ヒューマンピクトグラミング
－人物動画からのアニメーション人型ピクトグラムの自動生成－

高橋 将樹† 伊藤 一成†

†青山学院大学社会情報学部 〒252- 5258 神奈川県相模原市中央区淵野辺 5-10-1

E-mail: †a8115119@aoyama.jp, kaz@si.aoyama.ac.jp

あらまし 人間の身体動作の入力からピクトグラミングのコードを出力し，実行するプロセスをヒューマンピクトグラムと呼称している．これを実現するために，人間の姿勢の情報を推定するモジュールを実装し，人型ピクトグラムを用いたコンテンツ作成環境「ピクトグラミング」と連携することで，動画からピクトグラムアニメーションを生成するシステムを構築したので報告する．ピクトグラミングは，初学者向けプログラミング学習環境の用途でも広く用いられており，これにより身体を実際に動かして学ぶプログラミング学習などの応用も考えられる．

キーワード 人型ピクトグラム，教育用プログラミング言語，ピクトグラミング

1. はじめに

ピクトグラムとは，日本語で絵記号，図記号と呼ばれるグラフィックシンボルであり，意味するものの形状を使ってその意味概念を理解させる記号である[1]．我々の研究グループでは，人型ピクトグラムを用いたコンテンツ作成環境「ピクトグラミング（Pictogramming）」を開発し Web 上で公開している．「ピクトグラミング」は，ピクトグラム（pictogram）とプログラミング（programming）を組み合わせた造語であり，初学者向けプログラミング学習環境の用途でも広く用いられている[2]．また我々の研究グループでは応用研究として，単眼カメラで撮影した画像の人間の姿勢の情報からピクトグラミングのコードを生成するモジュールを実装し，これを「ピクトグラミング」と連携することで，身体を実際に動かしてプログラムコードを生成できるシステムをすでに構築している[3]．本稿では，文献[3]の実装を発展させ，入力を画像から動画に拡張し，文献[3]と同様にピクトグラミングのコードを生成することで，アニメーション人型ピクトグラムを自動生成する手法を提案する．

以下 2 章，3 章でピクトグラミング及び画像からピクトグラミングのコードを生成するモジュールについて解説した上で，4 章で本提案について説明する．5 章でその評価を行い，6 章で関連研究，7 章で今後の展望について述べる．

2. ピクトグラミング

2.1. 画面説明

PC 用ブラウザでピクトグラミングにアクセスした場合のスクリーンショットを図 1 に示す．

画面は 2 つの部分から構成される．図 1 において左側は人型ピクトグラム表示パネル，右側はプログラムコード記述領域である．人型ピクトグラム表示パネルには，初期状態でパネル全体を占有するほどの大きさの人型ピクトグラムを配置している．

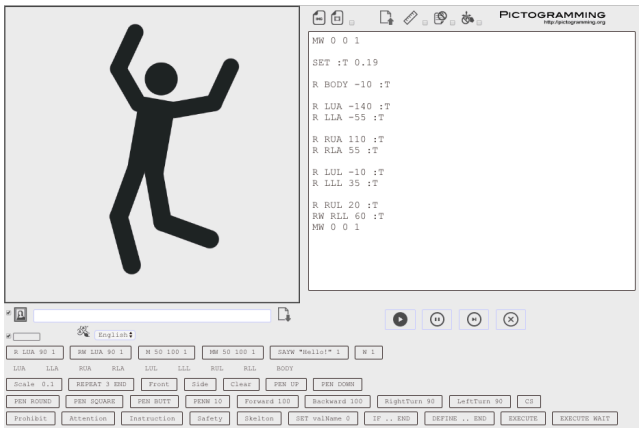

図 1 ピクトグラミングのスクリーンショット

人型ピクトグラム表示パネルには，ISO 3864 で定義されている正面方向あるいは側面方向の人型ピクトグラムのいずれかが表示される．正面方向のピクトグラムを図 2 に，側面方向の人型ピクトグラムの形状を図 3 に示す．いずれも体と頭を組み合わせた部分が 1 つと，上腕，前腕，大腿，下腿が左右それぞれ 1 つの計 9 種の部位から構成される．いずれの方向の人型ピクトグラムも各部位のサイズ比は ISO 3864 で定義されているものを忠実に再現している．

