情報理工学特別研究報告書

題目

二次元キャラクター画像の識別に関する研究

学生証番号　954327

氏名　森川　真伍

提出日　令和 5 年 1 月 27 日

指導教員　蚊野　浩

京都産業大学
情報理工学部

要約

アニメやゲーム,漫画などの二次元コンテンツはクールジャパン戦略の重要な一分野になっており，様々な人が触れている.その分野の活性化を期待できる技術である二次元のキャラクター画像に対する認識技術は,一般的な顔画像を用いた個人識別ほどに進んでいない.そのため本研究では二次元のキャラクター画像を用いた顔検出と,画像間の類似度算出を行った.

本論文において,顔検出ではHaar-like特徴とLBP特徴の二つの特徴量を比較し,どちらの検出精度が良いかを調べた.類似度算出では特徴点マッチングと色ヒストグラムの結果を最小二乗法を用いて組み合わせ,人の結果と比較することによって人の感性に近い検出結果を算出できるようにした.特徴点マッチングには AKAZE 特徴と ORB 特徴とBRISK 特徴の三つの特徴と,総当たりマッチングと k 近傍マッチングの二つのマッチング手法があり,それぞれを比較することで,どの手法が人の感性に近い結果を算出できるかを調べた.

作成したプログラムで顔検出を行なった結果,顔以外を顔と認識するフォールスポジティブは LBP 特徴の方が多く,顔を顔以外と認識するフォールスネガティブはHaar-like 特徴のほうが多いが,顔認識精度は LBP 特徴のほうが良いという結果が得られた.次に，キャラクター画像の類似度の算出を,作成したプログラムで行なった.用いるマッチング手法は，k 近傍マッチングの場合画像の順番を変えることによって得られる値が変わってしまう.そのため,総当たりマッチングが良いという結論を得た.また用いる特徴は,人の主観との誤差を何パターンかで算出した結果,ORB特徴が良いという結果を得た.ただし,最小二乗法の妥当性が高いのはBRISK特徴である.そのため,ORB特徴とBRISK特徴のどちらがより優れているかの結論を得ることができなかった.

目次

1 章 序論 1

2 章　顔画像の検出と類似度の計算方法 3

2.1　顔検出の手法 3

2.2　画像間の類似度の算出手法と選択 5

2.3　特徴点マッチングについて 6

2.4　色ヒストグラムについて 8

3 章　二次元キャラクター画像の顔検出と類似度の算出 9

3.1　二次元キャラクター画像の顔検出 9

3.2　二次元キャラクター画像の類似度 10

4 章　二次元キャラクター画像の顔検出と類似度算出の結果と考察 13

4.1　二次元キャラクター画像の顔検出の結果 13

4.2　二次元キャラクター画像の類似度の算出結果. 17

5 章　結論 24

参考文献 25

謝辞 25

付録 開発したプログラムの説明 26

1章 序論

アニメやゲーム，漫画などは二次元コンテンツとよばれており,現在ではクールジャパン戦略の重要な一分野になっている.本論文では，これらの二次元コンテンツにおける登場人物であるキャラクターの画像を識別する技術を研究した.キャラクター画像は,絵師と呼ばれるプロフェッショナルな作画者によるオリジナルな画像であり,経済的な価値がある画像である.

二次元コンテンツの業界では，日々様々なキャラクターが生み出されている.その中には自分の好みの見た目をしているキャラクターが存在する.しかし膨大な数のキャラクターの中から探すことは困難である.そこで人それぞれの好みのキャラクターを，プログラムを用いて探し出す.そうすることで,二次元コンテンツ好きの人々の助けになり,二次元コンテンツ分野がさらに盛り上がると素晴らしいと考えた.

一方，この分野ならではの問題が存在する.例えば他の作品を盗作することであるパクリや模写の一種であるトレースなどである.トレースは著作権侵害に当たらない限りは違法ではない行為である.しかし,構図の一部だけをトレースしたり,左右を反転させたりするなどバレないように悪質な行為が行われている.また近年ネット界隈を騒がせている AI 生成画像の問題もある.これは存在自体が問題なわけではないが,自分で描画した画像と嘘をついて AI 生成画像を公表,販売するなどの悪質行為も存在する.そのため,まともに描いている人が被害を被っている.これらは人の目で判断できないように巧妙に隠されている場合も多々存在する.このような行為を,プログラムを用いて検出できる可能性がある.

ここで，関連する技術について簡単に説明する．画像認識はカメラで撮影した画像などを処理することで，対象物の名称・性質・クラスを識別する技術のことである．印刷文字や手書き文字の認識、顔画像の認識など，対象を限定すれば実用化されている．一方，人間が視覚を用いて行うような広い意味での画像認識は非常に難しい問題である．しかし，最近では深層ニューラルネットワークを用いることで，通常の写真画像で捉えた対象物の名前を認識することが，ある程度可能になっている．

画像認識の中でも,顔画像を用いた個人の識別技術は広く実用化されている.これは,個人の顔画像から，識別するための特徴データとして目・鼻・口などの特徴的な部位を抽出し，個人の特徴データと照合することで識別を行うものである．また，iPhone の FaceID のように三次元顔画像を用いて識別する技術もある．但し，顔画像を用いた識別技術は秘匿性が高いため，実用化されている技術の詳細は不明である．

本研究の内容は，顔画像を用いた個人の識別技術に近いものである．しかし，二次元のキャラクター画像に対する認識技術は,顔画像を用いた個人識別ほどには研究や実用

化が進んでいない.そのためこれを研究することに意義があると考えた．以下，本論文では 2 章で顔検出の手法と画像間の類似度の算出方法について説明する．3 章では今回の研究で作成した顔検出と画像間の類似度算出のプログラムにおいて,どのようなことができ,何を出力するのかについて説明する.4 章では 3 章で説明したプログラムの実行結果と考察について説明する.

2 章　顔画像の検出と類似度の計算方法

2 章では, 二次元キャラクター画像から顔領域を検出する処理と, 2 つの顔画像の類似度を計算する方法について説明する. これらは基本的に, 一般の顔画像を検出, 認識する方法と同じである.

2.1　顔検出の手法

顔検出とは, 一部に顔が写っている画像から顔の領域を抽出する処理である. この処理を行うための代表的なアルゴリズムにカスケード型識別器がある. カスケード型識別器の内部で行われる，顔検出の方法の概要を図 2.1 に示す.

