令和５年度　特別研究報告書

ロボットアームを用いた
洗濯物片づけシステムの検討

龍谷大学　理工学部　情報メディア学科

T170503　田中　優希

指導教員　三好　力　教授

内容梗概

近年、日本では少子高齢化が進み介護人員が不足しており、今後も不足数が増加していく見込みである。そこで、洗濯物を自動で片づけるシステムがあると、介護士などの介護をする方の負担を減らすことが可能であると考えた。しかし、現在開発されている全自動衣類折りたたみロボットには靴下を片づけることができる機能が搭載されていない。靴下を自動で片づけるシステムを作るためには靴下を同じ種類同士で組み合わせなければならない。

そこで本研究では、ロボットアームを用いた洗濯物片づけシステムの一環として「ロボットアームを用いた靴下の色別振り分けシステム」を提案し、実験１では、自作したプログラムでロボットアームが想定通りに制御できていているか、またシステム全体が正常に作動するかをテストすることを目的として靴下を振り分ける実験を行う。また、実験２では、カメラで４枚の靴下を RGB 値により識別するために、4 枚それぞれの靴下の RGB 値を調べ、４枚の靴下を識別するためには、値をどのように区切ればよいのかを検討することを目的として実験を行い、４枚の靴下を色別に振り分けるためには RGB 値をどのような値で区切ればよいかを考察する。

目次

第 1 章　緒論

1.1　研究背景

近年、日本では少子高齢化が進んでおり介護士や介護が必要な方の家族など、介護人員が不足しており、今後も不足数が増加していく見込みである。2022 年の内閣府「令和 4 年版高齢社会白書」によれば、少子高齢化の進行により、日本の生産年齢人口（15〜64 歳）は 1995 年をピークに減少しており、2050 年には 5275 万人（2021 年から 29.2%減）に減少すると見込まれている。[1]高齢化の推移と将来推計を図 1.1 に示す。この少子高齢化の進行による高齢化率の上昇により、日本では、介護職員の不足が問題となっている。厚生労働省が令和 3 年 7 月 9 日に公表した、第 8 期介護保険事業計画の介護サービス見込み等に基づき、都道府県が推計した介護職員の必要数によれば、2025 年には約 243 万人（2019 年から 32 万人増）、2040 年には約 280 万人（2019 年から 69 万人増）の介護職員を確保する必要があると推計されている。[2]第 8 期介護保険事業計画に基づく介護人材の必要数についてのグラフを図 1.2 に示す。

介護職員の不足が年々増加していくと推計されていることから、介護職員や家族の介護をする方の負担を減らすため、家事を自動で行うシステムの開発が必要となっている。この家事を自動で行うシステムとして全自動衣類折りたたみ機が開発されている。しかし、この全自動衣類折り畳み機には靴下やハンカチなどの小さなものは片づけることができないというような問題点がある。

