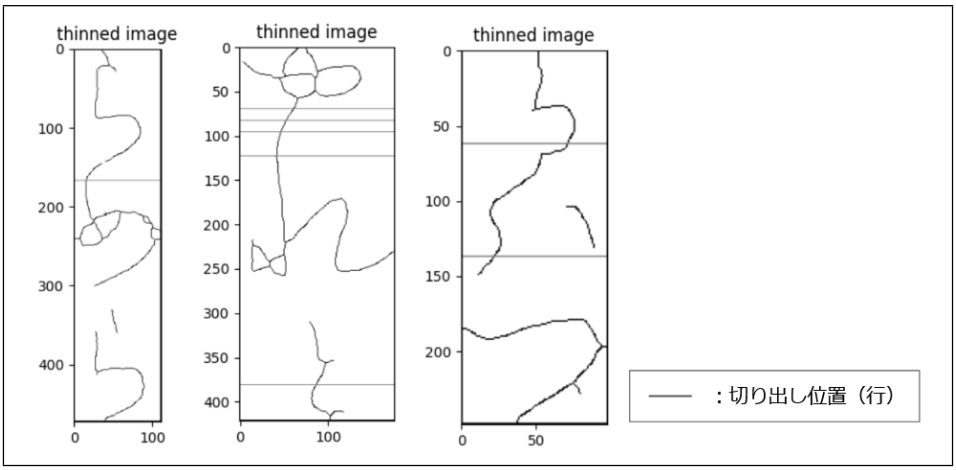

図 20　切り出し位置の出力

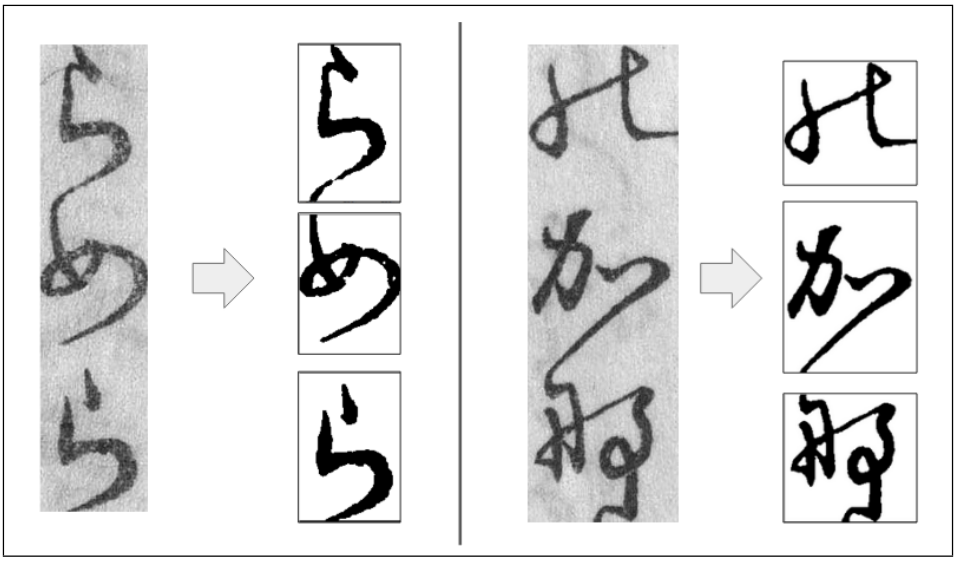

図 21　切り出し成功例

7.4.2　切り出し画像の失敗

三文字古文書画像に対して，三枚以上の切り出し画像を得たときの失敗例を挙げる．次の図 22 は，つなげ字が長いためにはらい検出を何回も受け，出力された．

また，別の失敗例として，「と」や「い」のような一文字自体にはらい検出され，行の黒ピクセル数 1 のために切り出しがうまく動作しない例がある（図 23）．

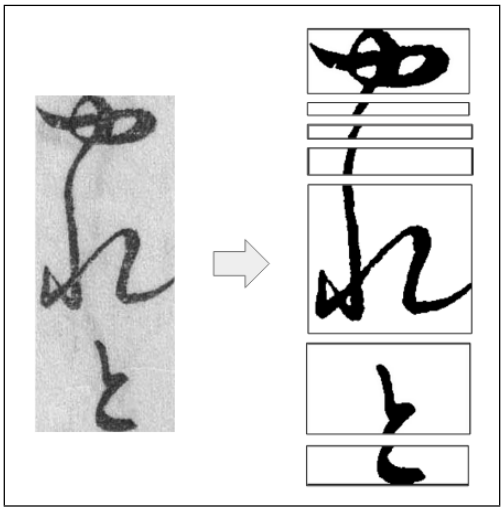

図 22　長いつなげ字による出力

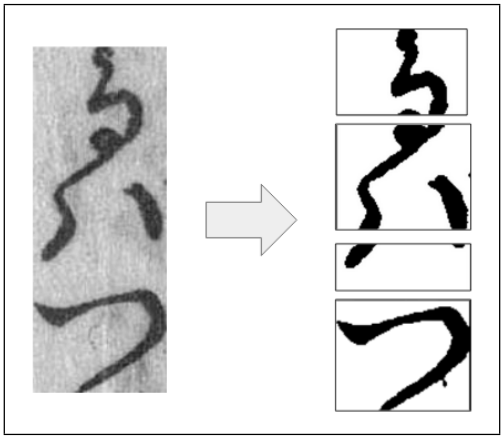

図 23　一文字かつピクセル数 1 の動作

8　考察

本研究では，はらい検出プログラムの連続移動回数の制限を目視で法則性を見つけて，実装した．目視による決め方では，三文字古文書画像に対してできるだけ候補を少なくする．下移動を5回，左下移動を5回した動作をはらいとした．これにより，従来の手法と同じ出力画像を得ることができ，グラフ的な処理でなくても画像切り出しを行うことができることが分かった．しかし連続移動回数は自ら決めていることもあり，一つの大きなはらいについて，二つ検出してしまうことが起きた．大きなはらいの中に小さなはらいが二つ検出されることにより，出力される結果で，はらいの部分が細切れになる．また，他の失敗例として，文字単体で行ごとの黒ピクセル数が１つなために，黒線検出プログラムでは，省けないようになってしまう．そこで，これらの失敗例を防ぐために，次の二点の方法が考えれる．

一つ目は，機械学習による連続移動回数の制限数を決めることである．教師あり学習によって，連続移動回数を可変させ最適解を求めることで，正しいはらいを検出できるのではないかと考える．

二つ目は，はらいの検出をより細かくし，クラスタリングによって切り出し候補数を削減する方法である．候補値付近を一つの候補にすることで出力時の画像の細切れを防ぐことができると考える．

以上の二つの処理を加えることで，はらい検出プログラムの最適回数と安定した切り出しが行えるようになる．

9　あとがき

日本の近世古文書を解読できる人々の数を増やすべく，近世古文書の翻刻初学者に対してのシステム向上を目指した．現在のシステムでは，入力画像の文字数が制限されており，本研究では，文字数制限を無くすために前処理である文字切り出しの新しい手法の提案をする．従来の手法では，文字数を制限し，入力画像に対してグラフ的な処理で切り出した．そこで，新しい手法では古文書の特性を活かし，古文書内のはらいを見つけることで文字同士を繋げるつなげ線を消すことができるのではないかと考えた．そこで，本研究では新たな提案手法を実装した結果からグラフ的な処理でなくとも切り出すことができることを示す．

実装では，入力画像に対し，細線化処理することで古文書の筆圧による文字の太さの変化を軽減させた．はらい検出プログラムでは，黒ピクセルの移動について注視して，左への移動を5回，左下への移動を5回したものをはらいとして取り扱った．さらに，文字の中のはらいを省くために，行ごとの黒線検出プログラムにより，はらいである行かつ一線しかないものを見ることで切り出し位置を見つけた．