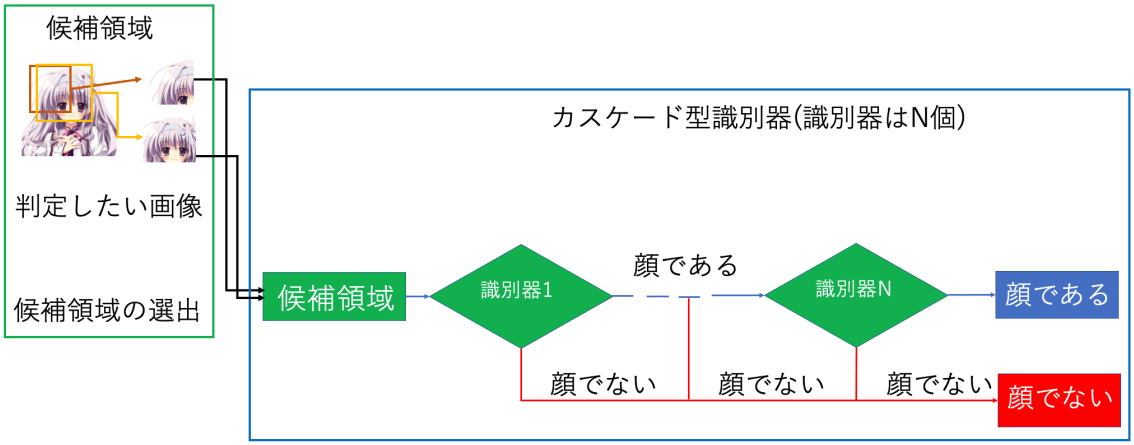


図 2.1 顔検出方法の概要

まず判定したい画像から顔の候補領域となる一部分を切り取り, その部分の顔判定をカスケード型識別器で行う. 候補領域は判定したい画像の中からさまざまな場所と大きさの部分を選出する. カスケード型識別器のなかのそれぞれの識別器は, 比較的単純なアルゴリズムで高速に顔であるかどうかの判定を行う. これらの識別能力は低いので弱識別器と呼ばれる. この図では, まず識別器 1 でその部分が顔かどうかを判断する. ここで顔であると判断されれば識別器 2 に行く. 顔でないと判断されればその候補領域は顔でないと判断し，次の候補領域の処理を行う. このような処理を繰り返して最終判定器で顔であると判定されればその候補領域は顔であると判定する. このように, たくさんの弱識別器をカスケード接続することで強識別器を作成し識別精度を高くしている. また一度でも顔ではないと判断されると, その時点で判定を終了するため高速な処理を実現できる.

次に識別器の内部で行われる処理について説明する. 顔の判定するための学習サンプルとして, 顔画像であるポジティブサンプルと, 顔でない画像であるネガティブサンプルをたくさん用意する. これらのサンプルを学習した識別器は, 入力した画像から特徴量を抽出し, それに基づく評価値が事前に決めた一定の値を超えている場合に，顔であ

る判定する.

特徴量にはいくつかの種類が存在する.今回のプログラムは OpenCV を用いているため, Haar-Like 特徴量と LBP(Local Binary Pattern)特徴量,HOG(Histogram of Oriented Gradients)特徴量の 3 種類から選択できる.このなかで HOG 特徴量はうまく動作しなかったため,Haar-Like 特徴量と LBP 特徴量の二つを比べる.OpenCV はインテルが開発した画像処理ライブラリである.

Haar-Like 特徴は画像の一部分を切り出し,そこの局所的な明暗差を算出することで特徴を捉える方法である.画像内の局所的な明暗差のパターンには,ありとあらゆる組み合わせが無数にあり,基本的にはその全てが Haar-Like 特徴の一つである.



図 2.2 グレースケール画像の例



図 2.3 Haar-Like 特徴の検出例の一部

図 2.2 の画像を Haar-Like 特徴で検出した例の一部を表したものが図 2.3 となっている.図 2.3 のような特徴検出を顔全体で行い,たくさんの明暗の領域の集合から,顔を有効に検出できるパターンを学習する.Haar-Like 特徴量の計算は,まず特徴抽出に用いるパターンを多数用意する.このパターンの例を以下に示す.

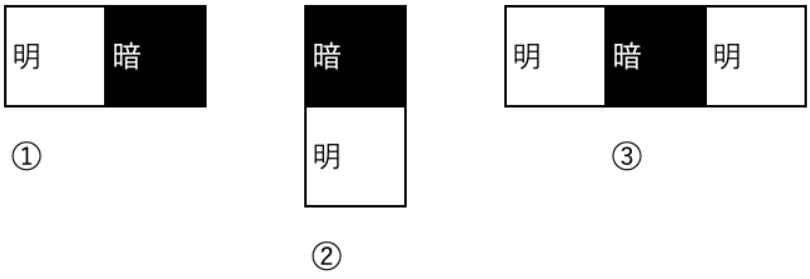


図 2.4 特徴検索に用いるパターンの例

図 2.4 のように上下左右,または斜めや,③のように明暗を挟んだ組み合わせパターンを用意する.図 2.3 の右目は①,左目は②,髪と背景のところは②の明と暗を反転させたものとなっており,これら部分では例示した Haar-Like 特徴で特徴量が大きくなることを示している.次に探索窓(入力画像の任意の範囲)を配置して,その中から矩形領域

(探索窓内のある Haar-Like 特徴を適用する場所)を配置する.そして,任意の特徴探索に用いるパターンを矩形領域に当てはめる.次にパターンの暗を A,明を B として,特徴量 H を H=S_1(領域 A の平均輝度)-S_2(領域 B の平均輝度)として求める.その後矩形領域を変更してまた特徴量を求め,求め終わったら探索窓を変更して同じようなことを行う.探索窓と矩形領域のパターンは膨大な数存在するため,ブースティングと呼ばれる方法で重要度の高い特徴量のみを用いる.重要度の高い特徴量とは,目鼻口や顔の輪郭などを抽出するものになる[1].

LBP 特徴量は中央画素と周辺画素の画素値の関係性を元に算出する特徴量のことである.図 2.5 を用いて説明する.

80	67	120
200	100	150
48	90	110

0	0	1
1		1
0	0	1

図 2.5 LBP 特徴量の計算例

まず,周囲 8 画素の画素値が中央画素の画素値より小さければ 1,大きければ 0 とする.これを図 2.5 の左の 3×3 領域に適用すると右の 3×3 領域になる.そして黄色の部分から時計回りに数字を並べると二進数で $00111001_{(2)}$ となり 10 進数では 57 となる.この計算を元画像の全ての画素で行い,求めた 10 進数の値でグレースケール画像を作成する.この求めた 10 進数の値が LBP 特徴量である.この値を扱いやすくするためにグレースケール画像の画素値を元にヒストグラムを作成する.このヒストグラムを比較することで顔であるかどうかを判定する[2].

2.2 画像間の類似度の算出手法と選択

画像間の類似度を算出する方法として,色ヒストグラムや特徴点マッチング,IMAGEHASH,DEEP LEARNING,COUNT_NONZERO()関数などがある.このなかで今回の研究では,特徴点マッチングと色ヒストグラムを用いた.その理由を説明する.

