論文

画像処理の古写真への応用

Application of image processing to old photographs

畔津忠博, 倉田研治
AZETSU Tadahiro, KURATA Kenji
山口県立大学国際文化学部
Faculty of Intercultural Studies, Yamaguchi Prefectural University

要　旨

本稿では、画像処理の古写真への応用について議論する。具体的な応用として、立札に記載された文字の可読性の向上と既存の深層学習に基づいた方法で着色された画像の品質改善を試みる。また、画像処理にPythonを用いることで、オープンソースのソフトウェアのみで処理が完結できることを目指す。

Abstract:

In this article, we discuss the application of image processing to old photographs. As concrete applications, we attempt to enhance the legibility of the characters on a notice board and to improve the quality of the image colored using the existing deep learning-based method. In addition, by using Python for image processing, we aim to complete the process using only open-source software.

キーワード: 古写真、画像処理、画像強調、色空間、Python
Key words: Old photographs, Image processing, Image enhancement, Color space, Python

1　はじめに

近年、デジタルデバイスの進化により、デジタル画像を容易に取り扱うことが可能になっている。デジタル画像処理には、画像認識、画像補正、画像ノイズ除去、画像加工など多くのものがあり、また、医用画像処理、衛星画像処理、製品検査、個人認証など実社会においても様々な場面で利用されている。

画像処理を行うソフトウェアには優れたものが多くあり、商用のものオープンソースのものなどユーザーが目的に応じて自由に選択できる。また、基本的な画像処理においてはアルゴリズムも簡単であり、ユーザーが処理内容をプログラミング言語で記述することも容易に行える。そのため、デジタル画像処理は、データサイエンス教育の題材としても適していると考えられる。

今回は、画像処理を用い古写真において、(i)立札の画像に記載されている文字を見やすくできるかという試みと、(ii)既存の深層学習に基づいた方法で着色された画像に対してカラー画像強調を適用することで、品質の改善が行えるかという試みを実施する。また、オープンソースのもので処理が完結できることを考慮する。

2　古写真のデータベース

インターネットと関連する技術の発達により、様々な資料がデジタルアーカイブされている。本学でも、寺内文庫に関連する貴重な資料が所蔵されており、2017年から古写真コレクションをはじめとしたデジタルアーカイブを進めている。ここでは、古写真のデータベースについての現状を紹介する。

古写真のデータベース構築作業は、1000点に及ぶ古写真、古文書、拓本などを対象として、資料整理、

データ化、Webサイト構築・運営などを実施している。あわせて、研究補助をする学生は、アーカイブ作業のプロセスから、実践的に資料の取り扱いやデータ制作を学ぶ機会となっている。工程は、適宜教員による専門的なチェックを行い、データベース構築を進めた。2021年11月に、資料を項目ごとに閲覧できる桜圃寺内文庫アーカイブス（https://terauchi-archives.kura-ken.net）を公開した。古写真アーカイブは順次更新し、所蔵品はほぼ公開を完了した。現在は、拓本編を公開し、古文書編の公開も2022年５月以降より段階的に進めている。

具体的な作業内容としては、各資料の整理とデータ化、Webサイトの運営がある。データ化では、古写真の場合はほとんどがプリントの為、フラッドヘッドスキャナーによるスキャン作業となった。対象の大きさにより調整し、原寸サイズで300～600dpiのTIFFデータを基本フォーマットとした。一覧に整理した台帳データと写真プリントを確認して、冊子ごとに取りまとめを行った。

古文書、拓本などスキャンできない形状や大きなサイズなどの物は、デジタルカメラによる複写でデータ化した。撮影データは、RAWデータとJPGデータを併用し、基本フォーマットはRAWデータとし、Webサイトなどの閲覧用には、JPGデータを使用している。

Webサイトの運営では、コンテンツ・マネジメント・システム（以下、CMS）のBiNDによりサイト構築を進めている。パッケージ型CMSは、コストとセキュリティの視点から採用を検討した。オープンソース型 CMSではサポートされないシステムメンテナンスやセキュリティの脆弱性などを回避しつつ、持続的な運営を想定して選択した。他の要因として、サイトデザインなどの管理に関して、HTML、CSS等のマークアップ言語やJavaScript、PHP等のプログラミング言語の利用を回避することで、運用が煩雑にならない仕様とした。運用するスタッフや学生は、平易な手順を学ぶことで、Webサイト制作や運用などのデータベース管理が可能となる。

