卒業論文

OpenPose による姿勢推定技術を用いた
動作状況評価手法の提案

松田　征也

令和 2 年 2 月 18 日

熊本大学
情報電気電子工学科

論文概要

人物の姿勢推定(Human pose estimation)は，人の頭部の検出だけでなく，肩，肘，手，腰，膝，足を検出し，人がどのような姿勢を取っているかを推定する技術である．数年前までは Deformable part model などの技術が用いられていたが，近年 Deep Learning が用いられるようになり飛躍的に性能が向上した．近年，姿勢推定技術の一つとして最も注目を集めているのが，カーネギーメロン大学の ZheCao らによって CVPR2017 で発表された「Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields」が実装されたライブラリである OpenPose である．

OpenPose は，動画像を入力するだけで，画像内の複数人物のポーズをほぼリアルタイムで推定することができる．高い推定精度と高速処理を両立しており，COCO 2016 keypoints challenge で 1 位に輝き，MPII Multi-Person benchmark において効率，精度ともに良い成績を残している．

本研究室では今年度より OpenPose を用いたシステムの開発に関する研究を行っている．本論文では災害時の救助活動を円滑に進めるため監視カメラやドローンなどに設置されたネットワークカメラより救助者を発見する活動支援に役立てるシステムの開発を見据え，OpenPose のアルゴリズムの理解と Python を用いて検出結果のデータ処理を行うプログラムの開発を目的とする．作成したプログラムの概要としては関節の角度を算出し時系列データを比較することで人物の状態を判別するものである．論文の前半では Windows への OpenPose の現状，アルゴリズムについての説明を行う．論文の後半では手法の提案，作成したプログラムが実際に利用できるかを検証する．本研究によって，動画像内の複数人に対し，点数の算出を行い得点化することに成功した．しかし，しきい値を利用した評価手法については今後改善が必要とされる．また，OpenPose とネットワークカメラを組み合わせることで今後の研究に役立てることができると考えられる．

目次

第1章 序論

1.1 はじめに

人物の姿勢推定(Human pose estimation)は，人の頭部の検出だけでなく，肩，肘，手，腰，膝，足を検出し，人がどのような姿勢を取っているかを推定する技術である．数年前までは Deformable part model などの技術が用いられていたが，近年 Deep Learning が用いられるようになり飛躍的に性能が向上した．

近年，姿勢推定技術の一つとして最も注目を集めているのが，カーネギーメロン大学の ZheCao[1]らによって CVPR2017 で発表された「Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields」が実装されたライブラリである OpenPose である．

動画像を入力するだけで，画像内の複数人物のポーズをほぼリアルタイムで推定することができる．高い推定精度と高速処理を両立しており，COCO 2016 keypoints challenge で 1 位に輝き，MPII Multi-Person benchmark において効率，精度ともに良い成績を残している[2].

OpenPose はモーションキャプチャと呼ばれる技術の「人間の身体にセンサを取り付けないと，関節点の情報を取得できない」という課題を解決するものである．OpenPose の利点は既存のモーションキャプチャ技術と比較し，特殊センサを使用しなくても，カメラ 1 つあれば，複雑な解析ができてしまうという点にある[3].

OpenPose を利用することで監視カメラやドローンなどに設置されたネットワークカメラより救助者を発見する捜索活動支援システムの開発が可能であると考えた．このシステムの開発により，従来よりも高精度かつ複数人に対し，災害時の救助活動をリアルタイムで円滑に進めることが可能になると考える．その準備として，本論文では Python を用いて作成したデータ処理プログラムの有効性についての検討を行う．

1.2 本論文の目的及び構成

本研究室では OpenPose を用いたシステムの開発に関する研究を今年度から行っている. 本論文は災害時の救助活動や捜索活動の際, 人が立ち入ることができない場所で監視カメラやドローンカメラなどを用いて救助者の状態や人数などを判別するシステムの開発を目的とする.

OpenPose は動画像内の複数人に対し関節点情報をリアルタイムにデータとして出力することができる. しかし, OpenPose は一定の誤検出やデータをフレームごとに出力するため, 各フレーム間での人物同士のデータを照合しきれていないなど本システムの開発を進めるにあたり障壁となる課題がある.

本論文では実際に OpenPose を用いて姿勢推定を行い, 上記の課題を解決するための取得した関節点情報に対するデータ処理の手法を提案する. また, 処理を行ったデータより人物の動作を判別する手法を提案し実験を行う.

論文の序盤では, OpenPose は近年, 姿勢推定技術の一つとして開発された最新技術であるため OpenPose についての詳細な説明を行う.

本論文は全 5 章で構成されている. 以下にそれぞれの概要について述べる. 第 2 章では, OpenPose の現状についての解説, 従来の姿勢推定技術との比較, アルゴリズムについての説明を行う. 第 3 章では, 本システムの基盤となる関節点情報からデータ処理を行い人物の動作の評価を行うための手法の提案を行う. 第 4 章では, 第 3 章で提案した手法を元に 3 種類の動画に対してデータ処理を行った実験の結果, 人物の動作を判別した結果を示す. 第 5 章では, 第 4 章より得られた実験結果をもとに今後の展望を交えて, 実際に目的を達成するための可能性を示すとともに本論文の結果より結論を導く.

第2章 OpenPose について

本章では，本研究の前提となっているOpenPoseの概要と従来の姿勢推定技術について述べる．

2.1 Deep lerning による姿勢推定技術の発達

数年前までは姿勢推定技術は Kinect を代表とするデプスカメラ（3 次元情報をリアルタイムに取得できるカメラ）を用いたものが一般的であった．しかし近年，Deep lerning を始めとする高度な画像処理技術の発達より，単眼カメラでのスケルトン検出アルゴリズムを用いた姿勢推定技術が開発された．従来の姿勢推定技術と比較し，特殊センサを使用する必要がなくカメラ一つで複雑な解析が可能である．この姿勢推定技術として OpenPose，VisionPose，DeepPose などが挙げられる．この姿勢推定技術の発達により，関節点取得による人物の姿勢推定や動作の判別を行うことで，作業支援や情報提示といった形で医療・スポーツ・災害・障害者支援・犯罪防止・エンターテインメントなどの様々な分野への応用が期待されている．

2.2 OpenPose の概要

本研究では，カーネギーメロン大学の ZheCao[1]らによって CVPR2017 で発表された「Realtime Multi-Person 2DPose Estimation using PartAffinityFields」が実装されたライブラリである OpenPose を用いる．OpenPose は GitHub から無償公開されている人物の姿勢推定解析処理である．

OpenPose とは，単一画像から複数の人間の身体や顔のキーポイントをリアルタイムに検出することができる機械学習型動作解析処理である[1]. Convolution Neural Network を用いて，画像に映る人物の肩や肘など 25 点の位置推定を行う．画像を入力すると，画像から検出した各点にマークとマーク間を線で結び人体モデルを表示した出力画像を得られる．また，各点の座標データを JSON,

XML，YML 形式の出力ファイルとして得られる．なお，検出できないキーポイントの座標データは，x 座標と y 座標ともに 0 となる．図 2.2.1 を入力画像とした場合，図 2.2.2 のような出力画像になる．

図 2.2.1 入力画像[4]

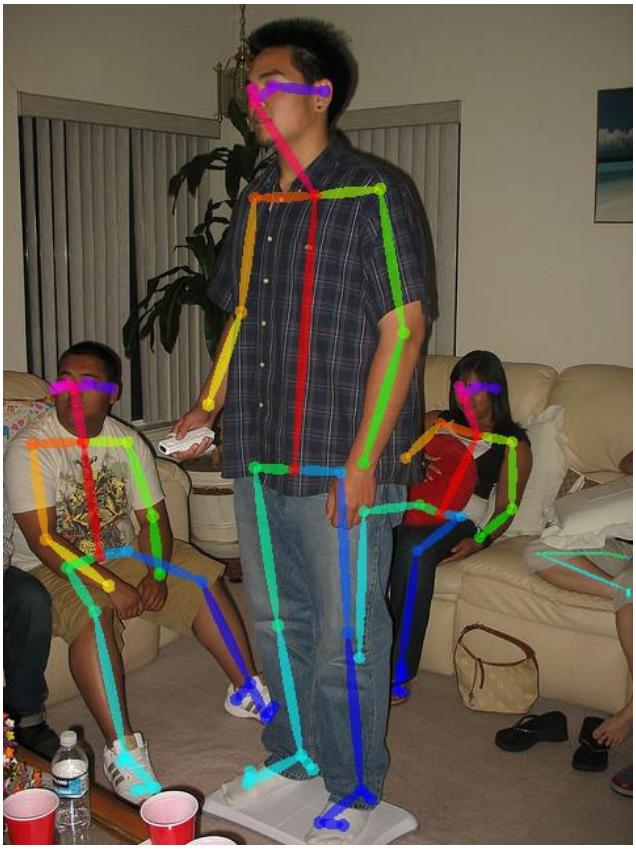

図 2.2.2 出力画像[4]

また，OpenPose は画像だけでなく動画に対しても利用可能である．動画を入力した場合，動画の各フレームに対して人物の姿勢推定を行い，座標データをフレームごとに別ファイルで出力される．

2.3 OpenPose の利点と課題

人物の動作解析では OpenPose 以外にモーションキャプチャやレーザーセンサを用いた手法がある．モーションキャプチャとは，マーカを取り付けた計測対象を複数台のカメラで撮影し，人物の動作を計測するアプローチである．スポーツ分野での人物の動作データの収集だけでなく，映画やゲームにおける CG で

作成されたキャラクターの動作の再現にも使用されている．レーザーセンサを用いた動作解析は，対象物に向けて当てたレーザーが反射して戻るまでの時間から距離を測定し，人物の動作を 3D で捕らえることができる．モーションキャプチャとは違いマーカを付ける必要がないため，人物の動作を制限しない．これらの方法では複数台のカメラ・マーカまたはレーザーセンサが必要となる．また，解析が撮影時にカメラやセンサの前でしか行えない．一方で，OpenPose はウェブカメラで撮影された動画や録画されている動画に対してキーポイントの位置推定が行える．また，スマートフォンなどの動画のように形式を限定せず，気軽にキーポイントの位置推定が行える．OpenPose は実写の人間だけではなく，写実的なイラストや絵画にも効果を発揮する[5]．絵画に対してキーポイントの位置推定を行った際の出力画像を図 2.3.1 に示す．

このため，特別な機器が必要ない上に撮影場所が自由である．よって，他の動作解析方法より多くの人物の解析に利用できる OpenPose による動作解析が，災害時の救助活動を目的としている本研究に適している．しかし，画像内で人物同士や人物と物が重なると誤検出が多くなる．すべてのキーポイントを検出できない場合や，ある検出人物と重なる人物の身体の一部を検出人物の身体の一部だと誤って検出してしまう場合がある．人物と物が重なりすべてのキーポイントを検出できない例を図 2.3.2 に示す．

また，OpenPose でキーポイントの位置推定を行うにはグラフィック処理を行うことのできる実行環境が必要とされる．現在，Github には GPU を用いて処理を行うソースコードと CPU を用いて処理を行うソースコードの二種類が公開されている．しかし，処理速度には大きな差があり課題となっている．

OpenPose の実行環境として推奨される PC のスペックを表 2.3.1 に示す[6].