本研究では、全自動衣類折りたたみ機による問題点を改善するため、靴下に焦点を当て、靴下を色別に振り分けるシステムの開発を目標とする。

これにより、全自動洗濯物たたみ機の問題点を改善し、新たな機能を搭載することができるようになると期待される。

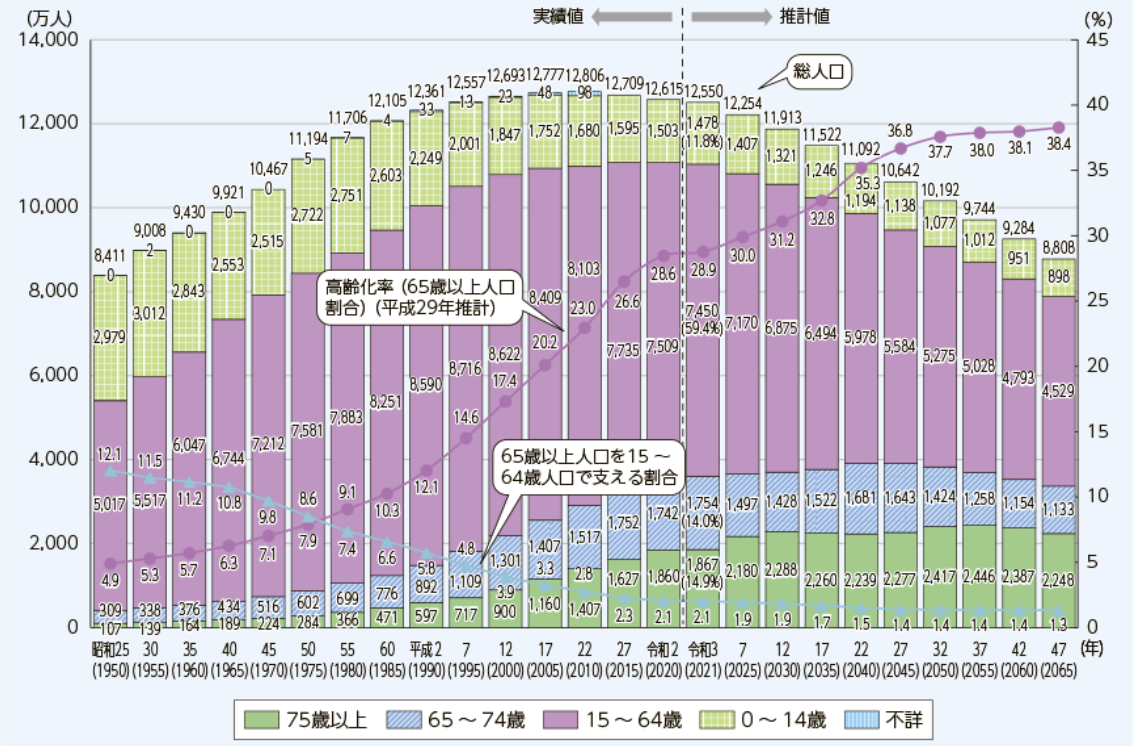

図 1.1　高齢化の推移と将来推計

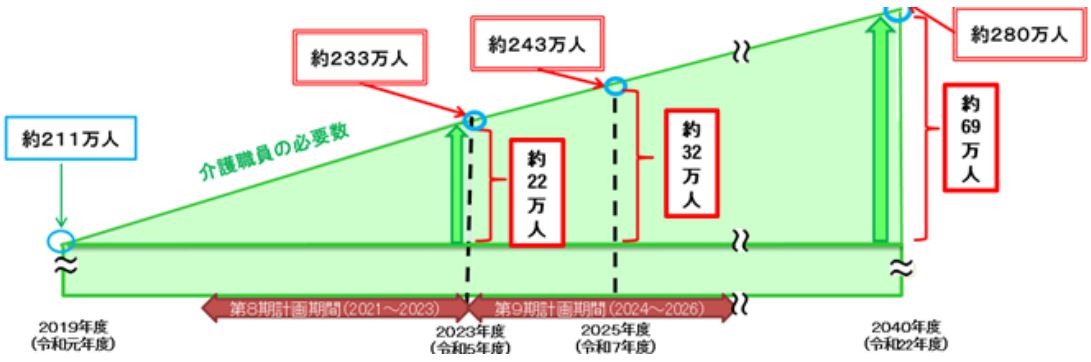

図 1.2　第 8 期介護保険事業計画に基づく介護人材の必要数について

（令和 3 年 7 月 9 日）

第 2 章　既存技術と問題点

2.1　全自動衣類折りたたみロボット

全自動衣類折りたたみロボットについては、家庭向けへの普及が望まれているものの、2022 年現在では普及はしていない。しかし、今日までにいくつかの企業が全自動衣類折りたたみロボットの開発を行ってきた。

2.2「全自動衣類折りたたみロボットの種類と特徴」では、セブン・ドリーマーズ・ラボラトリーズ社がパナソニック社と共同開発をしていたが、現在は開発を中止している「ランドロイド」と、2021 年に倒産したアメリカの Foldi　Mate 社が開発していた「Foldi　Mate」それぞれについて、説明する。

2.2 全自動衣類折りたたみロボットの種類と特徴

2.2.1　ランドロイド

「ランドロイド」は 2.1 でも記述した通り、セブン・ドリーマーズ・ラボラトリーズ社がパナソニック社と共同開発をしていた全自動衣類折りたたみ機である。[3]ランドロイドの全体像について図 2.1 に示す。

「ランドロイド」は、「画像解析」「人工知能」「ロボティクス」それぞれの技術の融合により、乾いた洗濯物をほうりこむだけでロボットアームが綺麗にたたんでくれるだけではなく、人工知能が衣類の特徴を覚え、タオル、T シャツ、ボトムスといった衣類ごとや、「お父さん」「子供」など、持ち主ごとに分けることもできる[4]。

「ランドロイド」が片づけることができる衣類の種類は、T シャツ、カットソー、ボトムス、ホームウェア、タオル（フェイスタオル、バスタオル）である。

2.2.2　Foldi　Mate

「Foldi　Mate」は 2.1 でも記述した通り、アメリカの Foldi　Mate 社が開発していた全自動衣類折りたたみロボットである[5]Foldi　Mate の全体像について図 2.2 に示す。

「Foldi　Mate」は、T シャツやタオル、ズボンなどの洗濯物をたたんでくれるロボットであり、手作業に比べ 3 倍速いという 1 枚当たり 10 秒以内で、プロ並みに美しくたたむことができる[6]。

図 2.1　ランドロイドの全体画像

図 2.2　Foldi　Mate の全体像

2.3　既存技術の問題点

既存技術の全自動衣類おりたたみロボットである「ランドロイド」と「Foldi　Mate」では、T シャツやタオル、ボトムスといった比較的大きな衣類を片づけることが可能であるが、ハンカチや靴下といった比較的小さな衣類を片づける機能がついていないことが問題点であると考えられる。