データベース構築の先行事例としては、長崎大学付属図書館の長崎大学電子化コレクションがある。大学図書館所蔵の古写真や資料をアーカイブ化することで、所蔵品の価値や活用方法が示された仕様になっている。一過的なプロジェクトではなく、データ集積から公開まで、堅実な内容が持続的に取り組まれている。例えば、コレクション内の日本古写真データベース（http://oldphoto.lb.nagasaki-u.ac.jp/top/jp_top.php）では、現在でもWebサイトのユーザビリティが改善されており、撮影者で検索、撮影対象で検索、撮影地で検索など、属性情報を活用した検索方法や画面表示がアップデートされている。持続的な運用や、利活用を想定した姿勢が感じられる。この他に先行する機関として、東京都写真美術館や国立国会図書館など、写真プリントや資料を保存、アーカイブを先進的に進めている機関の取り組みを注視し、時代に相応した環境整備を進める必要がある。

WebサイトやSNSによる情報発信は、先行する他大学や機関の取り組みからも、組織の付加価値や特徴を示す重要な役割を担っていることが分かる。本学であれば、寺内文庫の価値を示し再評価されることは、大学の特色や地域貢献のひとつとなる。

本稿で用いる古写真は、桜圃寺内文庫アーカイブスの古写真編[1, 2] 及び防長尚武館所蔵の寺内文庫に関する写真を使用した。古写真のデータベースや地域に残る古写真から、画像処理による新たな活路を探っていく。

3　画像処理の古写真への応用の例

3.1　使用するソフトウェア

画像処理を行うためのアプリケーションソフトは数多く存在するが、本稿ではプログラミング言語を用いて処理を行う。画像処理に用いられている言語の１つにMATLABがある。MATLABは数値解析のための商用のソフトウェアであり、非常に使いやすく、目的に応じて幅広いツールボックスが用意されており、また、多次元配列に対して高速な処理が可能である。さらに、多くの書籍やインターネット上における様々な情報があり、習得するための環境も整っている。

ただし、本稿では、授業等でも利用しやすい点を考慮してPythonを用いることとし、数値解析を行うためのモジュールや高機能なエディタが組み込まれているディストリビューションであるオープンソースのAnacondaを利用する。また、本稿で行うような基本的な画像処理においては、MATLABのコードをPythonのコードに書き換えることは、非常に容易である。

3.2　応用例

今回は画像処理の古写真への応用として、(i)図1

(a)の立札の画像に画像処理を適用することで、そこに記載されている文字を見やすくできるかという試みと、(ii)図２～４(a)の画像を既存の深層学習に基づいた方法を用いて着色し、その結果画像に対してカラー画像強調を適用することで、品質の改善が行えるかという試みを実施する。

3.3　色空間

まず、(i)について説明する。文字の部分を強調するためには幾つかの方法が考えられるが、ここでは基本的な処理として、明度成分にコントラスト強調を適用する。

画像処理を行うときに、色空間の選択は重要な要素の１つである。色空間には、処理の用途に応じて様々なものが存在する。デジタルデバイスの入出力における表現など幅広く利用されるRGB色空間、人間の視細胞の働きを表現するLMS色空間、色相・彩度・明度を成分にもつHSI色空間、空間の距離が人間の知覚にできるだけ合うように考案された均等色空間などがある。

一般的にデジタル画像はRGB色空間で保存されているため、明度成分は独立していない。そのため、明度成分をもつ色空間に変換する。明度成分をもつ色空間として、ここでは、均等色空間の1つであるCIELAB色空間を利用する。CIELAB色空間は、明度L^*、赤-緑クロマティック指数a^*、黄-青クロマティック指数b^*の各成分から構成される。L^*は０から100までの値をとり、0は黒、100は白を示す。また、a^*の正負で赤味、または、緑味が強いことを示し、b^*の正負で黄味、または、青味が強いことを示す。色相と彩度の値は、a^*とb^*から求めることができる。