図 2.3.1 絵画に対して位置推定を行った際の出力画像[5]

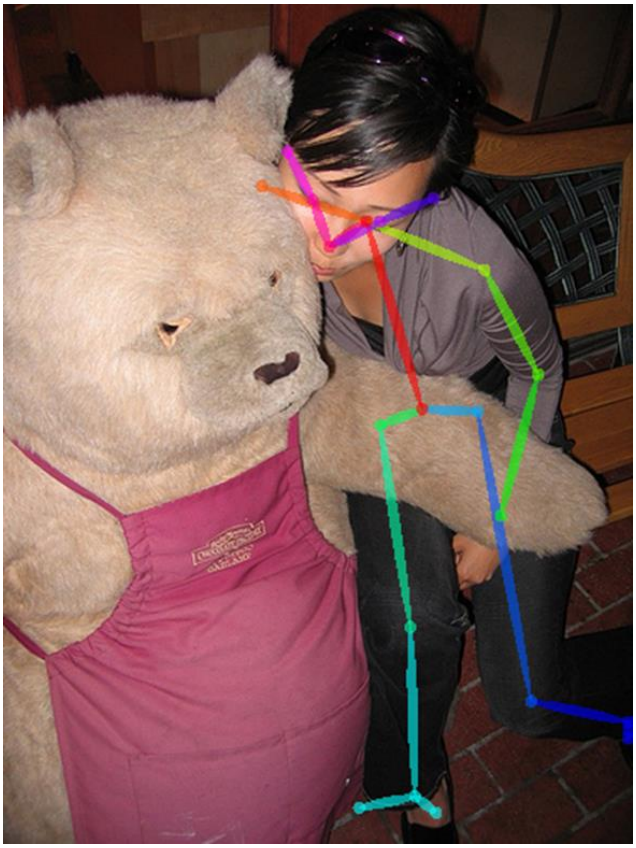

図 2.3.2 実際に生じる誤検出の例[4]

表 2.3.1 推奨される PC のスペック

OS	Ubuntu 14 / 16 Windows 8 / 10
GPU バージョン	NVIDIA 1.6GB 以上のグラフィックボード 2GB 以上のメモリ 8 コア以上の CPU
CPU バージョン	8GB 以上のメモリ 8 コア以上の CPU

2.4 OpenPose により得られるキーポイント

OpenPose は年々バージョンアップされており，現在 Github に OpnePose v1.5.1 が最新版として無償で公開されている．現在では身体に加えて，手，顔，足のキーポイント計 135 点を取得することができる．今回，本研究で使用する身体では 15 点，18 点，25 点の解析が可能となっている．

本研究で利用する身体のキーポイントを図 2.4.1，キーポイントの各番号を表 2.4.1 示す．また本研究で利用はしないが，実際に手と顔のキーポイント検出をした際の出力画像をそれぞれ図 2.4.2，図 2.4.3 に示す．

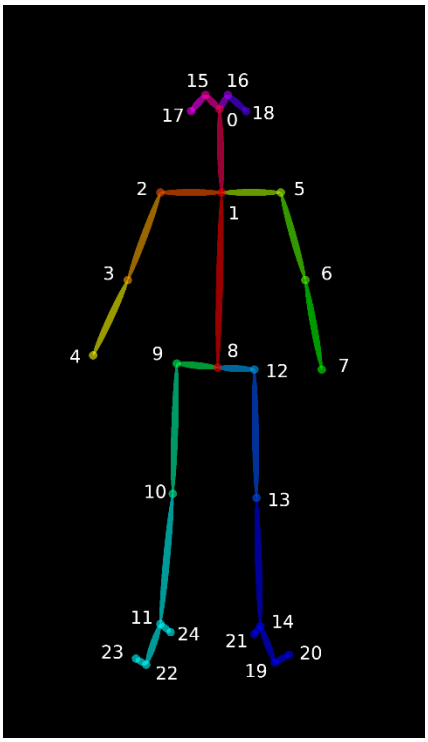

図 2.4.1 検出されるキーポイント

表 2.4.1 キーポイントの番号と名称

番号	部位	12	左腰
0	鼻	13	左膝
1	首	14	左足首
2	右肩	15	右目
3	右肘	16	左目
4	右手首	17	右耳
5	左肩	18	左目
6	左肘	19	左足付け根
7	左手首	20	左足爪先
8	腰の中心	21	左足踵
9	右腰	22	右足付け根
10	右膝	23	右足爪先
11	右足首	24	右足踵

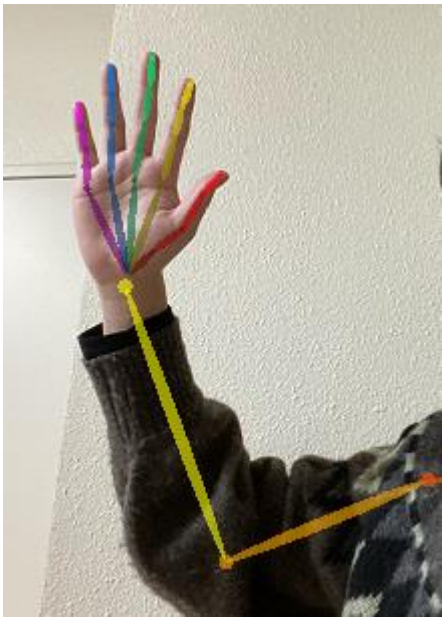

図 2.4.2 手のキーポイントを検出した出力画像

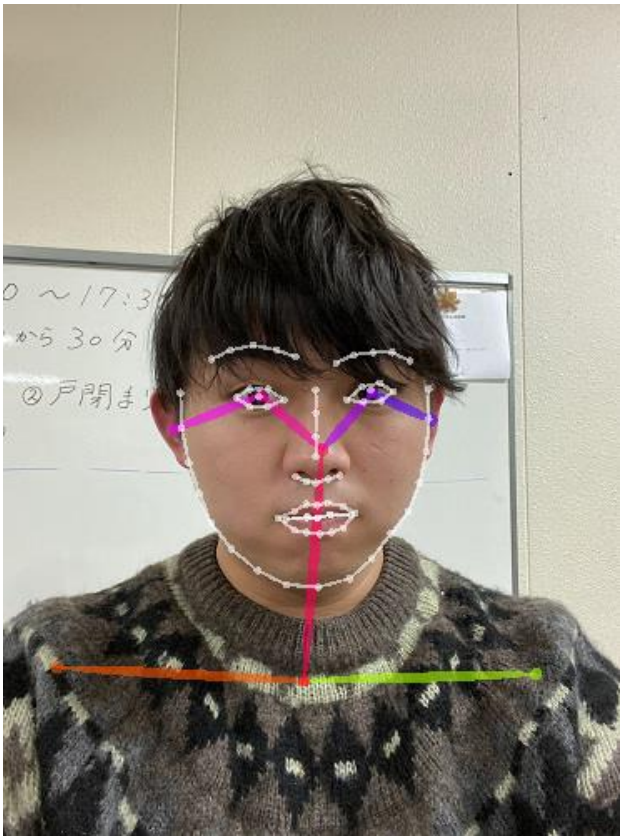

図 2.4.3 顔のキーポイントを検出した出力画像

2.5 OpenPose のアルゴリズム

OpenPose は，画像中の複数人物の 2D 姿勢を効率良く検出する手法である．画像内の人物を検出し，検出された各人物に対して姿勢推定を行うという従来の方式[5][6]とは異なり，Confidence Maps と Part Affinity Fields を用いた二つの逐次予測プロセスによって，画像内の体のパーツ位置およびパーツ間の関係性をボトムアップ的アプローチで推定している．これにより，リアルタイム性を保持しつつ高精度に姿勢情報を推定することができる．

2.5.1 Simultaneous Detection and Association

図 2.5.1 に OpenPose の構造図を示す．OpenPose はサイズ w×h[pel]のカラー画像を入力画像とし，画像内の各人物のパーツ位置座標を出力として生成する．入力画像は VGG-19[7]の最初の 10 層で初期化され，特徴マップ F として Stage1 の入力となる．各 Stage のネットワークは二つに分かれており，図 2.5.1 の Branch1 でパーツ位置を示す 2D Confidence Maps **S** を予測し，Branch2 ではパーツ間の関係性を示す 2D ベクトル場を表す Part Affinity Fields **L** を予測する．

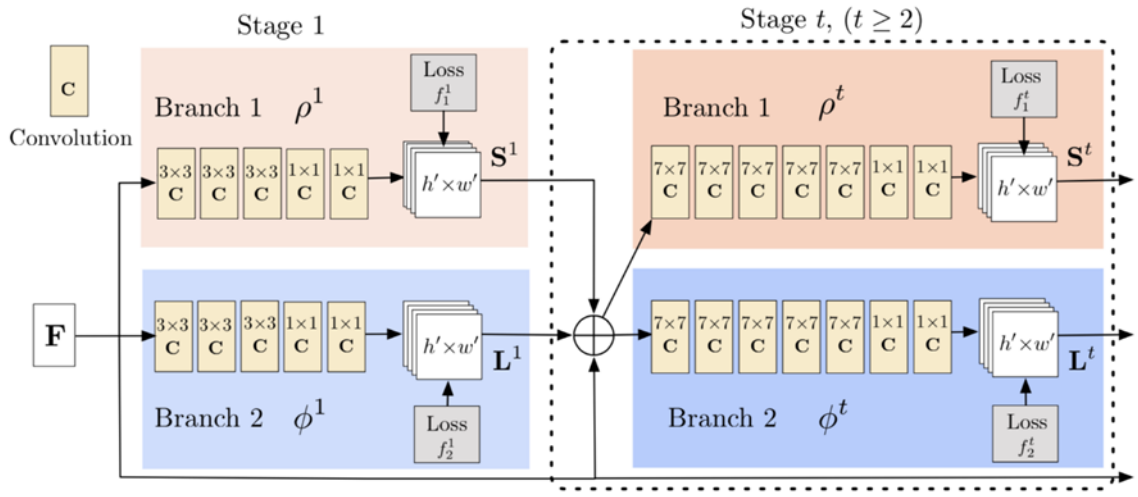

図 2.5.1　OpenPose の構造[4]

Confidence Maps $\mathbf{S}$ および Part Affinity Fields $\mathbf{L}$ はそれぞれ次の式(2.1), 式(2.2)で表される.

$$\mathbf{S} = (\boldsymbol{S_1},\ \boldsymbol{S_2},\ \ldots,\ \boldsymbol{S_J}),\quad \boldsymbol{S_j} \in \mathbb{R}^{w*h},\ j \in (1,\ \ldots,\ J) \tag{2.1}$$

$$\mathbf{L} = (\boldsymbol{L_1},\ \boldsymbol{L_2},\ \ldots,\ \boldsymbol{L_C}),\quad \boldsymbol{S_c} \in \mathbb{R}^{w*h*2},\ c \in (1,\ \ldots,\ C) \tag{2.2}$$

なお, J はパーツ数, C はパーツペア数である.