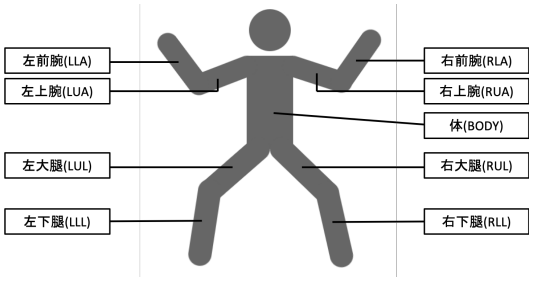

図 2 人型ピクトグラムを構成する部品(正面)

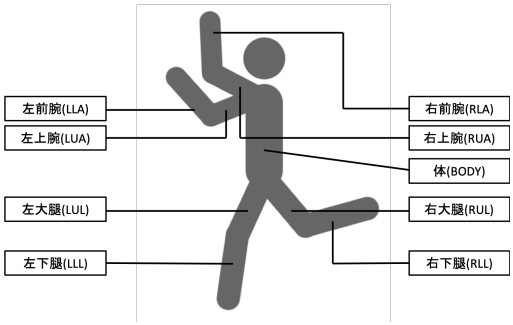

図 3　人型ピクトグラムを構成する部品(側面)

2.2. プログラム例と実行方法

人型ピクトグラムの動作は，プログラムコード記述領域に命令を入力し定義する．入力文字列は動作や状態を変化させるための「命令」コードと，引数列を空白で区切る以下の方式となっている．

命令 引数 1 引数 2 …

命令列を上から順番に列挙することでプログラムを作成する．プログラム例を図 4 に示す．これは，人型ピクトグラムが手を振るプログラムである．

1 行目の R(otate)命令は，LUA（左上腕: Left Upper Arm)を反時計回りに-120 度，つまり時計回りに 120 度，1 秒間かけて回転する．2 行目の R(otate)W(ait)命令は，RUL（右大腿: Right Upper Leg)を反時計回りに-10 度，1 秒間かけて回転する．

R(otate)命令は次の命令を同時に実行開始するが，R(otate) W(ait)命令は，命令の実行が終了するまで，次の命令の実行を開始しない命令である．逐次処理と並列処理を組み合わせることで，様々な動きを実現できる．3 行目から 6 行目では，左右に 3 回手を振る動作を行っている．

```
R LUA -120 1 // 左腕を斜めにあげる
RW RUL 10 1 // 右足を少し開く
REPEAT 3 // 左右に手を振る
RW LLA -60 0.3
RW LLA 60 0.3
END
```

図 4 プログラム例

3. 画像入力からのコード生成（文献[3]）

人間の身体動作の入力からピクトグラミングのコード出力し実行するプロセスであるヒューマンピクトグラムに関して，すでに我々の研究グループで構築した人間が写っている画像からコードを生成する方式を本章では述べる．

3.1. 部位の同定

人間が写っている画像から，姿勢の解析をはじめに行う，解析には，OpenPose[1][4]を用いている．画像に映る人物の肩，肘，肩など 18 点の位置推定を行う．図 5 に各キーポイントの番号と位置を示す．

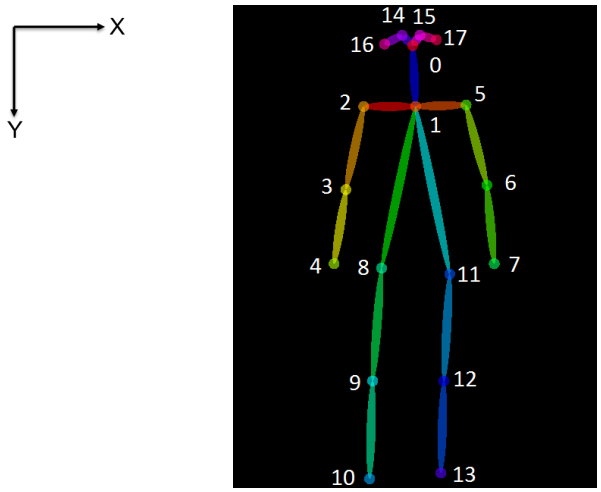

図 5　キーポイントの番号と位置

本プログラムでは，OpenPose が推定した 18 点のキーポイント(キーポイント番号 0～17)から，新たに 5 つのポイントを算出する．新たに算出する点の各座標について表 1 に示す．なお，キーポイント番号 n の x 座標を x[n]，y 座標を y[n]と表す．

[1] CMU-Perceptual-Computing-Lab が開発した機械学習型動作解析処理ライブラリである．単一画像から複数の人間の体や顔のキーポイントを検出することができる．