3.4　立札の文字の可読性の向上

コントラスト強調にはトーンカーブを利用する。トーンカーブは入出力を表現する関数であり、入出力の関係を調整することで、様々な処理が可能となる。ここでは、明度L^*にコントラスト強調を適用することで、文字を見やすくする。

アルゴリズムとしては、以下の通りである。

(1)　RGB色空間の座標(r, g, b)をCIELAB色空間の座標(L^*, a^*, b^*)に変換する。

(2)　CIELAB色空間の明度L^*を次式で表されるトーンカーブにより強調する。

$$L_e^* = \begin{cases} 0 & x < \mathrm{MIN} \\ 100\left(\dfrac{L^* - \mathrm{MIN}}{\mathrm{MAX} - \mathrm{MIN}}\right) & \mathrm{MIN} \le x \le \mathrm{MAX} \\ 100 & x > \mathrm{MAX} \end{cases}$$

ここで、MINとMAXはコントラスト強調の度合いを決めるパラメータである。

(3)　明度が強調されたCIELAB色空間の座標(L_e^*, a^*, b^*)をRGB色空間の座標(r_e, g_e, b_e)に変換する。

(4)　(r_e, g_e, b_e)の各成分において、最小値０より小さいときは０に最大値255より大きいときは255に置き換える。

図１(b)に結果画像を示す。パラメータのMINとMAXは、それぞれ70と90に設定した。明度のコントラストを強調することで、文字の可読性が向上していることがわかる。ただし、文字に関しての様々な情報を用いることで、さらに精度を向上させる必要がある。

3.5　言語の比較

上記のアルゴリズムをMATLABとPythonで記述した例を、それぞれ表１と表２に示す。Pythonでは、画像処理のためのライブラリであるOpenCVと数値計算を行うためのモジュールであるNumpyを利用する。表１と表２を比較すると、多くの点で類似していることがわかる。主な違いは以下の通りである。ただし、Pythonと記述しているときは、モジュールなど拡張機能を含めた使い方を表している。

・Pythonでは、行頭からの空白であるインデントに意味もたせているため、インデントを正確に入力する必要がある。インデントは、ブロックごとにスペース4文字分が推奨されている。

・配列の要素の表現が異なる。MATLABではx(i)、Pythonではx[i]となり、括弧の種類が異なる。また、要素番号の開始はMATLABでは１からPythonでは０からである。多次元の場合も同じである。

・ユーザー定義関数の書式が異なる。関数名をf、関数の入力をx、出力をyとすると、
MATLABでは、

```
function y=f(x)
処理内容
end
```

と記述し、関数を別のファイルとして保存して、呼び出すようにする。あるいは、MATLABの

バージョンがR2016b以降の場合は、表1のように1つのコードにまとめることができる。そのときは、基本的にコードの下側にユーザー定義関数をまとめる。

Pythonでは、

```
def f(x):
    処理内容
    return y
```

と記述する。defの右側に「:」をつけ、処理内容の部分はインデントをつける。また、出力はreturn命令により返す。ユーザー定義関数は表2のように基本的にコードの上側にまとめる。

・算術演算子が一部異なる。四則演算は同じであるが、べき乗は、MATLABでは「^」、Pythonでは「**」である。
・行列演算のための演算子が異なる。行列の積はMATLABでは「*」、Pythonでは「@」である。また、行列の転置はMATLABでは「.'」、Pythonでは「.T」である。
・配列演算を行うとき、MATLABでは「.」演算子を用いる。

他にも若干の違いはあるが、微修正により互いの書き換えが可能である。

3.6 着色画像の品質改善

モノクロ画像を着色する方法としては様々なものが提案されてきたが[3]、ここでは既存の深層学習に基づいた方法[4, 5]を用いる。このZhangらの方法はPythonでソースコードが公開されており[6]、Pythonを使える環境であれば、すぐに結果画像が得られる。この結果画像に対して、カラー画像強調を行って品質の改善を試みる。

カラー画像強調には、先ほどと同じくCIELAB色空間を用いる。アルゴリズムとしては、以下の通りである。

(1) RGB色空間の座標(r, g, b)をCIELAB色空間の座標(L^*, a^*, b^*)に変換する。

(2) CIELAB色空間のa^*とb^*を次式により強調する。

$$(a_k^*, b_k^*)=k(a^*, b^*)$$

ここで、kは1より大きい正のパラメータである。両方の成分をk倍するのは、色相を変化させないためである。これにより、もとの彩度C^*が強調された彩度C_k^*になる。また、色相θは、$\theta=\tan^{-1}(b^*/a^*)$で不変である。