Stage1 で Confidence Maps $\boldsymbol{S^1} = \rho^1(\boldsymbol{F})$および Part Affinity fields $\boldsymbol{L^1} = \boldsymbol{\phi}^1(\boldsymbol{F})$を生成する. ここで, ρ^1および$\boldsymbol{\phi}^1$は, Stage1 における CNN の推定結果である. 各 Stage における入力は, 1 つ前の Stage での結果および元の Stage1 への入力 F を合わせたものであり, 以下の式(2.3), 式(2.4)ように表される.

$$\boldsymbol{S^t} = \rho^t(\boldsymbol{F},\ \boldsymbol{S^{t-1}},\ \boldsymbol{L^{t-1}}),\ \forall t \geq 2 \tag{2.3}$$

$$\boldsymbol{L^t} = \phi^t(\boldsymbol{F},\ \boldsymbol{S^{t-1}},\ \boldsymbol{L^{t-1}}),\ \forall t \geq 2 \tag{2.4}$$

各 Stage の終わりにそれぞれの Branch に損失関数を適用する. ステージ t の各 Branch の損失関数は以下の式で示される.

$$f_S^t = \sum_{j=1}^{J} \sum_{\boldsymbol{p}} \boldsymbol{W}(\boldsymbol{p}) \cdot \left\| \boldsymbol{S_j^t}(\boldsymbol{p}) - \boldsymbol{S_j^*}(\boldsymbol{p}) \right\|_2^2 \tag{2.5}$$

$$f_L^t = \sum_{c=1}^{C} \sum_{\boldsymbol{p}} \boldsymbol{W}(\boldsymbol{p}) \cdot \| \boldsymbol{L_c^t}(\boldsymbol{p}) - \boldsymbol{L_c^*}(\boldsymbol{p}) \|_2^2 \tag{2.6}$$

ここで, $\boldsymbol{S}^*$, $\boldsymbol{L}^*$は正解の Confidence maps および Part Affinity Fields, $\boldsymbol{W}$ はバイナリマスクであり, 場所 $\boldsymbol{p}$ にアノテーションがない場合$\boldsymbol{W}(\boldsymbol{p}) = 0$ となる.

全体の目的関数は以下の式(2.7)で表される．

$$f = \sum_{t=1}^{T}(f_S^t + f_L^t) \tag{2.7}$$

ここで，T は最大 Stage 数を表す．

2.5.2 Confidence Maps for Part Detection

式(2.7)での評価のためには，正解の Confidence Maps $\boldsymbol{S}^*$ をアノテーションされた 2D キーポイントから生成する必要がある．人物が複数人いて，各パーツが見えるとき，各パーツ j，人物 k に対応する Confidence Maps のピークを求める．まず，個々の Confidence Maps $\boldsymbol{S}^*_{j,\ k}$(正解値)を各人物 k について生成する．以下の式(2.8)で表される．

$$\boldsymbol{S}^*_{j,\ k}(\boldsymbol{p}) = \exp\left(-\frac{\left\|\boldsymbol{p}-\boldsymbol{x}_{j,\ k}\right\|_2^2}{\sigma^2}\right) \tag{2.8}$$

ネットワークから予測される形式に合わせて，全ての人に関して Map を結合する．このとき，各パーツに関する正解の Confidence Maps は以下の式(2.9)ように表される．

$$\boldsymbol{S}^*_j(\boldsymbol{p}) = \max_k \boldsymbol{S}^*_{j,\ k}(\boldsymbol{p}) \tag{2.9}$$

2.5.3 Part Affinity Fields for Part Association

任意の人数の人物に関して，検知されたパーツ同士をどのように繋げるかが問題となる．図 2.5.2 に Part Affinity Fields のベクトル値の例を示す．図 2.5.2 において$\boldsymbol{x}_{j1,\ k}$，$\boldsymbol{x}_{j2,\ k}$を人 k の四肢 c における部位 j_1，j_2の正解値としたとき，四肢 c 上のポイント $\boldsymbol{p}$ に関して，はj_1からj_2への単位ベクトルとなる．定式化す

ると以下の式(2.10)で表される.

$$\boldsymbol{L}^{*}_{\boldsymbol{c},\ \boldsymbol{k}}(\boldsymbol{p}) = \begin{cases} \boldsymbol{v} & (if\ \ p\ \ on\ \ limb\ \ c,\ \ k) \\ \mathbf{0} & (otherwise) \end{cases} \tag{2.10}$$

ネットワークから予測される形に合わせて，全ての人に関して Part Affinity Fields を合わせて四肢ごとの Fields を生成する.

$$\boldsymbol{L}^{*}_{\boldsymbol{c},\ \boldsymbol{k}}(\boldsymbol{p}) = \frac{1}{n_c(\boldsymbol{p})} \sum_k \boldsymbol{L}^{*}_{\boldsymbol{c},\ \boldsymbol{k}}(\boldsymbol{p}) \tag{2.11}$$

テスト時は, 対応する Part Affinity Fields の線積分を候補パーツの位置を結ぶ線分に 沿って計算することで，候補パーツ間の関連度を算出する.

$$E = \int_{u=1}^{u=0} \boldsymbol{L}_c\left(\boldsymbol{p}(u)\right) \cdot \frac{\boldsymbol{d_{j2}} - \boldsymbol{d_{j1}}}{\left\|\boldsymbol{d_{j2}} - \boldsymbol{d_{j1}}\right\|_2} du \tag{2.12}$$

$$\boldsymbol{p}(u) = (1-u)\boldsymbol{d_{j1}} + u\boldsymbol{d_{j2}} \tag{2.13}$$

2.5.4 Multi-Person Parsing using PAFs

各パーツに関して複数検知される場所があり，その分関連づけられる可能性のある四肢のパターンは多くなる. Affinity Fields 上の線積分計算により各候補四肢にスコアをつける．最適な組み合わせを見つける問題は，NP-Hard 問題である K 次元マッチング問題に対応する.

$$\max_{Z_c} E_c = \max_{Z_c} \sum_{m \in D_{j1}} \sum_{m \in D_{j2}} E_{mn} \cdot z^{mn}_{j1j2} \tag{2.14}$$

$$\max_{Z} E = \sum_{c=1}^{C} \max_{Z_c} \tag{2.15}$$

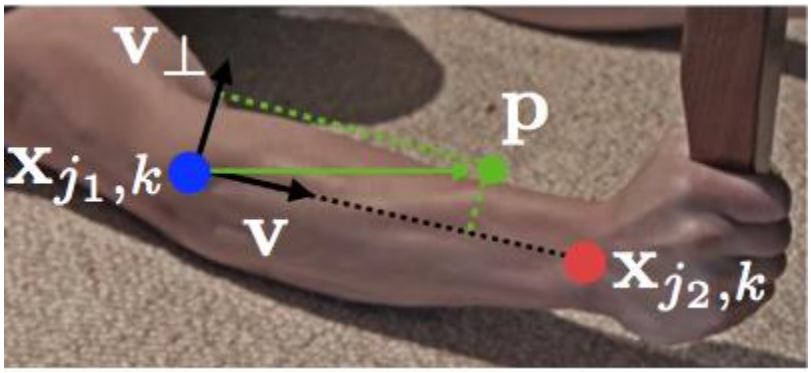

図 2.5.2　Part Affinity Fields のベクトル値例[4]

第3章 人物の姿勢推定によるシステム構築手法

本章では，OpenPose を用いて，複数人が動作している人物と複数人が静止している定点カメラ動画における動作状況を評価する手法を提案する．3.5.3 において提案手法の詳細について述べる．本研究で利用する座標データは表の 2.4.1 の番号 1~13 までの 13 点のキーポイントを処理対象とする．

3.1 OpenPose より得られた Json ファイル

得られた複数人の座標データより動作を推定する．そのため，システムの開発には OpenPose により得られた姿勢推定結果を利用する必要がある．姿勢推定結果はフレームごとに Json 形式で出力される．

Json ファイルは「JavaScript Object Notation」の略であり，JavaScript の一部をベースに作られた軽量のデータ交換フォーマットである．Python をはじめ Ruby，Perl や Java など多くのプログラミング言語で手軽に扱えるため，主に Web を介してのデータ交換に利用される[10].

OpenPose により出力された Json ファイル内に検出した人数分の二次元配列として関節点情報データが出力される．検出の対象とした身体の項目ごとにデータは格納されている．その中に各キーポイントについて x 座標，y 座標，信頼値の順に与えられる．実際に出力した Json ファイルの構造を以下の図 3.1 に示す．図 3.1 は動画像内の 1 人の人物に対して出力された Json ファイルの構造である． ここでは身体 25 点に関するキーポイントのみ検出を行っているため，pose_keypoints_2d 内にデータが x 座標, y 座標, 信頼値の順に格納されている．3.2 節以降，今節で説明した Json ファイルを用いてシステムを開発するための手法を提案する．

```
{
        "version":1.0,
        "people":[
        {
            "pose_keypoints_2d":[
             247.668, 514.465, 0.89905, 247.702, 527.27, 0.87725, 236.549,
527.306, 0
            ],
            "face_keypoints":[

            ],
            "hand_left_keypoints":[

            ],
            "hand_right_keypoints":[

            ]
        }
    ]
}
```


図 3.1 姿勢推定結果の例

3.2 本研究で作成するシステムの概要

本研究では 3.1 節において説明を行った Json ファイルを，Python3 を用いて作成するプログラムに組み込むことでシステムの構築を行う．

構築するシステムは災害時の救助活動や捜索活動の際，人が立ち入ることができない場所で監視カメラやドローンカメラなどを用いて救助者の状態や人数などを判別するシステムの実用化を最終的な目的とする．

システムの前段階として，動画像内の人物に対し，動作または静止の 2 種に動作状況を評価することが可能なシステムの作成を本研究では行う．

システムを構築するプログラムはデータ処理を行う部分と評価尺度を実装する二つの部分に分かれている．データ処理を行う部分ではデータの整形，角度の算出，外れ値の消去を行う．OpenPose はデータをフレームごとに出力するため，各フレーム間での人物同士のデータを照合しきれていないことが本システムの