IMAGEHASH とは画像の情報をハッシュ化(データを不規則な文字列に変換すること)する際に同じような画像だと同じような値に,似た画像だと似たような値を出力するものである.これは大まかな形で比較したり,大きさの違う同じ画像を調べたりなどの大まかな作業を行うのに適しているものである.顔を比べるといった細かな作業には向いていないため用いなかった.DEEP LEARNING とは深層学習のことであり,特徴量を人ではな

くコンピュータが探して学習を行うものである.これは今回自分がやろうとしている機械学習よりも精度が高くなるが,高度な知識が必要となり,この技術を習得するための時間が必要になる.そのため,今の自分では難しいと判断したため今回は用いなかった.COUNT_NONZERO関数()とは画像同士の同じ位置の画素値を比べ,全画素を調べた後に一致率を判定して類似度を判定する方法である.これは行なってみると精度が怪しかったため用いなかった.色ヒストグラムとは画像の RGB 値を用いる方法である.そのため肌の色や髪の毛の色などで類似度を判断できる.しかし,色だけの判断となり,色以外の目の感じやキャラクターの形状的な雰囲気などで判定することができないと考えた.この欠点を解消できる類似度算出方法として特徴点マッチングを選択した.

2.3　特徴点マッチングについて

一般的に画像における特徴とは,それぞれの画像を特徴づける性質のことである.画素値や画像のヒストグラムも特徴と言えなくもない.しかし,本論文では画像中で顕著な変化が生じている局所的な領域を対象とする.このような特徴を特徴点という.本研究で特徴というときには特徴点のことを意味する.

特徴点マッチングとは２画像の間で特徴点を対応させることである.本研究においては,対応させた特徴点の間で類似度を計算する.画像から特徴の検出を行う特徴検出器を利用するが,特徴検出器にはいくつかの種類が存在する.OpenCV ではさまざまな種類を用いることができ,今回はAKAZE特徴とORB特徴とBRISK特徴を用いる.特徴点を検出する場所は角(コーナー)や輝度の勾配が変化する箇所などである.

よく知られた特徴点検出アルゴリズムにSIFT特徴がある.これは,ガウシアンフィルタの差分を用いて特徴点を検出し,さらに,その近傍の勾配ヒストグラムを使って,スケールと回転に不変性を持たせたものである.SIFT 特徴の精度を改善したものに KAZE 特徴があり,AKAZE特徴はKAZE特徴を高速化したものである[3].

ORB 特徴は,決定木を用いてコーナー特徴を効率的に検出する FAST 検出器と,二点の画素値の比較によってバイナリコードを生成し,特徴記述子とする BRIEF 特徴の考え方に,スケールと回転の普遍性を導入したものである.二点の画素値の比較は,パッチ(ある注目点に対して比較を行う小領域)内であり得るペアを全て列挙し,その中からビットの分散が大きく,ペアの相関が低くなるようなペアを選出する[4].

BRISK特徴はBRIEF特徴の考え方に,スケールと回転の普遍性を導入したものである.特徴量記述にはパッチに配置された 4 つの同心円上において,等間隔にサンプリングされた60箇所の画素値を用いる[4].それを図2.6に示す.

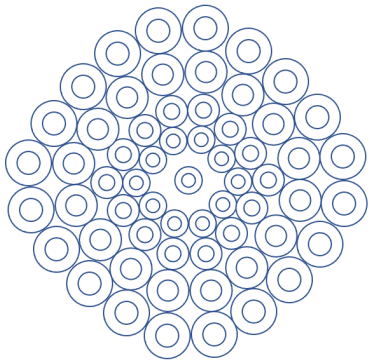


図 2.6 BRISK 特徴で輝度比較をする 60 箇所の位置

特徴検出器を用いて画像から検出した局所的な特徴を特徴点と呼び,特徴点周りの領域から取り出した特徴量を用いて特徴の性質を記述したものを特徴記述子と呼ぶ.画像から特徴点と特徴記述子の抽出を行い,類似度が高い特徴点同士をマッチングすることを特徴点マッチングと言う.特徴点マッチングには総当たりマッチングと近似近傍探索手法,k 近傍マッチングが存在する.近似近傍探索手法はさまざまなパラメータが存在しており,適当なパラメータは画像ごとに異なっている.そのため,さまざまな画像を用いる必要がある今回の研究では適当ではないと考えた.そのため本研究では総当たりマッチングと k 近傍マッチングを利用した.総当たりマッチングと k 近傍マッチングを,図 2.7 を用いて説明する.

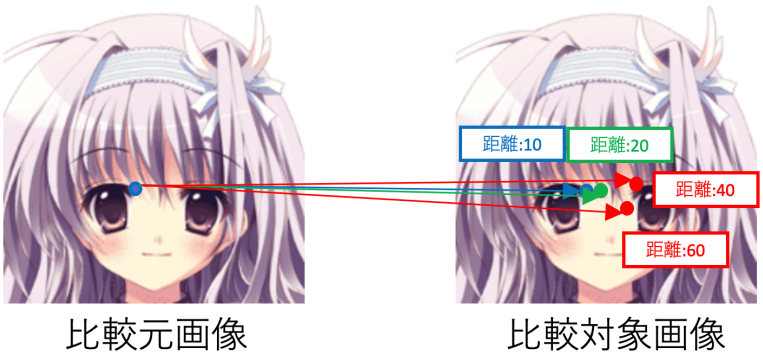


図 2.7 特徴点マッチングのイメージを説明するための図

(実際の結果ではない)

図 2.7 の距離は物理的な距離ではなく,特徴点同士の類似度に対応する距離である.この距離を算出する方法として,AKAZE 特徴と ORB 特徴と BRISK 特徴では,ハミング距離を用いている.ハミング距離とは等しい文字数を持つ二つの文字列の間で,対応した位置の文字が異なっている個数を値とした距離である.例えば“abc”と“abd”だと a と b は同じで c,d が異なっているため,ハミング距離は 1 となる.総当たりマッチングは元画像の一つの特徴記述子に対して,比較画像の全特徴記述子との距離を求める.その距離が最も小さい点を対応する特徴点とする方法であり,図 2.6 の場合だと青いところが

マッチングした特徴点となる.k近傍マッチングは全ての対応について距離を求めた後,距離が近い上位 k 個の平均距離を特徴点マッチングの距離とする方法である.図 2.6 で k=2 の場合,青と緑のところがマッチングした対応点となる.この方法は比較元画像の全特徴点に k 個の特徴点をマッチングさせる.そのため,比較元画像と比較対象画像を逆にすると,マッチングした特徴点の個数が変わってしまう.同じ画像同士なのに,入れ替えるとハミング距離が異なるのは今回の研究上ふさわしくないと考えた.よって,今回の研究で使用するマッチング手法には総当たりマッチングを用いた.