(3) 明度を次式のガンマ補正により強調する。

$$L_e^*=100(L^*/100)^{1/\gamma}$$

ここで、γはパラメータであり、1より大きいと低い明度を向上させる効果がある。

(4) CIELAB色空間における強調された明度と彩度(L_e^*, C_k^*)をRGB色空間に変換したときに、RGB色空間の色域に収まっているかどうかで下記の場合分けを行う。

$$C_e^*=\begin{cases} C_k^* & \text{色域内} \\ C_{\mathrm{MAX}}^* & \text{色域外} \end{cases}$$

ここで、C^*_{MAX}は色相θの条件で$(L_e^*, C^*_{\mathrm{MAX}})$をRGB色空間に変換したときに色域内に収まる最大の彩度である[7]。

(5) 彩度と明度が強調されたCIELAB色空間の座標(L_e^*, C_e^*)をRGB色空間の座標(r_e, g_e, b_e)に変換する。

図2～4に結果を示す。図2～4(b)は、Zhangらの方法で着色した画像、図2～4(c)は、(b)の画像にカラー画像強調を適用した結果である。パラメータのkとγは、それぞれ3と1.5に設定した。図2～4(c)から画像の彩度、明度は、それほど違和感なく強調されていることがわかる。ただし、図4(c)のように、図4(b)で着色が適切にされていない部分に色の不連続性もみられ限界もある。

4 おわりに

本稿では、画像処理の古写真への応用について幾つかの例とともに議論した。立札に記載された文字の可読性については、明度のコントラストを強調することで向上した。また、既存の深層学習に基づいた方法で着色された画像について、カラー画像強調を適用することで品質改善の可能性が示された。ただし、今回は基本的な画像処理のみを用いており、さらに幅広い応用が考えられる。

画像処理にPythonを用いることで、オープンソースのソフトウェアのみで処理を完結することができ、教育用の題材の1つとしても検討できることを示した。

参考文献

[1] 倉田研治, 「GIS・3Dデータによる立体情報アーカイブ －寺内文庫の旧本館・朝鮮館を素材として－」, 山口県立大学基盤教育紀要, 第2号, pp.145-150, 2022年.
[2] 山口県立大学図書館所蔵桜圃寺内文庫アーカイ

ブス, https://terauchi-archives.kura-ken.net/ (閲覧日 2022年9月20日)
[3] 雨車和憲, 半谷精一郎, 「カラリゼーション―白黒画像・映像を自動でカラー化する―」, IEICE Fundamentals Review, vol.11, no.3, pp.186–192, 2018.
[4] R. Zhang, P. Isola, A. A. Efros, Colorful image colorization, European Conference on Computer Vision (ECCV), pp.649–666, 2016.
[5] R. Zhang, J.-Y. Zhu, P. Isola, X. Geng, A. S. Lin, T. Yu, A. A. Efros, Real-time user-guided image colorization with learned deep priors, ACM Transactions on Graphics, vol.36, no.4, pp.1-11, 2017.
[6] Colorful Image Colorization Project Page, http://richzhang.github.io/colorization/ (accessed on 31 Aug. 2022).
[7] T. Azetsu, N. Suetake, Chroma enhancement in CIELAB color space using a lookup table, Designs, 5, 32, 2021.