動作として，大きなはらいが画像内に出現した場合，小さなはらいを複数個検出してしまい，出力画像が細切れになるという動作になった．また，「と」や「い」，「し」などの文字自体に横ピクセル数が1つになる文字は黒線検出プログラムによって，切り出し候補を省くことができなかった．

今後の課題としては，はらい検出プログラムの最適回数を教師あり機械学習によって求める．はらい検出をより細かくして，多くの候補を出し，クラスタリングによって切り出し候補を出すなどの方法が考えられる．

10 謝辞

本研究をするにあたって，プログラムの使用許諾を出してもらった桂尚輝氏，また論文の考査を頂いた松尾先生，岡村先生，発表への質問して頂けた松村先生，上野先生，岩田先生，市川先生，研究に付き合っていただけた五年生のみなさまに感謝の意を述べます．

また，研究を続けて行く上でご迷惑をおかけすると思いますが，何卒宜しくお願い致します．

参考文献

[1] 北本朝展, カラーヌワット・タリン, 宮崎智, 山本和明 : “文字データの分析” (1996).

[2] 橋本雄太 : “Ai 文字認識とクラウドソーシングを組み合わせた歴史資料の大規模テキスト化” (2020).

[3] 川上紳一, 高野雅夫, 小川克郎, 熊澤峰夫, 瀬野徹三 : “データ取得とモデリング” (1996).

[4] 静岡県歴史文化情報センター : “くずし字解読講座 テキスト一覧”, https://www.tosyokan.pref.shizuoka.jp/contents/history/kuzushi.html.

[5] 橋本雄太 : “「みんなで翻刻」による翻刻テキストの分析の試み”, 「人文科学とコンピュータシンポジウム」 (2018).

[6] 京都大学古地震研究会 : “みんなで翻刻”, https://honkoku.org/ (2017).

[7] 奈良文化財研究所, 東京大学史料編纂所 : “木簡・くずし字解読システム”, https://mojizo.nabunken.go.jp/ (2016-03).

[8] 北本朝展, カラーヌワット・タリン, 宮崎智, 山本和明 : “文字データの分析——機械学習によるくずし字認識の可能性とそのインパクト——”, 電子情報通信学会誌, **102**, 6 (2019-06).

[9] 北本朝展 : “データ駆動型人文学研究の発展と ai によるくずし字認識”, 月間 J − LIS, **6**, 8 (2019-11).

[10] 電子情報通信学会 : “第 23 回 prmu アルゴリズムコンテストくずし字認識チャレンジ 2019”, https://sites.google.com/view/alcon2019 (2019).

[11] ROSETTACODE.ORG: “Zhang-suen thinning algorithm”, http://rosettacode.org/wiki/Zhang-Suen_thinning_algorithm (last modified on 5 January 2021).

[12] hrs1985: “Python で zhang-suen の細線化アルゴリズムを実装”, https://qiita.com/hrs1985/items/7751d4b5241d5c314a6d (2020).