2.4　色ヒストグラムについて

画像の色ヒストグラムは,画素値のカラー情報をヒストグラム化したものである.　本研究では,RGB の成分ごとに求めた３つのヒストグラムを画像の色ヒストグラムとして用いる.ヒストグラム間の類似度を求めるために EMD(EARTH MOVER’S DISTANCE)関数を用いた.EMD 関数は,ヒストグラムが二つあったとき,一方のヒストグラムを他方のヒストグラムに等しくするために必要な数値の移動量の総和を求める関数である.EMD 関数の計算例を,図 2.8 を使って説明する.

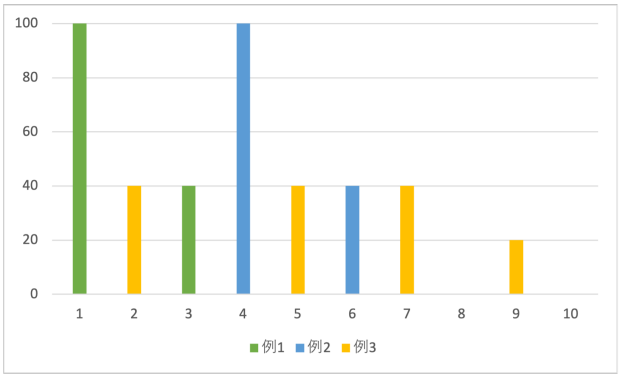


図 2.8 EMD 関数の計算例のためのヒストグラム

図 2.8 の例 1 と例 2 を比較すると,1→4 に 100 移動する 3×100 と,3→6 に 40 移動する 3×40 を合計して 420 となる.例 1 と例 3 を比較すると,1→2 に 40 移動する 1×40 と 1→5 に移動する 4×40,1→7 に 20 移動する 6×20,3→7 に 20 移動する 4×20,3→9 に 20 移動する 6×20 の全てを合計して 520 となる.そのため例 1 は例 2 の方が例 3 よりヒストグラムが似ていると判断できる.このようにして求めた値を色ヒストグラムの類似度として用いる.

3 章　二次元キャラクター画像の顔検出と類似度の算出

3.1　二次元キャラクター画像の顔検出

二次元キャラクターは基本的に,造形から三次元の顔と異なっていることが多い.



図 3.1 二次元キャラクターの顔画像の例



図 3.2 通常の顔画像の例

図 3.1 と図 3.2 の顔を見比べてみると,例えば二次元キャラクターは鼻が簡略化されていたり,口が簡略化(唇を省略など)されていたり,目が大きく描かれていたりすることがわかる.そのため三次元の顔と同じプログラムで顔検出すると精度が著しく落ちると考えた.そのため二次元キャラクター専用の顔検出プログラムである facescan.py を作成した.

facescan.py のプログラム中に,CascadeClassifier と呼ばれるカスケード分類器の情報が記載された xml 形式の設定ファイルを読みこむためのクラスが存在する.xml ファイルの情報によって,検出する対象を変更できる.OpenCV のカスケード分類器をカスタマイズする場合,多数の教師画像を用意し,それを学習させることで xml 形式の設定ファイルを作る.教師画像として,ポジティブサンプル 400 枚とネガティブサンプル 200 枚で作成したものと,ポジティブサンプル 1000 枚とネガティブサンプル 500 枚で作成したものの二つを用意した.これらの画像の画素数は 200×200 画素に固定した.ポジティブサンプルの画像として自分で選択したさまざまなキャラクターの正面顔画像を用いていた.ネガティブ画像は,顔以外の画像であり,例のいくつかを図 3.3 に示す.

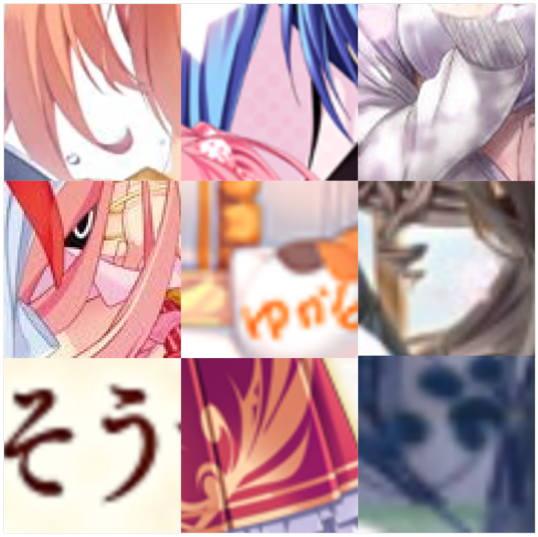


図 3.3 ネガティブサンプルの例

これらは,キャラクターの顔画像を検出するプログラムが誤検出したところや, 自分で判断した適当な画像を用いている.誤検出した画像は遠くから見ると何となく顔のように見える画像や,文字の画像が多い.

このようにして作成した facescan.py を用いて,二次元キャラクターが含まれた画像について顔検出を行い,どの特徴量が適しているのか,それぞれの利点と欠点などについて調べた.

3.2 二次元キャラクター画像の類似度

二次元キャラクター画像同士の類似度を算出するプログラムである goal.py を作成した.このプログラムは顔画像がたくさん含まれているファイルから一つ画像を選び,その画像とそれ以外の画像の類似度を特徴点マッチングと色ヒストグラムの二つの方法で算出する.

特徴点マッチングには AKAZE 特徴と ORB 特徴と BRISK 特徴の三つを用いる.その三つの結果を比較し,どの特徴が人との感性に近づくかを調べた.また,三つの結果を組み合わせて最小二乗法を用いることでより良い結果を得ることができるのかなどについて調べた.その後,総当たりマッチングを用いて特徴量の距離を,ハミング距離を用いて算出する.そして,それを画像の類似度として算出した.

色ヒストグラムによる比較では,画像の BGR 値を求め,3 色それぞれのヒストグラムを作成する.背景の色合いに左右されないように,すべての画像の背景を白抜きにし,ヒストグラムの作成において背景の白色を排除したいと考えた.それがうまく動作しなかったため BGR それぞれの 255 を排除した.このため(B,G,R)=(255,200,50)といった画素値のヒストグラムの計算に誤差が発生しているが,その影響は軽微である.このように求

めたヒストグラムと EMD 関数を用いてヒストグラムの距離を算出し, それをもとに色に関する画像の類似度を算出した.

今回の実験ではあらかじめ準備した多数の画像から 51 枚を選択した. そのうちの 1 枚を, 類似度を比較する参照画像とする. 残りの 50 枚との類似度を 8 人に 1〜10 段階で評価してもらい, その平均点を求める. 51 枚の画像を以下に示す.



図 3.4 類似度比較対象の 1 枚



図 3.5 図 3.4 と比較する 50 枚の画像

図 3.4 の画像のファイル名を 0.png とした. 図 3.5 の画像は, 左上から右に 1.png, 2.png …としており, 赤い四角で囲った画像は同じ原画家が書いたキャラクターである.