表1. MATLABによるコード

```
rgb_inp=imread('image.jpg');
rgb=double(rgb_inp)/255;
[cx,cy,cc]=size(rgb);
rgb=reshape(rgb,cx*cy,3);
rgb=inv_rgb_gamma(rgb);
xyz=rgb2xyz(rgb);
lsasbs=xyz2lsasbs(xyz);
lsasbs(:,1)=100*tone_map_inc(lsasbs(:,1)/100,0.7,0.9);
lsasbs=reshape(lsasbs,cx*cy,cc);
xyz_out=lsasbs2xyz(lsasbs);
rgb_out=xyz2rgb(xyz_out);
rgb_out=rgb_gamma(rgb_out);
rgb_out=255*rgb_out;
rgb_out(rgb_out>255)=255;
rgb_out(rgb_out<0)=0;
rgb_out=uint8(fix(rgb_out));
rgb_out=reshape(rgb_out,cx,cy,cc);
figure; imshow(rgb_inp,'Border','tight');
figure; imshow(rgb_out,'Border','tight');

function rgb2=inv_rgb_gamma(rgb)
rgb2(rgb(:,1)<=0.03928,1)=rgb(rgb(:,1)<=0.03928,1)/12.92;
rgb2(rgb(:,2)<=0.03928,2)=rgb(rgb(:,2)<=0.03928,2)/12.92;
rgb2(rgb(:,3)<=0.03928,3)=rgb(rgb(:,3)<=0.03928,3)/12.92;
rgb2(rgb(:,1)>0.03928,1)=((rgb(rgb(:,1)>0.03928,1)+0.055)/1.055).^2.4;
rgb2(rgb(:,2)>0.03928,2)=((rgb(rgb(:,2)>0.03928,2)+0.055)/1.055).^2.4;
rgb2(rgb(:,3)>0.03928,3)=((rgb(rgb(:,3)>0.03928,3)+0.055)/1.055).^2.4;
end

function rgb2=rgb_gamma(rgb)
rgb2(rgb(:,1)<=0.00304,1)=12.92*rgb(rgb(:,1)<=0.00304,1);
rgb2(rgb(:,2)<=0.00304,2)=12.92*rgb(rgb(:,2)<=0.00304,2);
rgb2(rgb(:,3)<=0.00304,3)=12.92*rgb(rgb(:,3)<=0.00304,3);
rgb2(rgb(:,1)>0.00304,1)=1.055*rgb(rgb(:,1)>0.00304,1).^(1/2.4)-0.055;
rgb2(rgb(:,2)>0.00304,2)=1.055*rgb(rgb(:,2)>0.00304,2).^(1/2.4)-0.055;
rgb2(rgb(:,3)>0.00304,3)=1.055*rgb(rgb(:,3)>0.00304,3).^(1/2.4)-0.055;
end
```

```
function xyz=rgb2xyz(rgb)
A = [0.4124 0.3576 0.1805; 0.2126 0.7152 0.0722; 0.0193 0.1192 0.9505];
xyz = rgb*A.';
end

function rgb=xyz2rgb(xyz)
A=[3.2410 -1.5374 -0.4986; -0.9692 1.8760 0.0416; 0.0556 -0.2040 1.0570 ];
rgb = xyz*A.';
end

function f=lsasbs_f(x,xn)
x=x/xn;
a=0.008856;
f(x>a)=x(x>a).^(1/3);
f(x<=a)=7.787*x(x<=a)+16/116;
end

function x=lsasbs_invf(f)
b=0.20689;
x(f>b)=f(f>b).^(3);
x(f<=b)=(f(f<=b)-16/116)/7.787;
end

function lsasbs=xyz2lsasbs(xyz)
xn=0.9505;yn=1.000;zn=1.089;
fx=lsasbs_f(xyz(:,1),xn);
fy=lsasbs_f(xyz(:,2),yn);
fz=lsasbs_f(xyz(:,3),zn);
lsasbs(:,1)=116*fy-16;
lsasbs(:,2)=500*(fx-fy);
lsasbs(:,3)=200*(fy-fz);
end

function xyz_out=lsasbs2xyz(lsasbs)
xn=0.9505;yn=1.000;zn=1.089;
fy=(lsasbs(:,1)+16)/116;
fx=lsasbs(:,2)/500+fy;
fz=-lsasbs(:,3)/200+fy;
xyz_out(:,1)=xn*lsasbs_invf(fx);
xyz_out(:,2)=yn*lsasbs_invf(fy);
xyz_out(:,3)=zn*lsasbs_invf(fz);
end

function y = tone_map_inc(x,MIN,MAX)
y=(x-MIN)/(MAX-MIN);
y(x<MIN)=0;
y(x>MAX)=1;
end
```