第 3 章　提案手法

2.3 で示した問題点を解決するために、「ロボットアームを用いた靴下の色別振り分けシステム」を提案する。この手法により、全自動衣類おりたたみロボットで靴下を片づける際に必要となる靴下を色別に振り分ける作業が可能になると考えた。

本研究では、ロボットアーム「My Palletizer 260 M5Stack」を使用して、靴下を色別に振り分けるシステムを提案する。システムの全体像を、図 3.1 に示す。このシステムは、画像中央の位置でロボットアームに 4 つ折りにした靴下を順に手渡ししていき、自作したプログラムの変数を色に対応した数に変更することで、色ごとに靴下を振り分けることができるものである。当初は、所定の位置に置いた靴下をロボットアームで色別に振り分ける予定であったが、アームのつかむ力が弱く、これ以上強い力にすることができなかったため、靴下を 4 つ折りにして、手渡しする仕様へ変更した。なお、靴下には様々な色のものや柄付きのものが存在しているが、本研究では無地の靴下で、黒色、オレンジ色、水色、黄緑色の物を対象とする。色に対応するプログラムの変数は黄緑色が 0、オレンジ色が 1 、水色が 2、黒色が 3 である。

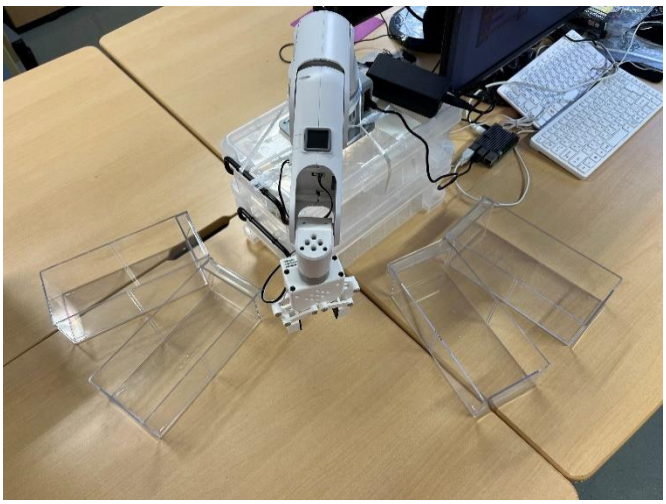

図 3.1 システムの全体像

第 4 章　実験

4.1　実験 1　ロボットアーム

本実験の目的は、自作したプログラムでロボットアームが想定通りに制御できているかまた、システム全体が正常に作動するかをテストすることである。

本実験では、作成したプログラムで Elephant Robotics 社のロボットアームである MyPalletizer 260 M5Stack を制御し、靴下を色別に振り分ける実験を行った。本実験では、黄緑色、オレンジ色、水色、黒色の 4 色で無地の靴下を 1 足（片足ずつ数えると 2 枚）ずつ使用した。この 4 足の靴下（片足ずつ数えると 8 枚）を 1 枚ずつ無作為に順番を決め、所定の位置でロボットアームにわたし、それぞれの色に対応したプログラムでロボットアームを動かし色ごとに振り分ける実験を行った。

4.1.1　実験環境

本実験にて使用した実験環境を以下に示す。開発環境として、OS は Ubuntu の 22.04、プログラミング言語として Python の 3.10.12 を使用した。また MyPalletizer を動かすためのファームウェアをコントローラである m5stack に書き込むために my studio の 3.5.3 を使用した。またプログラムを書き込むために myblockly の 1.3.8 を使用した。

Ubuntu 22.04.3 LTS
Python 3.10.12

myStudio 3.5.3
myblockly1.3.8

4.1.2 実験器具

実験には Elephant Robotics 社のロボットアーム MyPalletizer 260 M5stack を使用した。今回使用した MyPalletizer 260 M5stack の画像を図 4.1 に示す。MyPalletizer には軸が 4 軸あり、作業半径が 260 ㎜でコントローラに M5Stack basic を使用する。

図 4.1　MyPalltizer 260 M5stack の全体画像

4.1.3　作成したプログラム

本実験で使用したプログラムを付録 A に示す。なお、本実験ではプログラム中の変数 num を靴下の色に対応した数へと変更し用いる。0 が黄緑色、1 がオレンジ色、2 が水色、3 が黒色に対応している。

4.1.4　実験方法

本実験の実験装置の全体像を図 4.1 に示す。本実験では図 4.1 の装置を用いて、黄緑色、オレンジ色、水色、黒色の 4 色で無地の靴下 1 足ずつを透明の箱に振り分ける実験を行った。実験手順を説明する。まず始めに計 8 枚の靴下をどの順番で振り分けるかを無作為に決定した。次に、 1 番目に選ばれた靴下の色を確認しプログラムの変数 num を対応する数字に変更した。なお、対応する数字は 0 が黄緑色、1 がオレンジ色、2 が水色、3 が黒色である。変数を変更した後、選ばれた靴下を 4 つ折りにし、手で持った状態で図 4.1 の中央部分で保持した。その後、プログラムを実行した。するとロボットアームが下りてくるので、靴下を渡すことで靴下を色ごとに振り分けた。振り分けた靴下の色は図 4.1 の左の箱から黒色、水色、オレンジ色、黄緑色とした。これを靴下が 8 枚あるため 8 回繰り返した。また、実験の正確性向上のため計 8 枚の実験を 3 セット行った。