開発を進めるにあたり障壁となる．そのためシステムを構築するにあたり出力されたJsonファイルよりデータ整形を先立ち行う．その後，各人物の角度データを算出し，OpenPoseが誤検出を行ってしまった際の角度データに対し，外れ値として変換処理を行うことで評価尺度を実装に用いる角度データの信頼性を高める．評価尺度を実装する部分ではデータ処理より得られた角度データより動作状況の評価を行う．

3.3節以降でこのプログラムを作成するにあたりデータを処理する手法，および動作状況を評価するための評価尺度の提案を行う

3.3 データの整形方法の提案

OpenPoseにより出力されたそれぞれのJsonファイルのデータは1フレーム前の人物のデータが格納されている順番が異なる可能性がある．そのため，複数人の座標データより人物を推定し，各人物のデータを画像内の人物の左からの順番に整形する必要がある．ある一定数以上のフレームレートの動画から得られたJsonファイルの場合，前後のフレーム間で人物の立ち位置が大きく変動することはない．これにより，首のx座標を1 フレーム前の各人物の首のx座標と比較することで人物を推定できると考える．1フレーム前の人物の首のx座標との誤差が最も小さい座標データをその人物と判断する．これを動画内のすべての人物について処理を行う．

また座標データに欠損が存在する場合は，前後フレームの座標データを参照することで処理を行う．欠損フレームの直前フレームの座標から直後フレームの座標まで一定に変化するものとして，欠損フレームの座標データを決定する．

3.4 評価手法の提案

本研究の評価手法を提案するにあたり，角度を算出し評価する手法と座標点の変化より評価する手法の２つの方法に対して検討を行った．

3.4.1　角度による検討

特定の動作を行っている場合，脚や腕の角度には人物の身長や腕の長さなどの身体的特徴にかかわらず，動作が行われる際には関節同士の角度に変化が起こると考えられる．したがって，時系列における角度の変化量を分析することにより，動作状況を判定することができる． そこで，3.3 節で示したデータ整形で得られた座標から 6 つの身体の角を求める．6 つの身体の角は表 3.4.1 に示す．この 6 つの身体の角は動きがある際に最も可動範囲が広いことが考えられる左右の首と肩と肘，肩と肘と手首，腰と膝と足首が成す 6 つの身体の角である．求められた角度に関するデータを処理することで，動作状況を判定する ．

点 A，B，C の座標が与えられたとき，∠ABC を θ° とする．BA から BC に回転する方向が左回りのならば， θ >0 と定義する．このとき， θ の大きさは式(3.1)で与えられ，回転方向は外積の符号と等しくなる．本研究では Json ファイルより得られた x 座標，y 座標を複素平面上の点として角度を算出した．

$$\theta = \cos^{-1}\left(\frac{\overrightarrow{BA}\cdot\overrightarrow{BC}}{\left|\overrightarrow{BA}\right|\left|\overrightarrow{BC}\right|}\right) \tag{3.1}$$

角度から動作を判別する方法はカメラと人間の位置関係が重要となってくる．人物が関節点同士重なり合うように姿勢をとる場合，角度を算出するのに誤差が生じてしまい評価が困難になる．しかし，この手法は救助活動を支援するシステムの開発を想定した場合に適していることが考えられる．救出活動においては人物がカメラに対し関節点同士が重なり合うように姿勢をとることはあまり考えられないためである．救助を必要とする場合，カメラに対し身体を向けることが考えられる上に，もし体が動かせないような場合には元より角度変化が生じないため影響は出ない．そこで本研究では，身体的特徴を考慮する必要のない角度変化の極値による評価手法を採用する．

また，座標データに欠損が存在する場合，もしくは関節点が重なり合ってしま

い角度の算出が困難な場合，前後フレームの座標データを参照することで処理を行う．欠損フレームの直前フレームの座標から直後フレームの座標まで一定に変化するものとして，欠損フレームの座標データを決定する．

表 3.4.1　6 つの身体の角の番号

i	キーポイントの組み合わせ (A，B，C)	身体の角のキーポイントの組み合わせ
1	(1，2，3)	右の首と肩と肘
2	(1，5，6)	左の首と肩と肘
3	(2，3，4)	右の肩と肘と手首
4	(5，6，7)	左の肩と肘と手首
5	(8，9，10)	右の腰と膝と足首
6	(11，12，13)	左の腰と膝と足首

3.4.2　座標による評価

角度とは違い，身長や腕の長さは人物によって座標の増減量が異なる．複数の人物について評価を行う場合，個人の身体的特徴に応じて判定基準が異なってしまうことが考えられる．そのため，座標の増減量から正しく動作の状況を評価することはできない．しかし，複数人がある同じ動作をする場合には，身長や腕の長さに関わらず同時に時系列における座標変化の極値が生じる．その極値はある動作から次の動作に変わる瞬間であるため，そのタイミングで動作の評価が可能となる．そのため，座標による評価方法はスポーツに対して評価を行う際には有効な方法であると考えられる．本研究では，各フレームで移り変わる人物を推定する目的でのみこの手法を利用する．この手法については前段の 3.3 節において説明している．

3.5 外れ値の処理方法の提案

OpenPose により検出した各キーポイントは必ずしも正確であるとは限らない. そのため, 動画内で誤検出が生じた際に算出した角度は採点に影響を及ぼす. そのため本研究では外れ値に対しては, 値を変換する処理を行う. ある一定数以上のフレームレートの動画から得られた Json ファイルの場合, 前後のフレーム間で人物の関節間の角度が大きく変動することはないと考えられる. そのため, 算出した角度を直前フレーム前に算出した角度と比較し, 変化量が 50° 以上の場合, 算出した角度を外れ値とみなしデータとして扱わないこととする. このフレームの角度データは前後フレームの角度データを参照することで処理を行う. 直前フレームの座標から直後フレームの座標まで一定に変化するものとして, 角度データを決定する.

3.6 評価尺度の提案

3.6.1 得点の算出及び評価

各人物の角度の時系列データより動作状況を評価する二つの手法を提案する. ここで, 前述した 6 つの身体の角(i=1~6)について人物ごとに角度の時系列データを算出する. 6 つの身体の角 (i=1~6)についてそれぞれ全フレームでの平均値a_iを算出し, 誤差を時系列データとして求める. 誤差の平均値A_iを最終的に得点t_i (10 点満点)に対数関数を用いて換算し, 任意に設定した重みW_iを用いて任意にそれぞれの 6 つの身体の角に対する点数の比率を決定する. しきい値 K を設定し得点の合計を 100 点満点に換算した得点 T が K 以上か以下かで動作状況を判定する. 換算に対数関数を使用する理由としては, 角度変化がある一定数以上ある場合には大きい場合には確実に動作を行っていると判断することができるが, 角度変化が微小な場合, 動作を行っていると判断するためには細かく得点を算出する必要があるからである. 本手法ではこのような得点算出をする場合に対数関数を使用するのが適していると考え, 対数関数を使用する手法を採用した.

t_i，T を求める計算式は以下のようにして求める．

$$\begin{cases} t_i = 0 \ \ (A_i = 0) \\ t_i = 10 \times \left(\frac{\log_e(A_i+1)}{\log_e(360+1)}\right) \ \ (\log_e A_i \geq 0) \end{cases} \tag{3.2}$$

$$\mathrm{T} = 100 \times \frac{\sum_{i=1}^{6}(t_i \times W_i)}{60 \times \sum_{i=1}^{6} W_i} \tag{3.3}$$

3.6.2　しきい値の設定

しきい値 K を設定する場合，カメラを人物に対して向ける角度に応じてしきい値を変える必要がある．人物とカメラを水平にして撮影する場合，角度変化は 0~360°の間で変化することが考えられる．しかし，カメラに高さが生じる場合，OpenPose により出力される関節点座標は二次元座標であるため，0~360°よりも狭い範囲で角度変化が生じることになる．このため，しきい値を地面からの高さに比例してしきい値を低く設定する必要がある．実際に生じる状況を図 3.6.1 に示す．ここで，角度変化の範囲は水平での撮影に比べて θ°狭くなると考えられる．

このため，しきい値の補正は θ を考慮して行わなければならない．地面からの高さを考慮しないしきい値を k とし，実際に評価に使用するしきい値 K は以下の計算式(3.4)で補正を行うものとする．

$$\mathrm{K} = \mathrm{k} \times \cos\theta \tag{3.4}$$

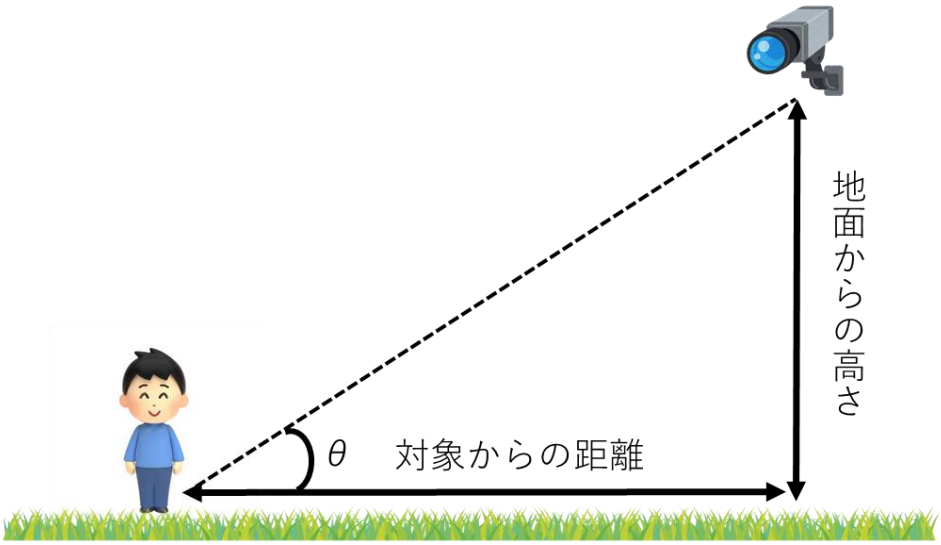

図 3.6.1 実際に想定される状況

3.7 本システムのフローチャート

本システムのフローチャートを図 3.7.1 に示す.