特徴点マッチングによる距離と, 色ヒストグラムによる距離の値が取る範囲は異なっている. また, 最大値と最小値の範囲も不明瞭であるため, 距離の値を次のように正規化した. まず, 平均値が 0 になるように値をシフトし, 標準偏差の 3 倍で割る. これにより, データが正規分布すれば, 約 99.7 が-1〜1 の範囲に収まる. その範囲を 0〜1 の範囲に変換する. 特徴点マッチングの距離と色ヒストグラムの距離は, 両方とも値が小さいほど画像が似ていると判断できるため, 値を反転し, 100 倍することで類似度の表現とした.

人の主観評価と, 特徴点マッチングと色ヒストグラムの二つの値を用いて算出した値を, 最小二乗法を用いて近づける. 今回の式は

人の主観評価 = a×特徴点マッチングの値 + b×色ヒストグラムの値 + c …(1)

となる. 誤差の二乗和が最小になるように a, b, c を決めるのが最小二乗法である. これによってプログラムで算出した類似度よりも人の主観に近づいた結果を出すことができると考えた.

この後図 3.5 とは異なった 50 枚の画像と図 3.4 の画像の類似度を 8 人にそれぞれ 1～10 段階で評価してもらう. それを図 3.6 に示す. 赤い四角で囲った画像は同じ原画家が書いたキャラクターである.



図 3.6 図 3.4 と新たに比較する 50 枚の画像

新たに求めた人の主観評価と, 図 3.4 と図 3.5 で最小二乗法を用いて求めた a, b, c を用いた(1)の式を比べて, どの程度の乖離があるのかを調べる. これによって, 今回のプログラムで求めた値がどのくらい人の主観と合っているのかを評価する. またこの結果をもとに二次元キャラクター画像に対して適している特徴量は何かを考察する.

4 章　二次元キャラクター画像の顔検出と類似度算出の結果と考察

4.1　二次元キャラクター画像の顔検出の結果

facescan.py を用いて，二次元キャラクターが含まれる図 4.1，図 4.2 の画像から顔検出を行った．



図 4.1 顔検出に用いる画像その 1



図 4.2 顔検出に用いる画像その 2

図 4.1 の画像に写っている顔は xml ファイルを作成するために用いた顔に含まれていない．図 4.2 の画像に写っている顔は xml ファイルを作成するために用いた顔に含まれている．図 4.1 と図 4.2 には，正面を向いている顔や少し横を向いている顔，目を瞑っている顔，傾いている顔など様々なパターンが含まれている．

まず通常の顔を検出する xml ファイルを用いて顔検出を行った[5]．その結果を図 4.3，図 4.4 に示す．



図 4.3 図 4.1 の画像に通常の顔検出 xml ファイルを用いた場合



図 4.4 図 4.2 の画像に通常の顔検出 xml ファイルを用いた場合

このように通常の顔検出 xml ファイルではキャラクターの顔を全く検出できなかった.

図 4.1 と図 4.2 に対して,Haar-like 特徴と LBP 特徴のそれぞれで作成した xml ファイルを用いて顔検出を行った.ポジティブサンプル 400 枚とネガティブサンプル 200 枚を学習させた xml での顔認識結果を図 4.5 から図 4.8 に示す.ポジティブサンプル 1000 枚とネガティブサンプル 500 枚を学習させた xml での顔認識結果を図 4.9 から図 4.12 に示す.



図 4.5 Haar-like 特徴での結果 1-1



図 4.6 LBP 特徴での結果 1-1



図 4.7 Haar-like 特徴での結果 1-2



図 4.8 LBP 特徴での結果 1-2



図 4.9 Haar-like 特徴での結果 2-1



図 4.10 LBP 特徴での結果 2-1



図 4.11 Haar-like 特徴での結果 2-2



図 4.12 LBP 特徴での結果 2-2

この 8 つの図について, 適合率や再現率, 正解率, F 値を求めたものを表 1 に示す.

表 4.1 図 4.4～4.12 の適合率, 再現率, 正解率, F 値

	適合率	再現率	正解率	F 値
図 4-5	0.71	0.5	0.42	0.59
図 4-6	0.56	0.9	0.53	0.69
図 4-7	0.71	0.56	0.45	0.62
図 4-8	0.47	0.89	0.44	0.62
図 4-9	1	0.5	0.5	0.67
図 4-10	0.75	0.6	0.5	0.67
図 4-11	1	0.56	0.56	0.71
図 4-12	1	0.78	0.78	0.88

この表 4.1 を見ると, 適合率が高いのは Haar-like 特徴で, どちらの特徴も枚数が増えると結果が上昇している. 再現率が高いのは LBP 特徴で, どちらの特徴も枚数が増えると結果は下降気味である. 正解率と F 値は LBP 特徴でたくさんの画像を用いたものが一番高くなっている. このことから, 顔以外を顔と認識するフォールスポジティブは LBP 特徴の方が多く, 顔を顔以外と認識するフォールスネガティブは Haar-like 特徴のほうが多いことがわかる. また LBP 特徴は画像が少ないときでも, フォールスポジティブこそたくさんあるが顔認識精度はかなりよく, 画像が増えるとフォールスポジティブの数も減る. 一方 Haar-like 特徴は画像が少なくてもフォールスポジティブは少ないが, 画像が増えたとしても顔認識精度はあまり良くならない. フォールスネガティブが発生しているのは, 顔が正面を向いていなかったり, 顔を傾けている場合が多い.

また, 学習に要した時間は, Haar-like 特徴では 7 分と 56 分で, LBP 特徴は 20 秒, 18 分 50 秒となっており, LBP 特徴のほうが短い.

このようなことから, 二次元キャラクター画像では LBP 特徴の方が適していると考えた. また画像を増やした結果, 顔と認識していたものを顔と認識できなくなっているの

は,恐らく増やした画像に不適切な画像が含まれていた為だと考えられる.そのような不適切な画像に対する耐性はHaar-like特徴の方に分があると考えられる.

4.2　二次元キャラクター画像の類似度の算出結果.

開発したgoal.pyを用いて画像同士の類似度を算出した.まず特徴点の検出を,図3.4と図3.5の43.pngの画像で行った.その例を図4.13～図4.15に示す.



図4.13　0.pngと43.pngのAKAZE特徴



図4.14　0.pngと43.pngのORB特徴



図4.15　0.pngと43.pngのBRISK特徴

この6つの画像を見比べたとき,特徴点の検出量はBRISK特徴>ORB特徴>AKAZE特徴となっている.またAKAZE特徴とORB特徴の場合はほとんど目と髪の先端しか検出していないが,BRISK特徴の場合は髪の先端以外も特徴と検出している.これらの特徴は,人間が特徴として判断する目口髪などの部位とは必ずしも対応していないことがわかる.

次に図3.4と図3.5の43.pngの特徴点をマッチングさせた結果を以下に示す.