表2. Pythonによるコード

```
import numpy as np
import cv2

def inv_rgb_gamma(rgb):
    rgb2=np.zeros((rgb.shape[0],rgb.shape[1]),dtype=np.float64)
    rgb2[rgb[:,0]<=0.03928,0]=rgb[rgb[:,0]<=0.03928,0]/12.92
    rgb2[rgb[:,1]<=0.03928,1]=rgb[rgb[:,1]<=0.03928,1]/12.92
    rgb2[rgb[:,2]<=0.03928,2]=rgb[rgb[:,2]<=0.03928,2]/12.92
    rgb2[rgb[:,0]>0.03928,0]=((rgb[rgb[:,0]>0.03928,0]+0.055)/1.055)**2.4
    rgb2[rgb[:,1]>0.03928,1]=((rgb[rgb[:,1]>0.03928,1]+0.055)/1.055)**2.4
    rgb2[rgb[:,2]>0.03928,2]=((rgb[rgb[:,2]>0.03928,2]+0.055)/1.055)**2.4
    return rgb2

def rgb_gamma(rgb):
    rgb2=np.zeros((rgb.shape[0],rgb.shape[1]),dtype=np.float64)
    rgb2[rgb[:,0]<=0.00304,0]=12.92*rgb[rgb[:,0]<=0.00304,0]
    rgb2[rgb[:,1]<=0.00304,1]=12.92*rgb[rgb[:,1]<=0.00304,1]
    rgb2[rgb[:,2]<=0.00304,2]=12.92*rgb[rgb[:,2]<=0.00304,2]
    rgb2[rgb[:,0]>0.00304,0]=1.055*rgb[rgb[:,0]>0.00304,0]**(1/2.4)-0.055
    rgb2[rgb[:,1]>0.00304,1]=1.055*rgb[rgb[:,1]>0.00304,1]**(1/2.4)-0.055
    rgb2[rgb[:,2]>0.00304,2]=1.055*rgb[rgb[:,2]>0.00304,2]**(1/2.4)-0.055
    return rgb2

def rgb2xyz(rgb):
    A=np.array([[0.4124,0.3576,0.1805],[0.2126,0.7152,0.0722],[0.0193,0.1192,0.9505]])
    return rgb@A.T

def xyz2rgb(xyz):
    A=np.array([[3.2410,-1.5374,-0.4986],[-0.9692,1.8760,0.0416],[0.0556,-0.2040,1.0570]])
    return xyz@A.T

def lsasbs_f(x,xn):
    f=np.zeros((x.shape[0]),dtype=np.float64)
    x=x/xn
    a=0.008856
    f[x>a]=x[x>a]**(1/3)
    f[x<=a]=7.787*x[x<=a]+16/116
    return f

def lsasbs_invf(f):
    x=np.zeros((f.shape[0]),dtype=np.float64)
    b=0.20689
    x[f>b]=f[f>b]**(3)
    x[f<=b]=(f[f<=b]-16/116)/7.787
    return x

def xyz2lsasbs(xyz):
```

```