図 4.1 実験装置の全体像

4.1.5　実験結果

実験にて、無作為に選ばれた靴下の振り分ける順番を表 4.1 に示す。

	1 枚目	2 枚目	3 枚目	4 枚目	5 枚目	6 枚目	7 枚目	8 枚目
1 回目	水色	黒色	黒色	オレンジ色	水色	黄緑色	オレンジ色	黄緑色
2 回目	オレンジ色	黒色	水色	黄緑色	黒色	黄緑色	水色	オレンジ色
3 回目	黄緑色	オレンジ色	オレンジ色	黄緑色	黒色	水色	黒色	水色

表 4.1　無作為に選ばれた靴下を振り分ける順番

1 セット目の 1 枚目から 8 枚目までの靴下の振り分けをそれぞれ行った後の画像を図 4.2 ～図 4.9 に示す。図 4.2 から水色靴下が想定通り振り分けられていることがわかる。図 4.3 から黒色靴下が想定通り振り分けられていることがわかる。図 4.4 から黒色靴下が想定通り振り分けられていることがわかる。図 4.5 からオレンジ色靴下が想定通り振り分けられていることがわかる。図 4.6 から水色靴下が想定通り振り分けられていることがわかる。図 4.7 から黄緑色靴下が想定通り振り分けられていることがわかる。図 4.8 からオレンジ色靴下が想定通り振り分けられていることがわかる。図 4.9 から黄緑色靴下が想定通り振り分けられていることがわかる。