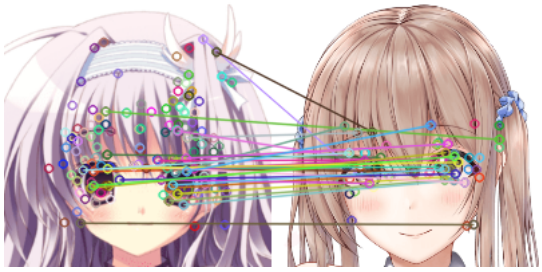


図 4.16 AKAZE 特徴を用いたマッチング



図 4.17 ORB 特徴を用いたマッチング

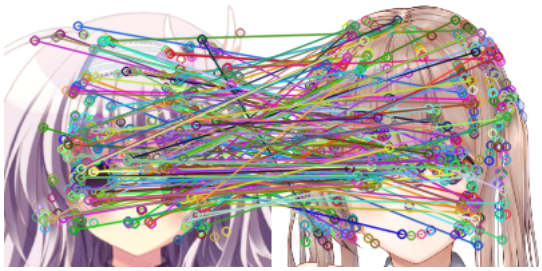


図 4.18 BRISK 特徴を用いたマッチング

またそれぞれの画像の特徴点抽出量と特徴点同士がマッチングした個数,ハミング距離で算出した距離を表 4.2 に示す.

表 4.2 各方法の特徴点とマッチングの個数とハミング距離

	0.png の特徴点の個数	43.png の特徴点の個数	マッチングした個数	ハミング距離
図 4.16	133	63	29	116.66
図 4.17	391	377	93	59.35
図 4.18	453	645	157	129.15

表 4.2 を見ると,特徴点の個数とハミング距離に関連性がないことがわかる.また特徴検出器によって出力する値に差異があることがわかる.これはハミング距離の計算に特徴量記述子の値を用いているためである.AKAZE 特徴の場合は 61 次元,ORB 特徴の場合は 32 次元,BRISK 特徴の場合は 64 次元の特徴量記述子が複数格納された配列がある.その配列の値を 10 進数から 2 進数に変更し,その 2 進数でハミング距離を算出する[6].このため同じ画像同士でも,特徴検出器ごとのハミング距離に差異がある.また特徴点

の検出が多いほど総当たりマッチングでマッチングした特徴点が多くなっていることがわかる. ただし, 特徴検出器の優劣はここの数字だけでは判断できない.

次にそれぞれの特徴で算出した上位 5 件と下位 5 件の画像と値を以下に示す. 図の中の値はハミング距離ではなく, 標準化処理などを行った後の数字である.