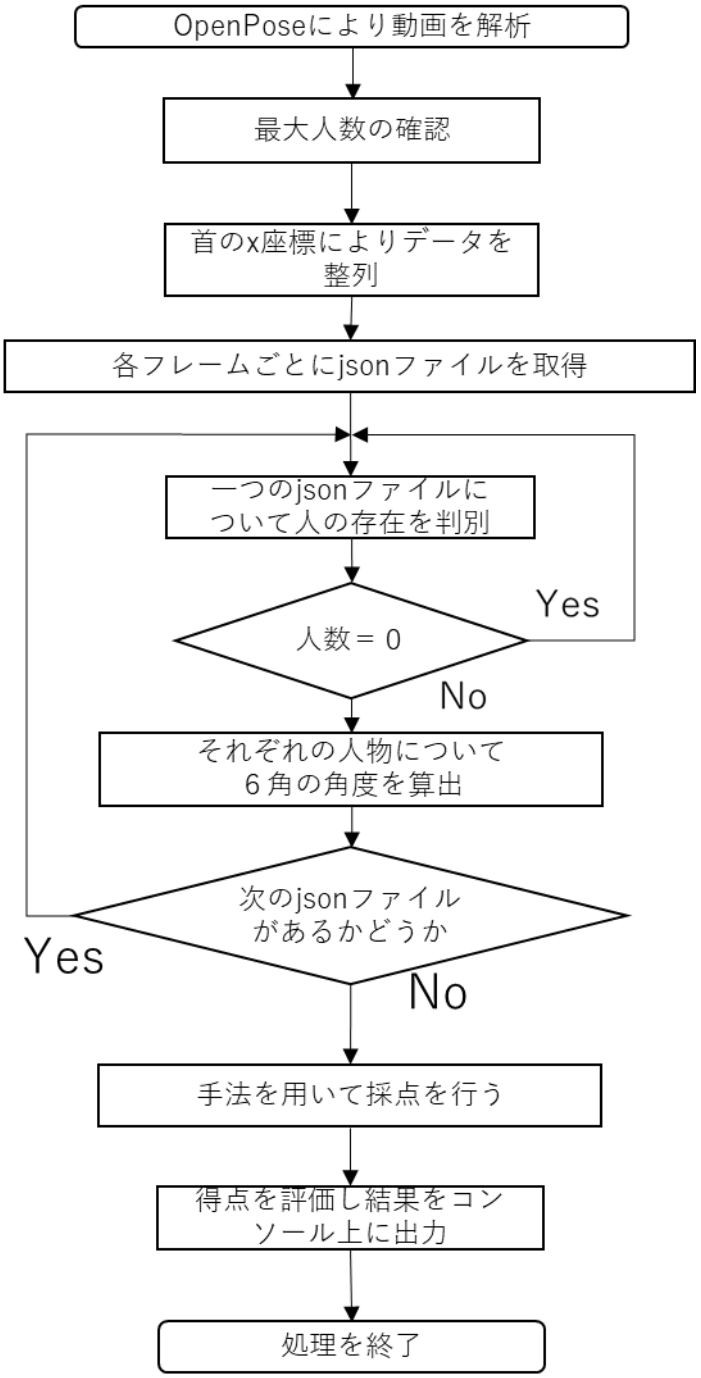

図 3.7.1　本システムのフローチャート

第4章 実験環境及び結果

4.1 実験に使用する動画

本実験に使用する高度を変えた 3 種類の動画を撮影した．本研究では，4 人の人物が特定の動作を行う 2 種類の動画を使用する．しきい値の有効性を示すため 3 種類の動画は水平方向での撮影と地面からの高さを変えて撮影した．人物は移動せず 3 人が手を振る動作を行い，残りの 1 人は一切の動作を行わない動画を使用する．3 種類の動画において同様の動作を行った．これは実験結果における動画間での比較を可能な限り容易にするためである．使用する動画は撮影する高度を変えて検討を行う．これは実際に救助活動への支援を想定した場合，上空から撮影するカメラの高度により結果が異なる場合が考えられるためである．人物はどちらも左から A，B，C，D とする．いずれの動画も人物 A は一切の動作を行わない．それぞれの動画に対し，OpenPose を使用し解析を行った動画を No.1~3 とし，それぞれの動画の詳細を表 4.1.1 に示す．

表 4.1.1 No.1~3 までの動画の詳細

No	秒数	フレームレート (フレーム/秒)	地面からカメラまでの距離	地面からカメラまでの高さ
1	10 秒	65.20fps	4.00m	0.00m
2	10 秒	31.00fps	4.00m	5.52m
3	10 秒	30.05fps	4.00m	9.43m

4.2 実験環境

使用器具を 4.2.1，動作環境を表 4.2.2 と実験に使用する動画を撮影したスマートフォンカメラの詳細を表 4.2.3，開発環境を表 4.2.4 に示す．

表 4.2.1 使用器具

名称・型式	製造元	型番
デスクトップ PC OMEN by HP 880-000jp	日本 HP	YON68AA#ABJ
撮影に使用したスマートフォン	Apple	MWC72J/A

表 4.2.2 動作環境(デスクトップ PC)

製造元	日本 HP
モデル	OMEN by HP 880-000jp パフォーマンスモデル
CPU	Intel(R)Core (TM)i7-7700 CPU@ 3.60GHz　3.60GHz
GPU	NVIDIA　GeForce GTX 1070
実装メモリ	32.0GB
システムの種類	64 ビットオペレーティングシステム
OS	Windows 10 Pro

表 4.2.3 撮影環境(スマートフォンカメラ)

製造元	Apple
製品名	iPhone 11 Pro
型番	MWC72J/A
解像度	2436×1125(4K)
フレームレート	60fps

表 4.2.4 開発環境

プラットフォーム	Anaconda3
開発言語	Python3
ライブラリ	OpenPose v1.5.1

4.3 OpenPose により出力された骨格モデル

No.1~2 までのそれぞれの動画の第 1 フレームの骨格モデルを図 4.3.1 と図 4.3.2，図 4.3.3 に示す．なお，No.3 の動画については OpenPose により解析を行った際，人物を認識せず，背景を認識してしまいシステムに利用することが不可能であったため，4.4 節以降は No.1~2 の動画を実験結果として検討を行う．No.3 の動画についての検討は 4.5 節及び第 5 章で行うものとする．

図 4.3.1 動画 No.1 の第 1 フレームの骨格モデル

図 4.3.2 動画 No. 2 の第 1 フレームの骨格モデル

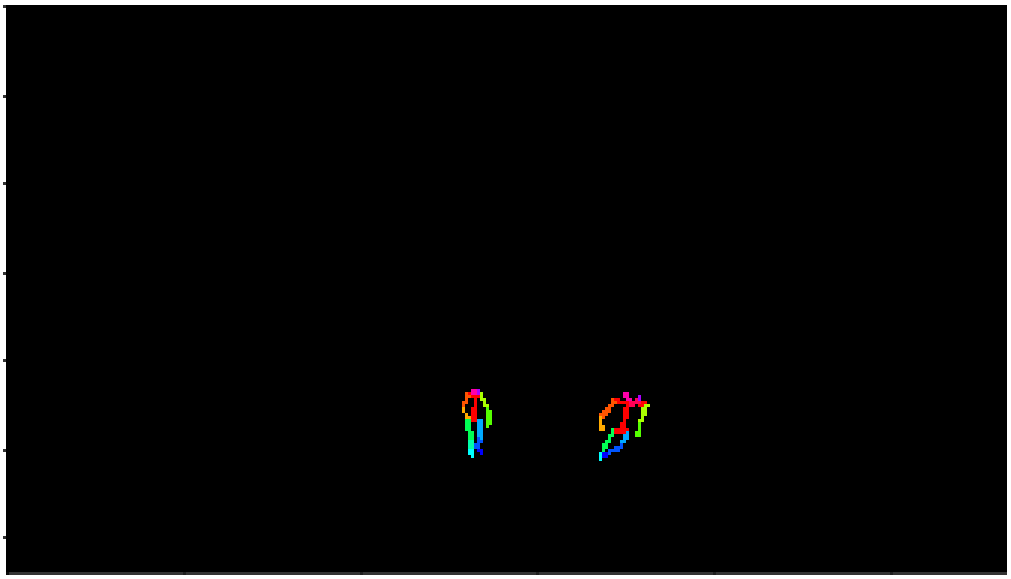

図 4.3.3 動画 No.3 の第 1 フレームの骨格モデル

4.4 実験結果

4.4.1 実験条件および使用したパラメータ

実験では，実験条件として重みW_i ，しきい値 K をそれぞれ表 4.4.1，表 4.4.2 に示すパラメータの値を使用した．本実験では 4 人とも下半身は動作を行って

いないため，重みW_5，W_6については 0.0 として採点の方を行う．

表 4.4.1 実験に使用した重みW_i

No	W_1	W_2	W_3	W_4	W_5	W_6
1~2	1.5	1.5	1.5	1.5	0.0	0.0

表 4.4.2 実験に使用したしきい値 K

No	しきい値 k	θ	cos θ	しきい値 $\boldsymbol{K}$
1	50	0.00	1.00	50.00
2	50	54.07	0.58	29.06

4.4.2　データ整列の有効性の確認

3.3 節の手法に対する評価を行った．3.3 節の手法を実装せず出力された json ファイルの座標データのうち，No.1 の第 1 フレームから最終フレームまでの各人物の首の x 座標の時系列における変化の様子を図 4.4.1 に示す． 3.3 節の手法を用いてデータ整形を行い得られた座標データのうち，No.1 と No.2 の第 1 フレームから最終フレームまでの各人物の首の x 座標の時系列における変化の様子をそれぞれ図 4.4.2，図 4.4.3 に示す．この時系列データを出力することで 3.3 節の有効性を確認することができる．

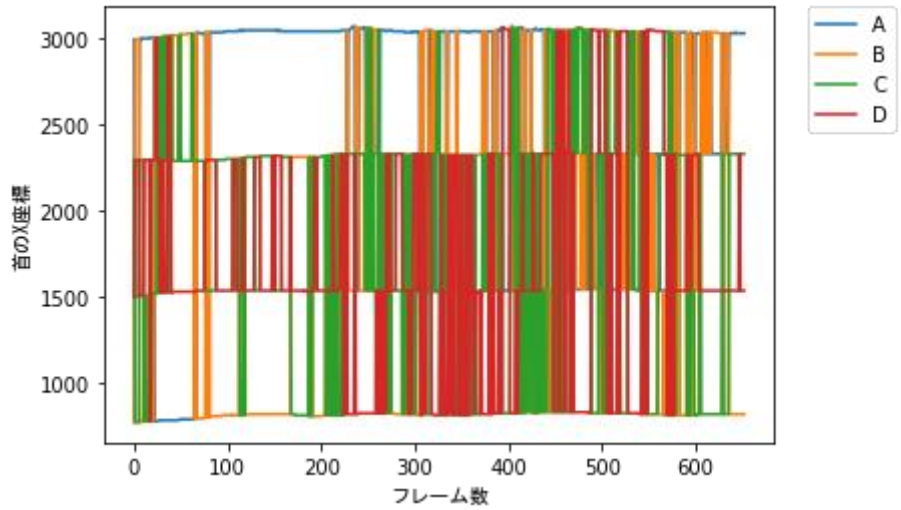

図 4.4.1　No.1 の各人物の首の x 座標の時系列における変化(実装前)