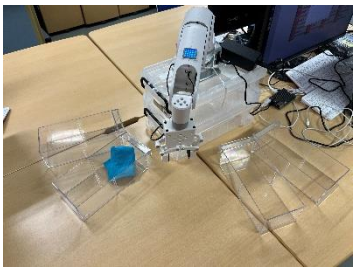

図 4.2　1 枚目（水色）

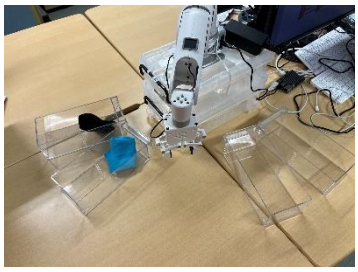

図 4.3　2 枚目（黒色）

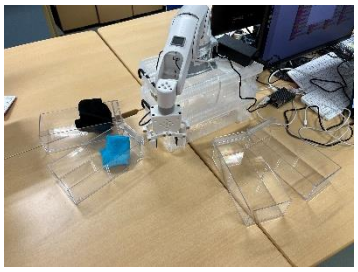

図 4.4　3 枚目（黒色）

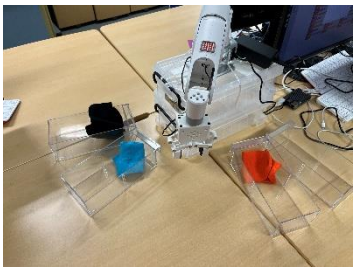

図 4.5　4 枚目（オレンジ色）

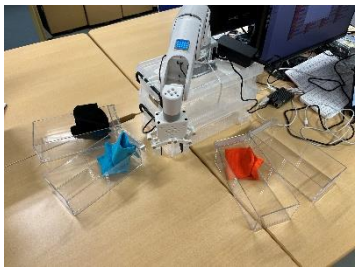

図 4.6　5 枚目（水色）

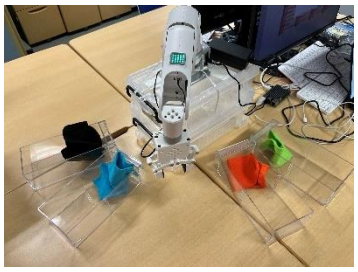

図 4.7　6 枚目（黄緑色）

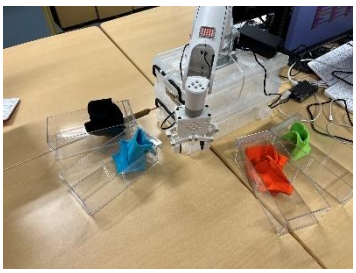

図 4.8　7 枚目（オレンジ色）

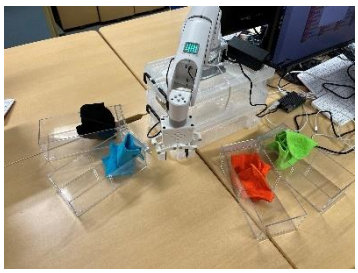

図 4.9　8 枚目　（黄緑色）

2 セット目の 1 枚目から 8 枚目までの靴下の振り分けをそれぞれ行った後の画像を図 4.10〜図 4.17 に示す。図 4.10 からオレンジ色靴下が想定通り振り分けられていることがわかる。図 4.11 から黒色靴下が想定通り振り分けられていることがわかる。図 4.12 から水色靴下が想定通り振り分けられていることがわかる。図 4.13 から黄緑色靴下が想定通り振り分けられていることがわかる。図 4.14 から黒色靴下が想定通り振り分けられていることがわかる。図 4.15 から黄緑色靴下が想定通り振り分けられていることがわかる。図 4.16 から水色靴下が想定通り振り分けられていることがわかる。図 4.17 からオレンジ色靴下が想定通り振り分けられていることがわかる。

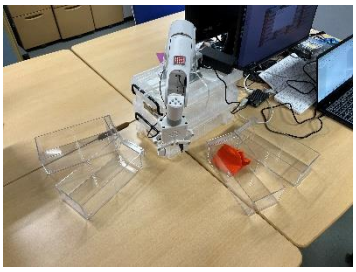

図 4.10　1 枚目（オレンジ色）

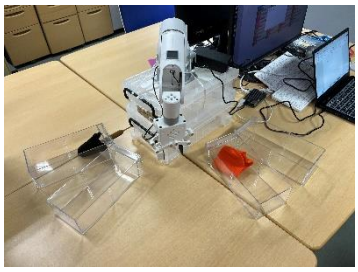

図 4.11　2 枚目（黒色）

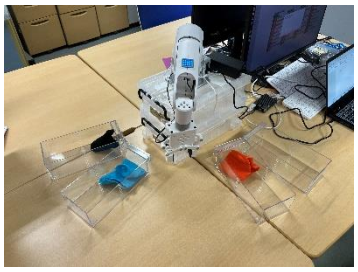

図 4.12　3 枚目（水色）

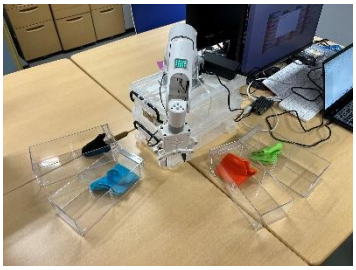

図 4.13　4 枚目（黄緑色）

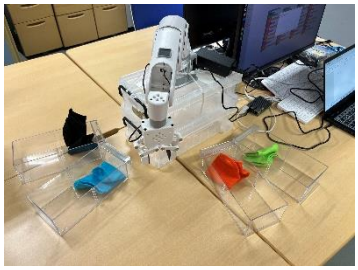

図 4.14　5 枚目（黒色）

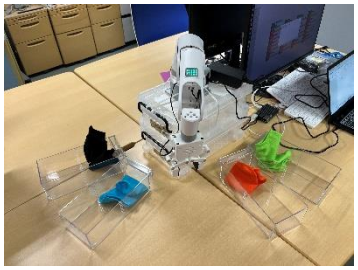

図 4.15　6 枚目（黄緑色）

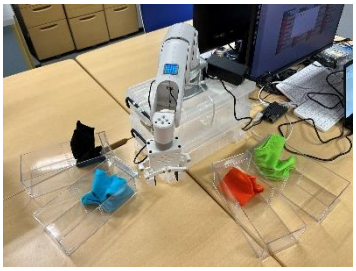

図 4.16　7 枚目（水色）

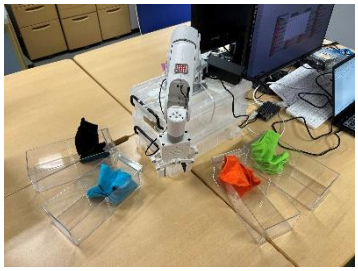

図 4.17　8 枚目（オレンジ色）

3 セット目の 1 枚目から 8 枚目までの靴下の振り分けをそれぞれ行った後の画像を図 4.18～図 4.25 に示す。図 4.18 から黄緑色靴下が想定通り振り分けられていることがわかる。図 4.19 からオレンジ色靴下が想定通り振り分けられていることがわかる。図 4.20 からオレンジ色靴下が想定通り振り分けられていることがわかる。図 4.21 から黄緑色靴下が想定通り振り分けられていることがわかる。図 4.22 から黒色靴下が想定通り振り分けられていることがわかる。図 4.23 から水色靴下が想定通り振り分けられていることがわかる。図 4.24 から黒色靴下が想定通り振り分けられていることがわかる。図 4.25 から水色靴下が想定通り振り分けられていることがわかる。