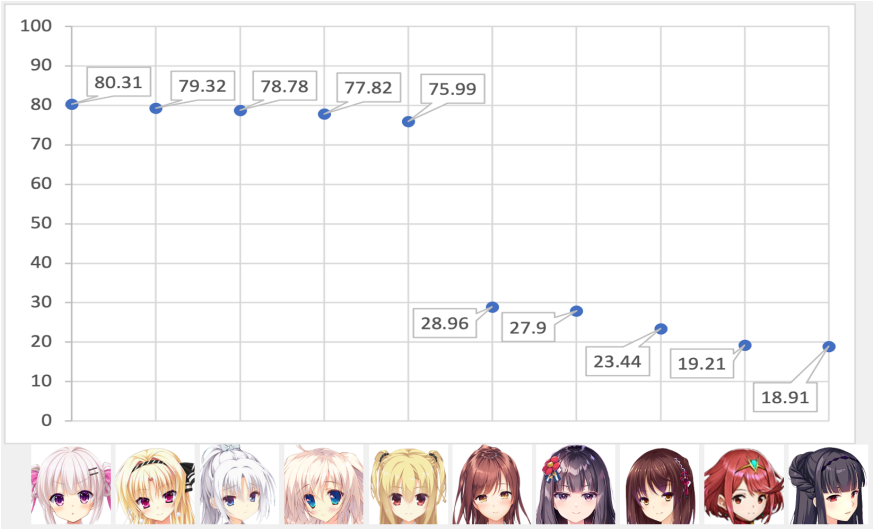


図 4.19 AKAZE 特徴を用いた特徴点マッチングの上位, 下位 5 件の結果

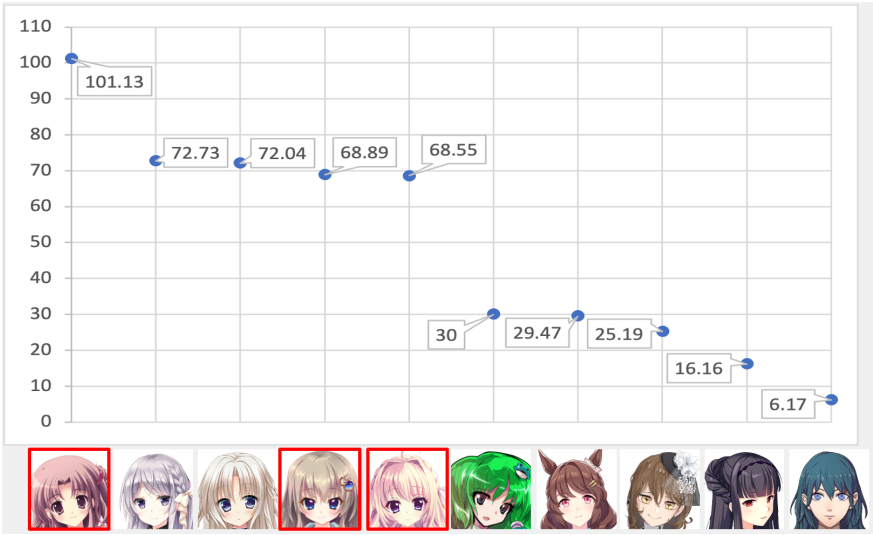


図 4.20 ORB 特徴を用いた特徴点マッチングの上位, 下位 5 件の結果

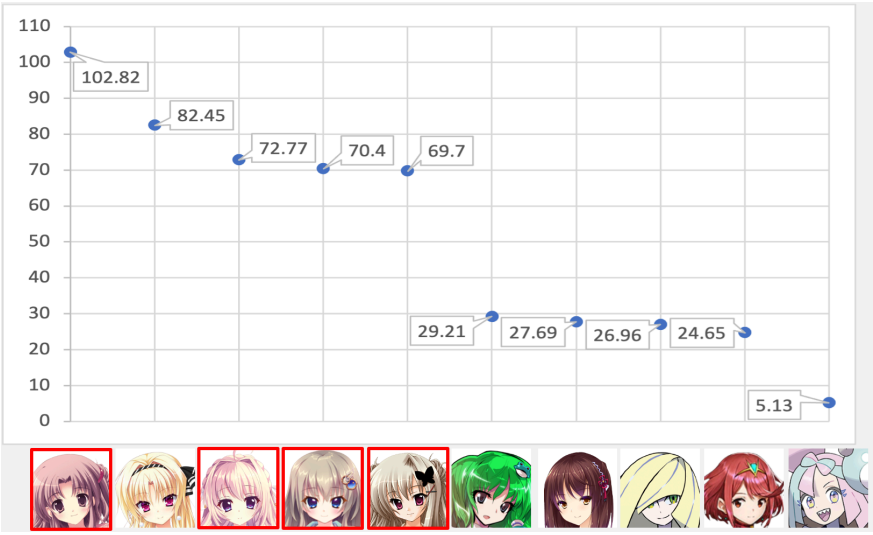


図 4.21 BRISK 特徴を用いた特徴点マッチングの上位, 下位 5 件の結果

3 つの結果を比べると, AKAZE 特徴が一番値の範囲が狭く, ORB 特徴と BRISK 特徴は同じ程度である. また ORB 特徴と BRISK 特徴の上位 5 件は似ているが, 下位 5 件はそれぞれの特徴で 1 つの画像しか共通していないことがわかる. ORB 特徴と BRISK 特徴の一番値が大きい画像は図 3.4 の母親キャラクターなので, 特徴量の値が大きくなる妥当性は高い. また赤い四角で囲った画像は, 原画家が同じであるため, 類似度の値が大きくなる妥当性が高い. ORB 特徴には 3 枚, BRISK 特徴には 4 枚あるが, AKAZE 特徴には 1 枚も存在しない。そのため, AKAZE 特徴は精度が ORB 特徴と BRISK 特徴よりも劣っていると考えた。一方下位 5 件の共通点を見ると, 目が小さく描かれていたり, 顔が正面向きではなかったりしていることが挙げられる.

次に色ヒストグラムを用いた類似度算出を図 3.4 と図 3.5 の 43.png の画像で行った例を説明する.

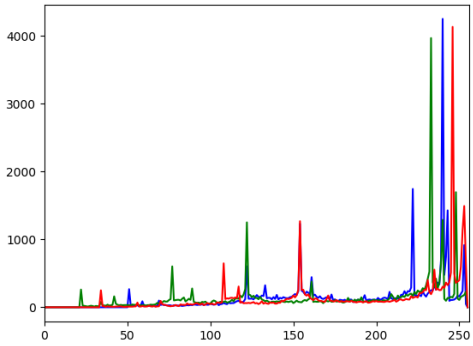


図 4.22　0.png の色ヒストグラム

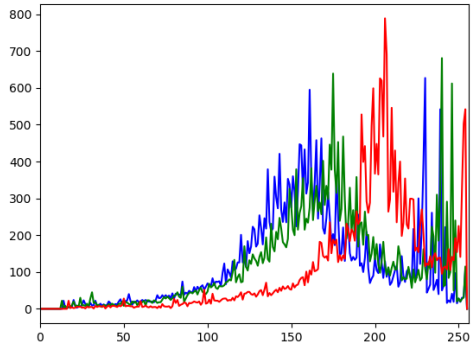


図 4.23　43.png の色ヒストグラム

基本的にキャラクター画像の色ヒストグラムの山は, 肌の色と髪の色の部分に生まれる. 図 4.22 からわかるように, 0.png は肌の色である (g, b, r)=(230, 220, 250) あたりと,

銀色の髪の色である(g, b, r)=(235, 240, 245)あたりに山ができている. 一方図 4.23 からわかるように, 43.png は肌の色である (g, b, r)=(230, 220, 250)あたりと, ベージュ色の髪の色である(g, b, r)=(170～220, 160～200, 200～240)あたりに山ができていることがわかる. この二つの距離は62.29となっている. 色ヒストグラムで算出した上位5件と下位5件の画像と値を以下に示す.

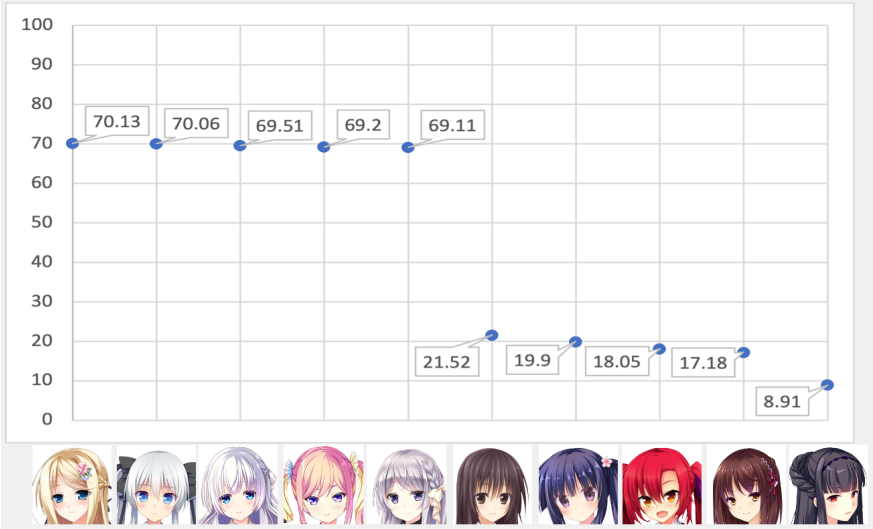


図 4.24 色ヒストグラムの上位, 下位 5 件の結果

図 4.24 の値を見てみると, 上位5件の値はほとんど変わっていない. 図 3.4 の髪の色は銀色であるが, 一番値が大きい画像は金髪となっている. また 4 枚目の画像はピンク色であり, 銀系統の髪ではない. こうなる原因として, 金色は(b, g, r)=(210～240, 230～250, 220)あたりが多く, ピンク色は(b, g, r)=(170～200, 230, 250)あたりが多いため, 銀色の(b, g, r)=(220～240, 230, 250)あたりで似通っているためだと考えられる. 一方下位5件は黒色の(b, g, r)=(100, 100, 100)あたりや, 赤色の(b, g, r)=(50, 50, 250)あたりなど銀色と離れている色となっている.

次に特徴点マッチングと色ヒストグラムを組み合わせて, 最小二乗法を用いる. その最小二乗法で求めた式(1)の a, b, c の値を表 4.3 に示す. また特徴量マッチングや色ヒストグラム, 最小二乗法で求めた値と人の主観評価の、画像一枚あたりの誤差を表 4.4 に示す.