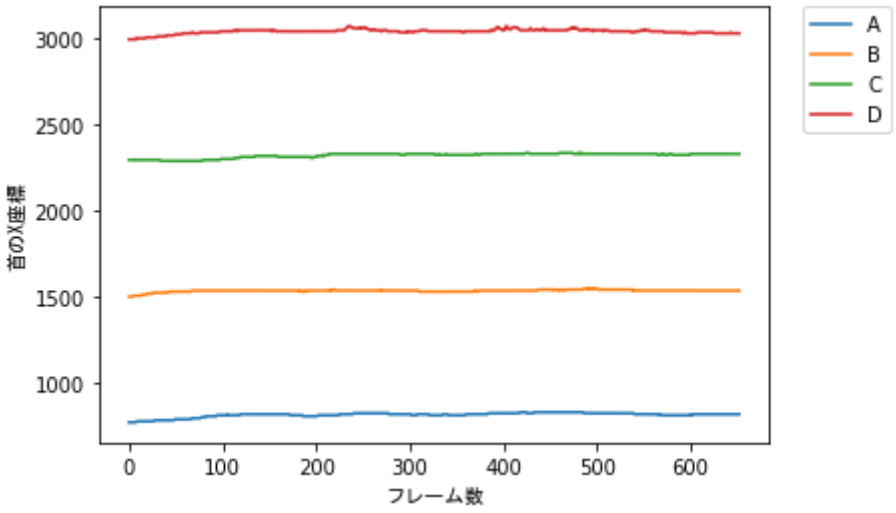

図 4.4.2　No.1 の各人物の首の x 座標の時系列における変化(実装後)

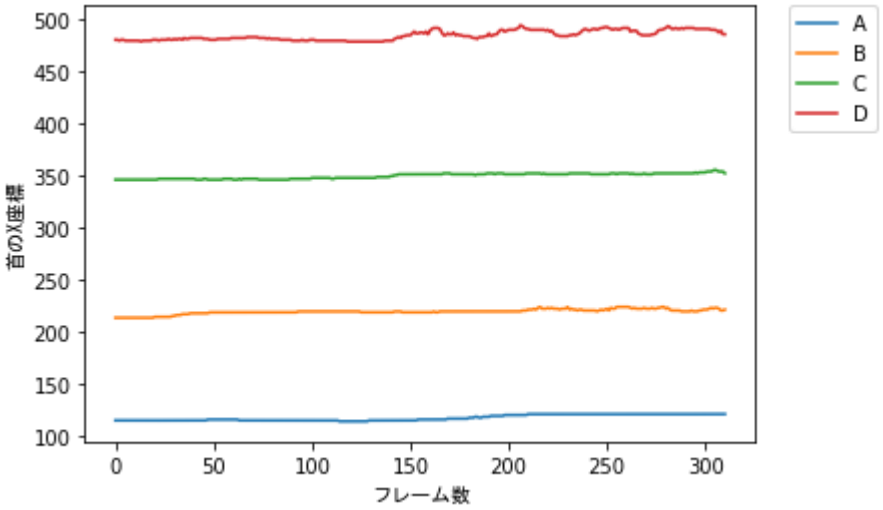

図 4.4.3　No.2 の各人物の首の x 座標の時系列における変化(実装後)

得られた時系列データより首の x 座標がフレームごとに大きく変化していないこと分かる．人物同士が交差しない動画の中で，フレームごとに整列した座標データが左から順に格納されていない場合，時系列データに交わりが生じることが考えられる．3.3 節の提案手法の実装を行わなかった図 4.4.1 に対してはこのような結果が得られた．図 4.4.2，図 4.4.3 について交点が一つも生じてないことより，データが 4 人に対し，動画内の左から順番に格納されていることがわかる．

4.4.3　各人物の角度変化

動画 No.1 の第 1 フレームから最終フレームまでの 4 人について 6 つの身体の角(i=1~6)の時系列における角度変化の様子をそれぞれ図 4.4.4，図 4.4.5，図 4.4.6，図 4.4.7 に示す．

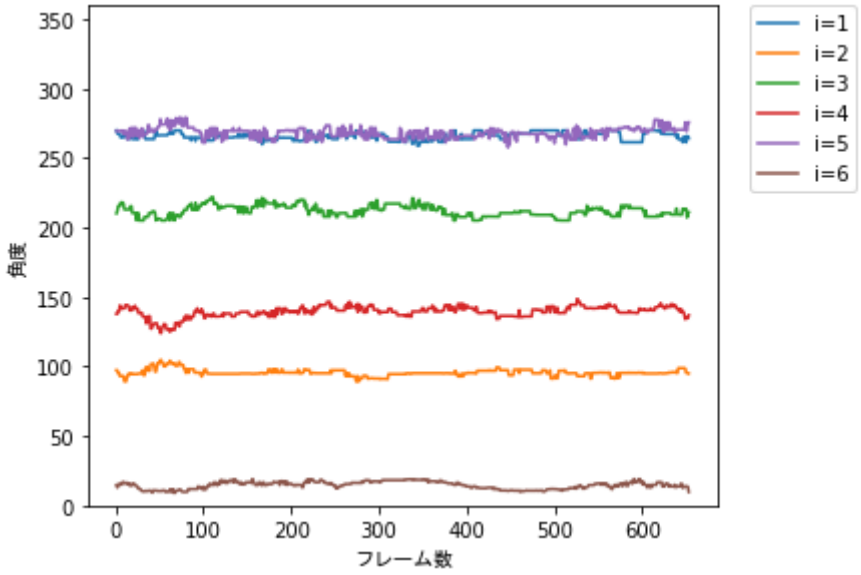

図 4.4.4 動画 No.1 の人物 A の 6 つの身体の角の時系列における角度変化

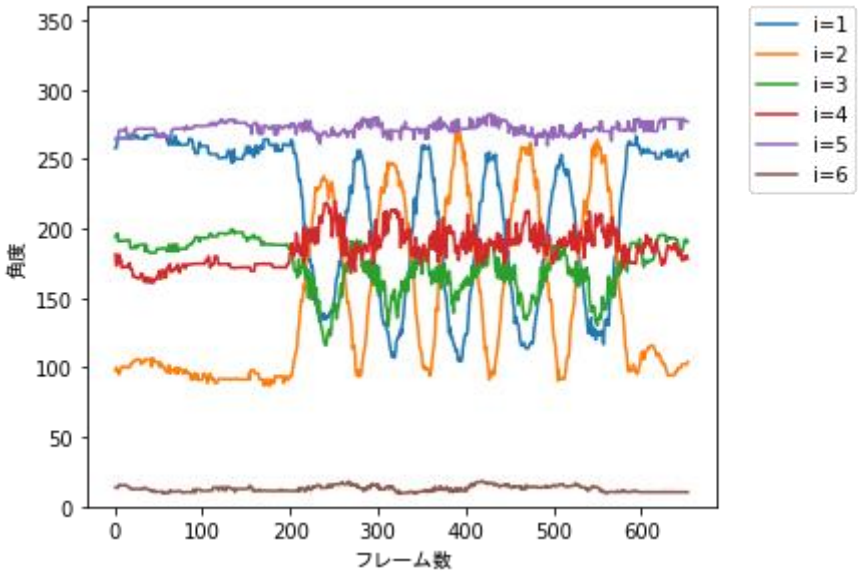

図 4.4.5 動画 No.1 の人物 B の 6 つの身体の角の時系列における角度変化

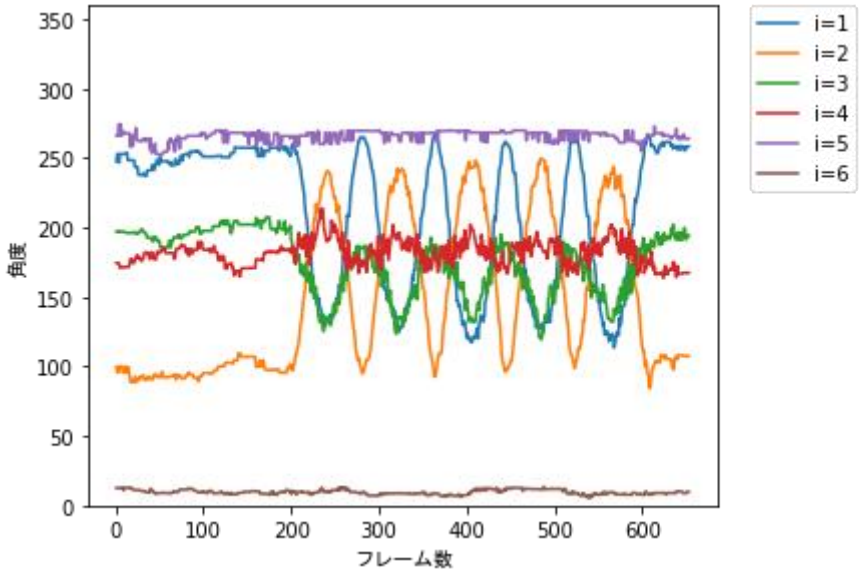

図 4.4.6 動画 No.1 の人物 C の 6 つの身体の角の時系列における角度変化

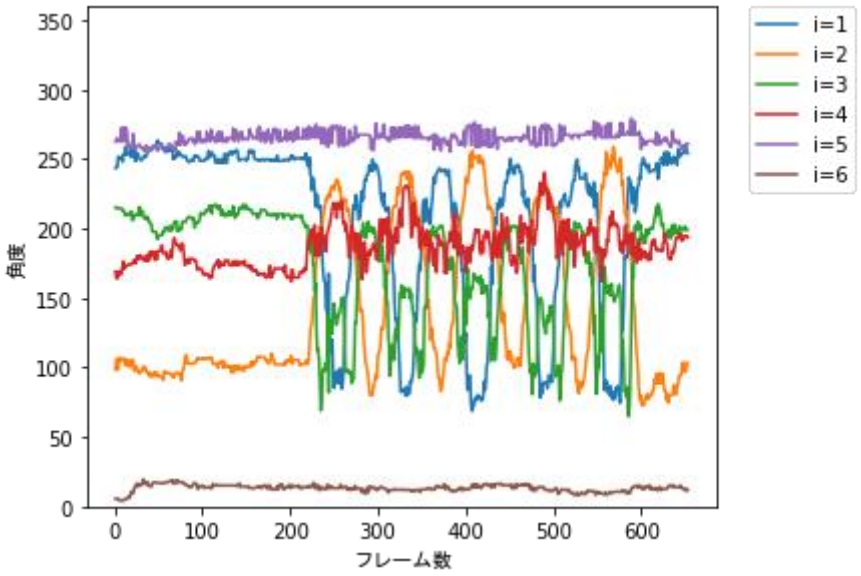

図 4.4.7 動画 No.1 の人物 D の 6 つの身体の角の時系列における角度変化

動画 No. 2 の第 1 フレームから最終フレームまでの 4 人について 6 つの身体の角(i=1~6)の時系列における角度変化の様子をそれぞれ図 4.4.8，図 4.4.9，図 4.4.10，図 4.4.11 に示す.