    lsasbs=np.zeros((xyz.shape[0],xyz.shape[1]),dtype=np.float64)
    xn=0.9505;yn=1.000;zn=1.089
    fx=lsasbs_f(xyz[:,0],xn)
    fy=lsasbs_f(xyz[:,1],yn)
    fz=lsasbs_f(xyz[:,2],zn)
    lsasbs[:,0]=116*fy-16
    lsasbs[:,1]=500*(fx-fy)
    lsasbs[:,2]=200*(fy-fz)
    return lsasbs

def lsasbs2xyz(lsasbs):
    xyz_out=np.zeros((lsasbs.shape[0],lsasbs.shape[1]),dtype=np.float64)
    xn=0.9505;yn=1.000;zn=1.089
    fy=(lsasbs[:,0]+16)/116
    fx=lsasbs[:,1]/500+fy
    fz=-lsasbs[:,2]/200+fy
    xyz_out[:,0]=xn*lsasbs_invf(fx)
    xyz_out[:,1]=yn*lsasbs_invf(fy)
    xyz_out[:,2]=zn*lsasbs_invf(fz)
    return xyz_out

def tone_map_inc(x,MIN,MAX):
    y=(x-MIN)/(MAX-MIN)
    y[x<MIN]=0
    y[x>MAX]=1
    return y

rgb_inp=cv2.imread('image.jpg')
rgb=cv2.cvtColor(rgb_inp,cv2.COLOR_BGR2RGB)
rgb=rgb/255
cx,cy,cc=rgb.shape
rgb=rgb.reshape((cx*cy,3),order="F")
rgb=inv_rgb_gamma(rgb)
xyz=rgb2xyz(rgb)
lsasbs=xyz2lsasbs(xyz)
lsasbs[:,0]=100*tone_map_inc(lsasbs[:,0]/100,0.7,0.9)
xyz_out=lsasbs2xyz(lsasbs)
rgb_out=xyz2rgb(xyz_out)
rgb_out=rgb_gamma(rgb_out)
rgb_out=255*rgb_out
rgb_out[rgb_out>255]=255
rgb_out[rgb_out<0]=0
rgb_out=rgb_out.astype(np.uint8)
rgb_out=rgb_out.reshape((cx,cy,cc),order="F")
rgb_out=cv2.cvtColor(rgb_out,cv2.COLOR_RGB2BGR)
cv2.imshow('inp',rgb_inp)
cv2.imshow('out',rgb_out)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

(a)　(b)

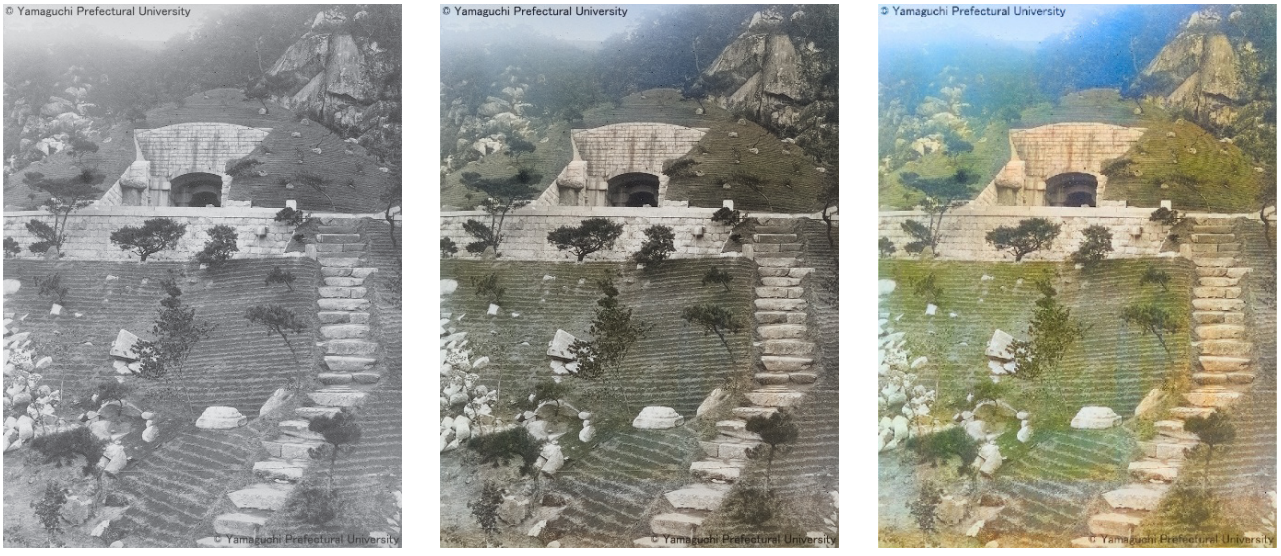
図1. 結果画像1。(a)原画像、(b)明度にコントラスト強調を適用した画像。防長尚武館所蔵。

(a)　(b)　(c)

図2. 結果画像2。(a) 原画像、(b) Zhangらの方法で着色した画像、(c) (b)にカラー画像強調を適用した画像。山口県立大学図書館所蔵。

(a) (b) (c)

図3. 結果画像3。(a) 原画像、(b) Zhangらの方法で着色した画像、(c) (b)にカラー画像強調を適用した画像。山口県立大学図書館所蔵。



(a) (b) (c)

図4. 結果画像4。(a) 原画像、(b) Zhangらの方法で着色した画像、(c) (b)にカラー画像強調を適用した画像。山口県立大学図書館所蔵。