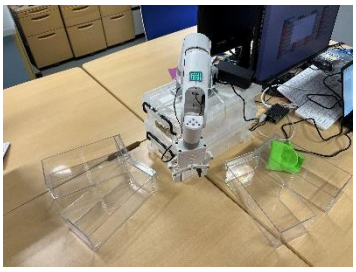

図 4.18　1 枚目（黄緑色）

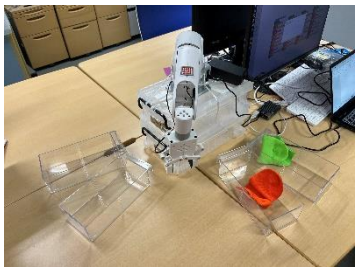

図 4.19　2 枚目（オレンジ色）

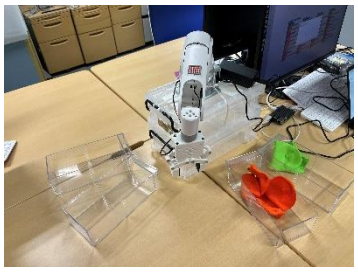

図 4.20　3 枚目（オレンジ色）

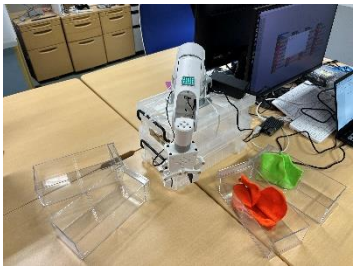

図 4.21（黄緑色）

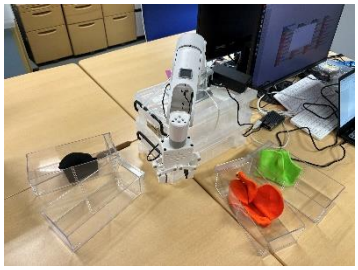

図 4.22（黒色）

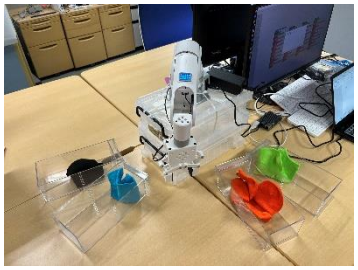

図 4.23（水色）

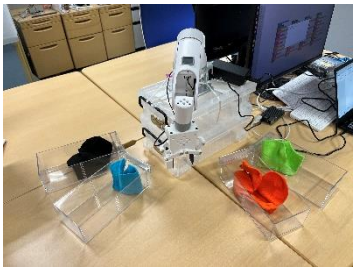

図 4.24（黒色）

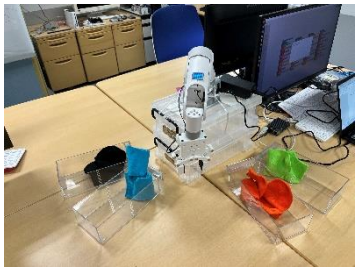

図 4.25（水色）

4.1.7　考察

今回、自作したプログラムでロボットアームが想定通りに制御できているか、また、システム全体が正常に作動するかをテストすることを目的として実験を行った。結果は、1 回目、2 回目、3 回目のすべての実験でロボットアームが想定通りに動き、左から順に黒色、水色、オレンジ色、黄緑色と靴下を同じ色でそれぞれ振り分けることができたため、実験は成功したといえる。

4.2　実験２　カメラによる色識別

実験 2 の目的は、カメラで 4 枚それぞれの靴下を RGB 値により識別するために、4 枚それぞれの靴下の RGB 値を調べ、4 枚の靴下を識別するためには、値をどのように区切ればよいか考察することである。

本実験では、カメラで撮影した 4 つの靴下の画像を、カラー画像の RGB 値をテキストファイルに出力するプログラム[7]にかけることで、4 つの靴下の RGB 値を調べる実験を行った。またその結果から、カメラで自動的に 4 つの靴下を識別するためには、値をどのように区切ればよいか考察した。

4.2.1　実験環境

本実験にて使用した実験環境を以下に示す。開発環境として、OS は Ubuntu の 22.04、プログラミング言語として Python の 3.10.12 を使用した。また、カラー画像の RGB 値をテキストファイルに出力するプログラムを使用するためのライブラリとして OpenCV の 4.8.0 を使用した。

Ubuntu 22.04
Python 3.10.12

OpenCV 4.8.0

4.2.2 使用したプログラム

本実験で使用したプログラムを付録 B に示す。

4.2.3　実験方法

本実験では、カメラにて黒色、水色、オレンジ色、黄緑色の 4 枚の靴下の画像を撮影した。それらの靴下の画像を図 4.26〜図 4.29 とした。また、このままの画像では背景が入ってしまっているため靴下部分のみを切り取った画像を図 4.30〜図 4.33 とした。この図 4.30〜図 4.33 の画像ファイルをプログラムにかけることで、靴下の RGB 値が記載されたテキストファイルを出力させた。出力したテキストファイルを分析することで、それぞれの靴下の RGB 値を求め、 4 枚の靴下を識別するためには、値をどのように区切ればよいか考察した。