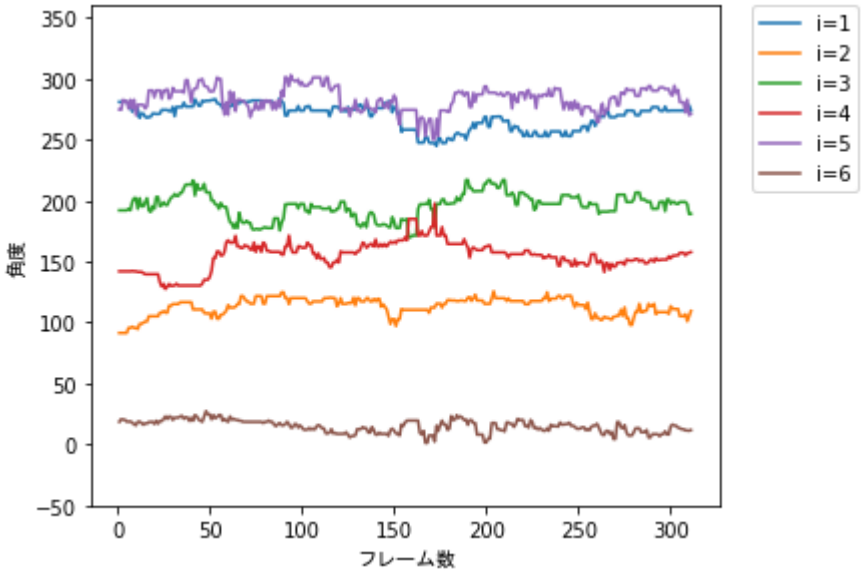

図 4.4.8 動画 No.2 の人物 A の 6 つの身体の角の時系列における角度変化

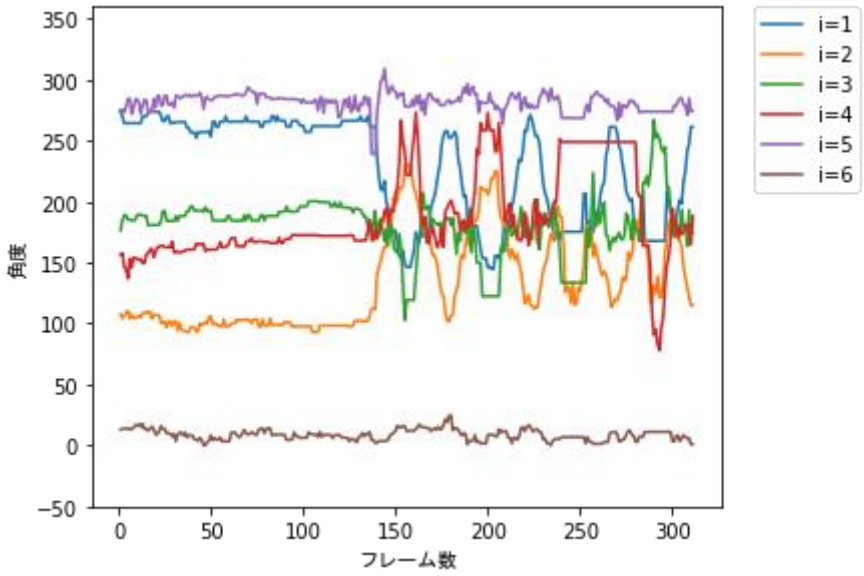

図 4.4.9 動画 No.2 の人物 B の 6 つの身体の角の時系列における角度変化

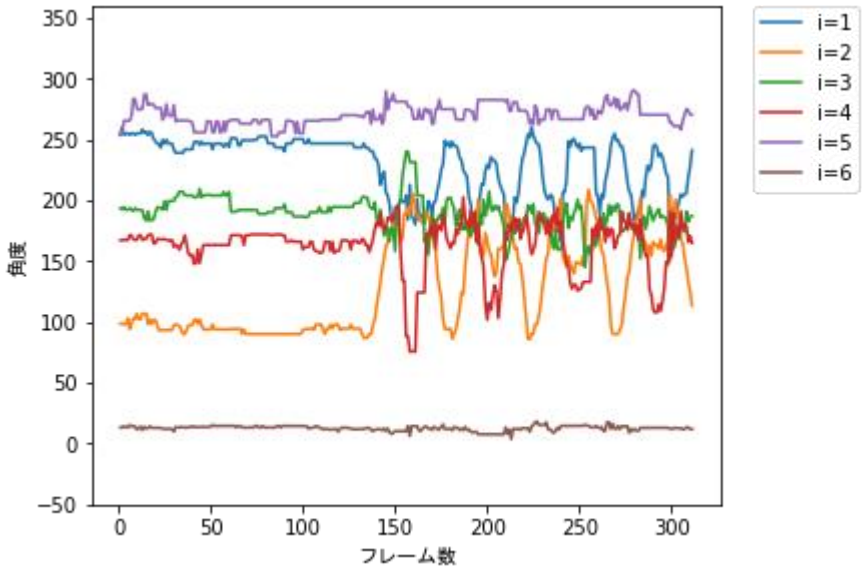

図 4.4.10 動画 No.2 の人物 B の 6 つの身体の角の時系列における角度変化

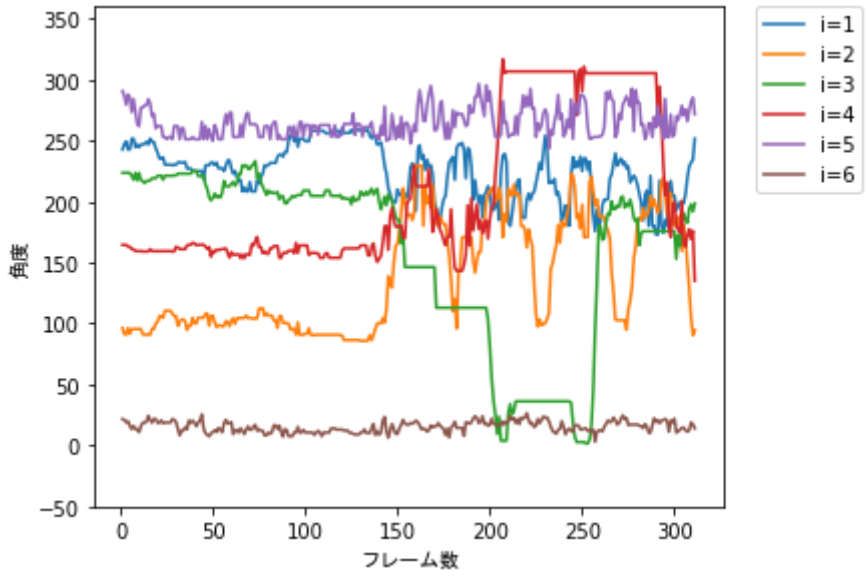

図 4.4.11 動画 No.2 の人物 B の 6 つの身体の角の時系列における角度変化

4.4.4 3.6 節の手法による採点結果

3.6 節で述べた手法による実験結果を示す.

それぞれの動画の第 1 フレームから最終フレームまでの 6 つの身体の角の角

度の平均値a_iを表 4.4.3 と表 4.4.4 に示す.

表 4.4.3 動画 No.1 の 6 つの身体の角の角度平均a_i

	人物 A	人物 B	人物 C	人物 D
a_1	265.80	213.16	211.85	208.08
a_2	95.47	147.41	148.32	144.20
a_3	211.99	173.71	174.95	174.25
a_4	139.34	185.18	181.60	187.22
a_5	268.43	272.41	266.05	265.45
a_6	14.36	12.48	9.38	13.04

表 4.4.4 動画 No. 2 の 6 つの身体の角の角度平均a_i

	人物 A	人物 B	人物 C	人物 D
a_1	269.17	227.62	229.94	224.85
a_2	112.90	132.96	127.39	137.52
a_3	194.74	141.50	188.88	159.29
a_4	155.01	181.16	162.60	207.53
a_5	283.10	281.42	270.21	265.16
a_6	14.69	8.52	12.49	15.16

それぞれの動画について第 1 フレームから最終フレームまでの平均値との誤差a_iの最大値d_iを最終的に換算した得点t_i，及び重み付けを行いそれぞれの人について算出したT, 動作状況のしきい値Kを用いた判定結果を表 4.4.5 と表 4.4.6 に示す.

表 4.4.5 動画 No.1 の得点t_i，T と判定結果

	人物 A	人物 B	人物 C	人物 D
t_1	2.01	6.60	6.53	6.62
t_2	1.60	6.78	6.69	6.72
t_3	2.45	4.73	5.12	6.05
t_4	2.28	4.13	3.55	4.33
t_5	2.33	2.44	2.37	2.63
t_6	2.00	1.77	1.45	1.80
T	20.84	55.59	54.71	59.29
判定結果	静止	動作	動作	動作

表 4.4.6 動画 No. 2 の得点t_i，T と判定結果

	人物 A	人物 B	人物 C	人物 D
t_1	3.84	6.25	5.22	5.04
t_2	3.28	5.98	6.10	6.41
t_3	3.83	6.80	4.05	6.95
t_4	3.91	5.37	4.69	6.88
t_5	3.75	3.21	3.41	4.04
t_6	2.83	2.56	1.56	2.65
T	37.11	61.00	50.16	63.19
判定結果	動作	動作	動作	動作

4.5 考察

4.5.1 動画 No.1 について

人物 A は常に動作を行っていないためフレーム数が変化しても角度データの変化は起こらないことが推定される．図 4.4.4 より 6 つの身体の角全てにおいて，フレームの変化にかかわらず，角度データに変動が生じていないことが読み取れる．正しく人物 A の角度変化を計測できていることがわかる．

人物 B, C, D は腕を振る動作を行っているため，上半身に関係する 4 角（i=1, 2, 3, 4）はフレーム数の変化に伴い波状に角度データが変化すると推定される．下半身に関係する 2 角（i=5，6）についてはフレーム数が変化しても角度データの変化は起こらないことが推定される．図 4.4.5，図 4.4.6，図 4.4.7 より動作を行っている上半身に関係する 4 角（i=1，2，3，4）についてはフレームの変化に伴い，角度データが変化していることが読み取れる．動作を行っていない下半身に関係する 2 角(i=5，6)については，フレームの変化にかかわらず，角度データに変動が生じていないことが読み取れる．