表 4.3 最小二乗法で求めた a, b, c の値

	a	b	c
AKAZE 特徴	0.645	0.269	-0.838
ORB 特徴	0.689	0.384	-8.777
BRISK 特徴	0.521	0.603	-11.352

表 4.4 図 3.4 と図 3.5 を用いた時の検出結果の一枚あたりの誤差

	特徴点マッチング	色ヒストグラム	最小二乗法
AKAZE 特徴	12.85	14.93	11.05
ORB 特徴	11.84	14.93	8.94
BRISK 特徴	16.03	14.93	10.33

この二つの表を見ると,二つの類似度を組み合わせたとき,AKAZE 特徴と ORB 特徴は特徴点マッチングの重みが大きく,BRISK 特徴は色ヒストグラムの重みが大きいことがわかる.画像一枚あたりの誤差は小さいほど推定誤差が小さく,優れていると考えられる.従って表 4.4 より,今回の場合は ORB 特徴>BRISK 特徴>AKAZE 特徴であると言える.a,b,c を求める際の値は,図 3.5 の画像で求めた類似度を用いている.そのため,図 3.5 の画像に最適化した値となっており,図 3.5 以外の画像で用いた場合は結果が変わるかもしれない.今回の研究では色々な画像で用いることができるようにしたい.最小二乗法で求めた式の妥当性を検証するため,図 3.4 と図 3.6 の画像で新たに算出した a,b,c を用いて算出した値と,表 4.3 の値を用いて算出した値の誤差を調べる.その値と,特徴量マッチングや色ヒストグラム,最小二乗法で求めた類似度との、画像一枚あたりの誤差を表 4.5 に示す.

表 4.5 図 3.4 と図 3.6 を用いた時の画像一枚あたりの誤差と比較結果

	特徴点マッチング	色ヒストグラム	最小二乗法	比較結果
AKAZE 特徴	9.33	14.11	8.83	2.36
ORB 特徴	12.15	14.11	8.23	3.08
BRISK 特徴	15.01	14.11	8.16	2.31

表 4.5 より, 比較結果を見比べると,BRISK 特徴>AKAZE 特徴>ORB 特徴となっていることがわかる.そのため最小二乗法で求めた式で,一番妥当性が高いのは BRISK 特徴であるといえる. 図 3.4 と図 3.6 の画像に,最適化した値を用いた場合は,BRISK 特徴>ORB 特徴>AKAZE 特徴となっていることがわかる.一方表 4.4 の値は, ORB 特徴>BRISK 特徴>AKAZE 特徴となっており,順番が一致しない.そのため,表 4.4 と表 4.5 のそれぞれの特徴の平均を取ってみる.すると ORB 特徴>BRISK 特徴>AKAZE 特徴となる.

今回の研究で,最小二乗法の妥当性が一番高いのは BRISK 特徴である.これは,今回の場合,BRISK 特徴を用いた場合のみ,色ヒストグラムの方に比重が向いていたことが関係していると考えられる.そのため,特徴点の値の変動に比較的鈍感となっている.また AKAZE 特徴も BRISK 特徴とほとんど変わらない値となっている.これは図 4.19～図 4.22

でわかるように,AKAZE特徴だけ値がとる範囲が狭い.よって極端な値の差が生まれにくく,誤差が大きくなりにくいことが関係していると考える.また最小二乗法の誤差は一番大きくなっている.そのため,この結果からAKAZE特徴が,二次元キャラクター画像の類似度比較に優れているとは言い難いと考える.

このことから,二次元キャラクター画像の類似度比較で用いる特徴量は,ORB特徴≧BRISK特徴>AKAZE特徴という順番を得ることができた.ただし,ORB特徴とBRISK特徴のどちらが優れているかは今回の研究だけでは結論を得ることができなかった.そのため,今回の2パターンだけではなく,たくさんの画像で研究を行わないといけない.

5章　結論

Haar-like 特徴と LBP 特徴の二つの方法で顔検出を行なったが,フォールスポジティブこそ多いものの,顔検出精度が高い LBP 特徴のほうが二次元キャラクター画像において優れているという結果を得た.

AKAZE 特徴と ORB 特徴と BRISK 特徴の三つの特徴と,総当たりマッチングと k 近傍マッチングの二つのマッチング手法を用いて,画像間の類似度算出を行った.その結果,二次元キャラクター画像において優れているマッチング手法は総当たりマッチングとなり,特徴はORB特徴という結果を得た.ただし最小二乗法の妥当性が高いのはBRISK特徴である.そのため ORB 特徴と BRISK 特徴のどちらがより優れているかの結論を得ることができなかった.

今後の課題として,顔検出は,元々図 3.4 のような頭のてっぺんから顎の下までの顔を集めてポジティブ画像としていたが,全然うまくいかなかったため今回のように頭の中部から首にかけての検出となった.そのため頭のてっぺんから顎の下までの画像を認識できるように改善したい.

特徴点マッチングにおいて, BRISK 特徴以外ほとんど目だけで判断している.そのため,髪や輪郭といった部位でも特徴点を検出できるように改善し,精度を上げていきたい.また,ORB 特徴と BRISK 特徴をうまく組み合わせることで,より良い結果を得ることができる可能性があるため,その実験を行いたい.

色ヒストグラムにおいて,銀髪が白髪よりも金髪の方が近いと判断されることは,違和感があるため改善したい.また背景とした(b,g,r)=(255,255,255)だけをヒストグラムから除外できるようにしたい.また図 4.27 の一番似ている画像の時の標準化前の値は 3.86 と 0 にかなり近くなっているが,標準化した値は 70.13 となっており,100 とはかけ離れており正常とは言いにくくなっている点も改善したい.

参考文献

[1]【Haar-Like 特徴量】原理と計算式・高速化
https://algorithm.joho.info/image-processing/haar-like-feature-value/
2023 年 1 月 19 日確認.

[2] Local Binary Pattern の理論と実装 - Qiita
https://qiita.com/tancoro/items/959ae9c14048c06bea8e
2023 年 1 月 19 日確認.

[3] 特徴点抽出とマッチング - け日記
https://ohke.hateblo.jp/entry/2019/08/03/235500
2023 年 1 月 19 日確認.

[4] 藤吉弘亘, 安部満,「局所勾配特徴抽出技術 -SIFT 以降のアプローチ-」, 精密工学会誌, Vol.77, No.12, 2011, pp.1109-1116.

[5] https://github.com/opencv/opencv/tree/master/data/haarcascades
Haarcascade_frontalface_default.xml
2023 年 1 月 19 日確認.

[6] OpenCV をつかった特徴点マッチングについて少しだけ掘り下げる
2020/09/16
https://qiita.com/komiya　　/items/c024e38959e389442dd0
2023 年 1 月 19 日確認.

謝辞

本論文を作成にあたり, 丁寧な御指導を賜りました蚊野浩教授に感謝いたします.

付録 開発したプログラムの説明

Facescan.py

顔検出を行うプログラム.ある画像を指定すると,その画像から顔を検出し,その部分をファイルに入れる.

Similarity.py

画像間の類似度を検出するプログラム.指定した画像が入ったフォルダを指定し,そのファイルの中に入った画像を一つ指定する.そうするとそのファイル内の画像で,特徴点マッチングでの類似度,色ヒストグラムでの類似度,最小二乗法を用いた後の類似度などを出力する.そして指定したファイルに類似度が高い順に出力する.