4.2.4　実験結果

実験 2 でカメラにて撮影した画像を図 4.26〜4.29 として以下に示す。

図 4.26
black.JPG

図 4.27
blue.JPG

図 4.28
orange.JPG

図 4.29
green.JPG

図4.26～図4.29から靴下部分のみを切り取った画像を図4.30～図4.33として以下に示す。

図 4.30
black_2.JPG

図 4.31
blue_2.JPG

図 4.32
orange_2.JPG

図 4.33
green_2.JPG

図 4.30～図 4.33 の画像のピクセル数は 738×738 である。

図 4.30～図 4.33 の画像をプログラムにかけ、それぞれの靴下の画像 1 枚につき R、G、B の三つのファイルが出力された。それぞれのファイルには、738×738 のそれぞれのピクセルの RGB 値が出力された。今回の実験では、それぞれのデータから 12 ピクセル分を使用した。使用する部分は、それぞれの画像の左上、右上、左下、右下からそれぞれ 3 ピクセル分とする。

図 4.30～図 4.33 の画像をプログラムにかけ、出力されたデータから左上、右上、左下、右下からそれぞれ 3 ピクセル分を抽出したデータの表を表 4.2～表 4.5 として以下に示す。なお、中央値、平均値は小数点以下第一位を四捨五入し計算する。

	左上	右上	左下	右下	中央値	平均値
R 値	42,44,64	82,80,74	38,43,46	43,42,48	45	54
G 値	41,43,65	84,80,74	43,48,51	46,45,51	50	56
B 値	46,48,69	83,80,74	49,54,57	51,50,56	55	60

表 4.2　black_2.JPG の RGB 値

	左上	右上	左下	右下	中央値	平均値
R 値	20,32,26	0,3,22	0,0,0	0,0,0	0	9
G 値	197,208,197	134,150,174	134,132,130	142,146,148	147	158
B 値	247,255,253	202,219,241	224,225,231	228,231,232	231	232

表 4.3　blue_2.JPG の RGB 値

	左上	右上	左下	右下	中央値	平均値
R 値	203,230,246	204,226,255	255,245,235	192,194,198	228	224
G 値	82,112,128	81,103,139	121,109,99	59,61,65	101	97
B 値	37,66,82	40,62,98	79,67,57	24,26,30	60	56

表 4.4　orange_2.JPG の RGB 値

	左上	右上	左下	右下	中央値	平均値
R 値	125,121,142	135,118,112	86,89,93	121,69,76	115	107
G 値	218,214,235	228,211,205	167,170,174	215,163,170	208	198
B 値	173,169,190	181,164,158	124,127,131	165,113,120	161	151

表 4.5　green_2.JPG の RGB 値

考察では、このデータを基に、カメラで 4 枚それぞれの靴下を RGB 値により識別するためには、値をどのように区切ればよいか考察する。

4.2.5 考察

考察では実験の結果から、 4 枚の靴下を識別するためには値をどのように区切ればよいか考察する。4 枚の靴下の RGB 値を比べた時に、特徴的な値として水色靴下の B 値が 200 を超えていること、オレンジ靴下の R 値が 190 を超えていることがあげられる。この特徴的な値から、水色靴下とオレンジ靴下の判別を行う際は、12 ピクセルの RGB 値をカメラで読み取り、その平均値が、B 値が 200 を超えていた場合は水色、R 値が 190 を超えていた場合はオレンジ色と判別すればよいと考えられる。次に黄緑靴下であるが、G 値に少し特徴があったが、水色靴下の G 値も似たような数値であるため、G 値のみからの判別は難しいと考えられる。そこで水色靴下の R 値が 50 以下の数値を出しているところに着目し、黄緑靴下の判別は、G 値が 150 以上かつ R 値が 50 以上の場合に黄緑色と判断すればよいと考えられる。最後に黒靴下の判別であるが、他 3 つの靴下にそれぞれ 150 を超える値があることに着目し、黒靴下の判別は R 値が 150 以下かつ G 値が 150 以下かつ B 値が 150 以下の場合に黒色と判断することができると考えられる。

第 5 章　結論

本研究では、「ロボットアームを用いた靴下の色別振り分けシステム」を提案した。

実験 1 では、プログラムを自作しロボットアームを用いて靴下を色別に振り分ける実験を行い、想定通りに作動することを確認した。実験 2 では、 4 枚の靴下の RGB 値を求め、 4 枚の靴下を識別するためには、値をどのように区切ればよいか考察した。

当初の予定ではカメラを用いて色を読み取り、自動で色別に靴下を振り分けるシステムを目指していたが、そこまで実験をすることができなかった。

このシステムの今後の展望としては、実験 2 で実験した結果を基に、カメラで靴下の RGB 値を読み取り、B 値が 200 を超えていた場合は水色、R 値が 190 を超えていた場合はオレンジ色、G値が 150 以上かつ R値が 50 以上の場合に黄緑色、R 値が 150 以下かつ G 値が 150 以下かつ B 値が 150 以下の場合に黒色と判断し、これを実験 1 で作成したプログラムと統合することで、4 つの靴下を振り分けるシステムを自動化することが可能であると考えられる。