以上より動画 No.1 については角度変化を正確にとらえることができていると考えられる．

4.5.2 動画 No. 2 について

推定される角度変化は動画 No.1 の時と同様に角度データの変化が起こると考えられる．しかし，動画 No. 2 おいては OpenPose によりキーポイントを取得した際，誤検出されることが多かった．左右でキーポイントが逆転することが誤検出の大きな要因である．実際に生じた誤検出の際の動画の骨格モデルを図 4.5.1 に示す．左から 3 番目の人物 C について左足と右足に関する検出したキーポイントが逆移転していることが読み取れる．そのため，3.5 節で述べた手法を用いて外れ値を消去する処理を行うことが多かったため，データを推測することで補う部分がある．直線状に角度データが変化している部分がみられるのはこのような要因が考えられる．

図 4.4.8 より人物 A については 6 つの身体の角全てにおいて，フレーム数の変化にかかわらず，多少の角度データの変化はあるものの角度データに大きく変動が生じていないことが読み取れる．多少の角度データの変化が起こっていっる理由として外れ値を完全に認識しきれていないことが理由として挙げられる．

図 4.4.9，図 4.4.10，図 4.4.11 より人物 B，C，D については動作を行っている上半身に関係する 4 角（i=1，2，3，4）はフレーム数の変化に伴い，角度データが変化していることが読み取れる．動作を行っていない下半身に関係する 2 角(i=5，6)については，フレームの変化にかかわらず，角度データに変動が生じていないことが読み取れる．

しかし，No.1 の時と同様にある程度正確な角度データをとることができていると考えられる．

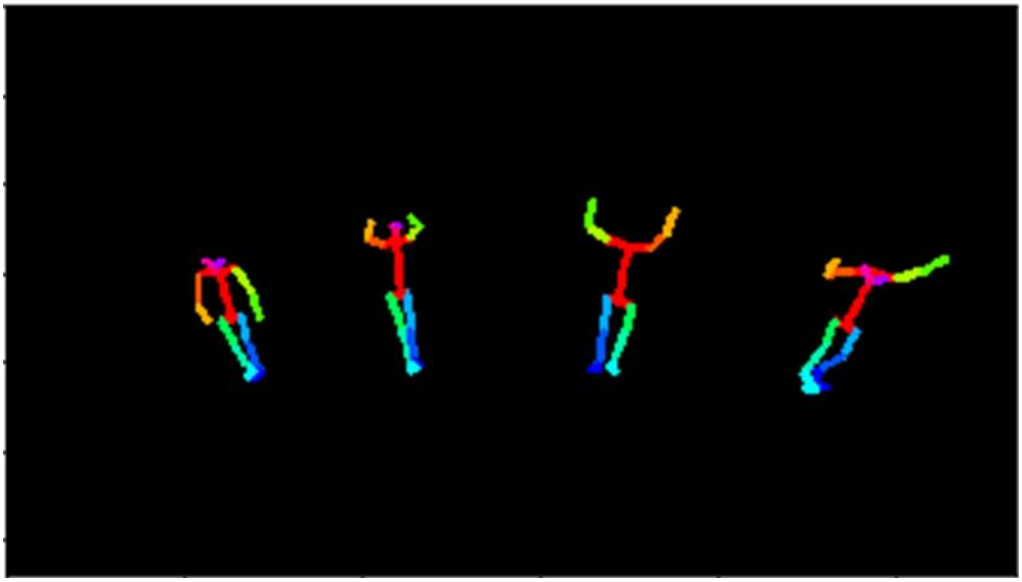

図 4.5.1 誤検出の際の動画の骨格モデル

4.5.3　動画 No.3 について

動画 No.3 について OpenPose により人物の検出が行えなかった理由として，人物からのカメラまでの距離が離れていたことが要因として考えられる．また，動画像内で背景に写っている影や自転車などを人として認識しているフレームもあった. OpenPose では人数の多い動画や写り込む人物の大きさが多い映像に対しては誤検出が多く課題が残る．実際にシステムの実用化を検討するにはこのような課題を解決しなければならない.

4.5.4　採点結果について

表 4.4.5 と表 4.4.6 より，動画 No.1，動画 No.2 のどちらについても動作を行っていない人物 A の得点 T が他の動作を行っている 3 人の得点 T と比べて大きく下回ったことが確認できる. そのため，評価尺度は適切な手法であるといえる. しかし，動画 No.2 の人物 A について静止しているにもかかわらず，動作と判定を行った. この理由として挙げられるのは，カメラに対して人物が動作を行う際にカメラを向いて動作を行ったためだと推測される．そのため，しきい値 K をカメラの地面からの高さに応じて変更させる必要はなかったと考えられる．実際に No.1 の時と同様のしきい値 K を適用した場合，人物 A は静止と判断される. そのため今後，救助活動などへ応用するシステム実装を考えると救助対象人物がカメラを見つけた際にはカメラを向いて動作を行うことが考えられるため，カメラと地面の高さを考慮し，しきい値 K を設定する必要はない．しきい値 K の設定については今後システムの実装に向けて検討していく必要があると考えられる.

第5章 結論

近年, 姿勢推定技術の一つとして最も注目を集めているのが, カーネギーメロン大学の ZheCao らによって CVPR2017 で発表された「Realtime Multi-Person 2D Pose Estimation using Part AffinityFields」が実装されたライブラリである OpenPose である．この姿勢推定技術の発達により，関節点取得による人物の姿勢推定や動作の判別を行うことで，作業支援や情報提示といった形で医療・スポーツ・災害・障害者支援・犯罪防止・エンターテインメントなどの様々な分野への応用が期待されている．

本研究室では今年度より OpenPose を用いたシステムの開発に関する研究を行っている本論文では災害時の救助活動を円滑に進めるため監視カメラやドローンなどに設置されたネットワークカメラより救助者を発見する活動支援に役立てるシステムの開発を見据え, OpenPose を用いたシステムの開発に必要な環境構築と Python を用いて検出結果のデータ処理を行うプログラムの開発を行った．

第 2 章では，OpenPose の利点や課題，取得されるキーポイントの情報，OpenPose のアルゴリズムについての説明を行った．

第 3 章では，姿勢推定技術の一つである OpenPose を使用して作成する災害時救助活動支援を目的としたシステムの構想，および動作状況を評価する事を目的とする手法を提案した．

第 4 章では，特定の動作を行った動画を解析し，得られた関節点情報に対し提案手法を実装したプログラムを適用し得られた結果を示した．初めに実験環境を示した．次に 3.3 節で提案した手法が有効的に処理を行っているかどうか確認をした．二種類の動画に対して OpenPose により得られたデータが各人物ごとに並び変えることができているかどうか時系列データを用いて確認した．次に算出した角度データの変化を時系列データとして出力することで，実際の人物の動作に合わせて角度データが変化しているか確認した．また，3.5 節で提案した手法による外れ値の処理が行われていることも確認した．次に 3.6 節で

提案した評価尺度が有効であるか実際に得られた得点などのデータより検証を行った. 結果, 動作を行っていない人物については動作を行っている人物よりも低い得点が得られることが確認された. しかし, 評価を行う手法で提案したしきい値の設定方法は正しく動作を評価することができず，しきい値のカメラの高さによる補正方法は有効性がないことが分かった．人物はカメラを向いて動作を行うのか動作している人物を上空からカメラで撮影するのかなどの条件によりしきい値設定を変えなければならないことが推測される. また, 評価尺度にもより良い計算手法を提案し，今後改良していくことが可能であると考えられる.

実際に本システムを災害時の救助活動の支援に役立てるためには，本研究のシステム開発段階では OpenPose をシステム自体に組み込んではいない．そのため, 今後実用化に向けて, ネットワークカメラを利用したリアルタイム処理を行えるようにする必要がある．そこで取得した関節点情報に対し本システムを適用することで, 実用化が可能であると考える. また, 動画像内の人数が変化する場合や人同士の交差が激しい場合には本システムは利用できないと考えられるため，ソースコードの改良が今後必要である.

本研究の結果により，本研究室での今後の OpenPose を用いた災害時捜索活動支援システムの開発に役立てることができると考えられる.

謝辞

本研究の場を与えていただき，御指導と助言をしていただいた西本昌彦教授に深く感謝の意を表します.
そして，多岐にわたり的確な助言を頂いた緒方公一准教授に厚く御礼を申し上げます. また，日頃，励ましの言葉をかけていただく田邉将之助教に深く感謝の意を表します.

実験に御協力を頂いた学部 4 年の植山真太朗氏，川畑拓也氏，高取晃司氏，山口翔平氏に深く感致します.
また， 研究及び学生生活を共にし，様々なことで助言いただいた修士課程 2 年の入江奎輔氏，佐田実季氏，中谷俊輔氏，松永尚基氏，村田憲亮氏，山本健人氏，修士課程 1 年，有村啓佑氏，加藤和太氏，岸川真氏，松山翔氏，学部 4 年の伊東麻沙美氏，植山真太朗氏，川畑拓也氏，佐藤瞭氏，高取晃司氏，廣嶋あみ氏，森拓己氏，山口翔平氏，山崎飛稀氏に感謝致します.

本研究の一部は科学研究費補助金（JP17K06464）の援助によることを記し，謝意を表する.

参考文献

[1] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, “Realtime multi-person 2d pose estimation using part affinity fields,” in *CVPR*, 2017

[2]“人物の姿勢推定(1) –OpenPose,” https://news.mynavi.jp/article/cv_future-47/，参照Jan．17，2020.

[3]“OpenPose試してみた．～ディープラーニングで人のポーズを解析,” https://ledge.ai/OpenPose/，参照Jan．18，2020.

[4]“GitHub, inc. CMU-Perceptual-Computing-Lab/OpenPose,” https://github.com/CMU-Perceptual-Computing-Lab/OpenPose ，24，April，2017

[5]“動画や写真からボーンが検出できる OpenPose を試してみた,” http://hackist.jp/?p=8285，参照Jan．20，2020.

[6]“Windows への OpenPose導入手順【2018/12/30追記】,” https://qiita.com/miu200521358/items/539aaa63f16869191508, 参照Jan．20，2020.

[7] Shih-En Wei，Varun Ramakrishna，Takeo Kanade，and Yaser Sheikh，”Convolutional pose machines. *CoRR*,” Vol. abs/1602.00134，2016.

[8] Adrian Bulat and Georgios Tzimiropoulos，”Human pose estimation via convolutional part heatmap regression. *CoRR,* ” Vol. abs/1609.01743,

2016.

[9] Karen Simonyan and Andrew Zisserman.,　”Very deep convolutional networks for large-scale image recognition. CoRR,　” Vol. abs/1409.1556, 2014.

[10] 柴田淳, “みんなのPython 第4版, ”　SBクリエイティブ株式会社, p.405, 2019.