謝辞

本研究を進めるにあたり、様々なご指導をいただきました三好力教授に深く感謝いたします。また、多忙の中研究の過程で様々な助言をいただきました同研究室の皆様に深く感謝いたします。

参考文献

[1].総務省
＜https://www.soumu.go.jp/johotsusintokei/whitepaper/ja/r04/html/nd121110.html＞
[2].厚生労働省
<https://www.mhlw.go.jp/stf/newpage_02977.html＞
[3].ランドロイドワン｜夢のような未来家電！？話題の全自動洗濯物たたみ機！
<https://magazine.premoa.co.jp/laundroid/>
[4].全自動衣類折りたたみ機「ランドロイド」の詳細と新サービスが発表
<https://kakakumag.com/seikatsu-kaden/?id=10365>
[5].Foldimate, le robot qui s'occupe de vos linges
< https://www.innovant.fr/2016/11/11/foldimate-robot-soccupe-de-vos-linges/>
[6].洗濯物を畳んでくれるロボット「FoldiMate」--800 ドル前後で 2017 年発売
＜https://japan.cnet.com/article/35084681/＞
[7]. 【Python-OpenCV】カラー画像の RGB 値をテキストファイルに出力する方法！
https://www.higashisalary.com/entry/cv2-img-rgb-txt#google_vignette

付録 A

```
from pymycobot.mycobot import MyCobot
import time
num = None
mc = MyCobot('/dev/ttyAMA0',115200)
num = 0
if num == 0:
    mc.set_color(103,196,56)
    time.sleep(4)
    mc.send_angles([(-90),0,(-90),0],10)
    time.sleep(4)
    mc.set_gripper_state(0, 10)
    time.sleep(4)
    mc.send_angles([(-90),0,(-65),0],10)
    time.sleep(4)
    mc.set_gripper_state(1, 10)
    time.sleep(4)
    mc.send_angles([(-90),0,(-90),0],10)
    time.sleep(4)
    mc.send_angles([(-20),0,(-90),0],10)
    time.sleep(4)
    mc.set_gripper_state(0, 10)
    time.sleep(4)
    mc.send_angles([(-90),0,(-90),0],10)
    time.sleep(4)

if num == 1:
    mc.set_color(247,84,19)
    time.sleep(4)
    mc.send_angles([(-90),0,(-90),0],10)
    time.sleep(4)
    mc.set_gripper_state(0, 10)
    time.sleep(4)
    mc.send_angles([(-90),0,(-65),0],10)
    time.sleep(4)
    mc.set_gripper_state(1, 10)
```

```
    time.sleep(4)
    mc.send_angles([(-90),0,(-90),0],10)
    time.sleep(4)
    mc.send_angles([(-55),0,(-90),0],10)
    time.sleep(4)
    mc.set_gripper_state(0, 10)
    time.sleep(4)
    mc.send_angles([(-90),0,(-90),0],10)
    time.sleep(4)

if num == 2:
    mc.set_color(0,136,204)
    time.sleep(4)
    mc.send_angles([(-90),0,(-90),0],10)
    time.sleep(4)
    mc.set_gripper_state(0, 10)
    time.sleep(4)
    mc.send_angles([(-90),0,(-65),0],10)
    time.sleep(4)
    mc.set_gripper_state(1, 10)
    time.sleep(4)
    mc.send_angles([(-90),0,(-90),0],10)
    time.sleep(4)
    mc.send_angles([(-125),0,(-90),0],10)
    time.sleep(4)
    mc.set_gripper_state(0, 10)
    time.sleep(4)
    mc.send_angles([(-90),0,(-90),0],10)
    time.sleep(4)

if num == 3:
    mc.set_color(2,7,11)
    time.sleep(4)
    mc.send_angles([(-90),0,(-90),0],10)
    time.sleep(4)
    mc.set_gripper_state(0, 10)
```

```
time.sleep(4)
mc.send_angles([(-90),0,(-65),0],10)
time.sleep(4)
mc.set_gripper_state(1, 10)
time.sleep(4)
mc.send_angles([(-90),0,(-90),0],10)
time.sleep(4)
mc.send_angles([(-160),0,(-90),0],10)
time.sleep(4)
mc.set_gripper_state(0, 10)
time.sleep(4)
mc.send_angles([(-90),0,(-90),0],10)
time.sleep(4)
```

付録 B

```
#ライブラリのインポート
import cv2
file_name='sample.jpg'
pic=cv2.imread(file_name)
h,w=pic.shape[:2]
pic=cv2.cvtColor(pic,cv2.COLOR_BGR2RGB)
RGB=['R.txt','G.txt','B.txt']
for k in range(3):
    txt=open(RGB[k],'w')
    array=pic[:,:,k]
    for j in range(h):
        for i in range(w):
            txt.write(str(array[j,i])+',')
        txt.write('¥n')
    txt.close()
```