表 1　新たに算出した 5 点の座標

キーポイント番号 n	x 座標	y 座標
18	x[2]	y[3]
19	x[5]	y[6]
20	x[8]	y[9]
21	x[11]	y[12]
22	$\frac{x[8]+x[11]}{2}$	$\frac{y[8]+y[11]}{2}$

3.2. 角度の算出

3.1 節で示したキーポイントから，表 2 に示すキーポイント対応表に従いピクトグラミングにおける部位ごとの角度を算出する．図 2 には，角度の名前とそれに対応する始点と終点のキーポイント番号を示している．例として左前腕の角度，すなわち左肘の角度を求める方法を示す．

まず，1 点を共通の始点とし，終点の異なるベクトル 2 本を考える．左前腕の場合，共通の始点はキーポイント番号 6 であり，異なる 2 つの終点は，キーポイント番号 5，7 となる．この 2 本のベクトルの成す角を左前腕の角度とする．

表 2　部位ごとのキーポイント対応表

部位	始点	終点(順不同)	
右上腕（RUA）	2	3	21
右前腕（RLA）	3	2	4
左上腕（LUA）	5	6	20
右前腕（LLA）	6	5	7
右大腿（RUL）	8	9	18
右下腿（RLL）	9	8	10
左大腿（LUL）	11	12	19
左下腿（LLL）	12	11	13

全身の傾き(BODY)の算出方法は他の部位と異なる．BODY の値は，キーポイント番号 1 とキーポイント番号 22 を結ぶ直線の傾きによって算出する．

3.3. 処理例

3.1 節，3.2 節で解説した処理の例を図 6，図 7 に示す．図 6 を入力とした際の出力が図 7 となっている．

図 6　人間のポーズ例

```
R BODY 52.0
R LUA -82.0
R LLA 2.0
R RUA 75.0
R RLA -15.0
R LUL -57.0
R LLL 13.0
R RUL 8.0
```

図 7　図 6 のポーズを入力した場合の出力

4. アニメーションへの拡張手法の提案

4.1. 概要

本稿で提案する手法は，動画内に映っている人間の姿勢を推定し，その結果からアニメーションのピクトグラミングコードを生成するというものである．以下 4.2 節から 4.4 節で各処理の詳細を述べる．

4.2. 動画処理

はじめに入力動画をフレームに分割し，ピクトグラミングにおける最小動作単位時間である 0.02 秒ごとにフレーム情報を取得する．取得した画像情報ごとに処理を行う.動画のフレームレートが 50 未満の場合，毎フレームに処理を行い，フレームレートの逆数をアニメーションの動作単位時間とする．

4.3. 時系列データからの多項式近似

4.2 節にて取得したそれぞれの画像情報に対して，文献[3]と同様の手法で各部位の角度を算出する．フレーム i の時間 t(i), とそのフレームにおける部位 PART の角度$\theta_{PART}(i)$ からなる時系列データに対して近似を行う．時系列データの一部または全部に対して，次に

示す(1)から(2)の方式で，0 次，1 次または 2 次多項式近似を行う．

(1)　連続する 2 点（ t(i)，$\theta_{PART}(i)$ ）と（ t(i+1)，$\theta_{PART}(i+1)$ ）において，$\theta_{PART}(i)$と$\theta_{PART}(i+1)$の差の絶対値が 3 以内である場合，点(t(i)，$\theta_{PART}(i)$)を 0 次近似を行うべき区間の始点として記録する．その後，整数 j を i+1 からはじめて 1 ずつ加算していき，$\theta_{PART}(j)$と$\theta_{PART}(j+1)$の差の絶対値が 3 より大きくなった場合，点(t(j)，$\theta_{PART}(j)$)をその区間の終点とする．この区間内に存在する点の θ 座標の平均と$\theta_{PART}(i)$との差の絶対値が 3 以内である場合，その区間は 0 次近似を行う．これを i の小さい連続する 2 点から順に繰返し行う．

(2)　0 次近似を行わない連続する区間にそれぞれに対して，次に示す 2-a)から 2-c)の順番で処理し，一次近似するか二次近似するかのいずれかを決定する．

2-a)　処理対象とする 0 次近似を行わない連続する区間における最小のフレーム番号を start に，end を start+2 に設定する．

2-b)　時間区間[t(start)，t(end)]に対して算出される 1 次近似式を f(t)，2 次近似式を g(t)とする．その上で，　$\sum_{i=start}^{end}|f(t(i))-\theta_{PART}(i)|$ と $\sum_{i=start}^{end}|g(t(i))-\theta_{PART}(i)|$との大小を比べ，等しいまたは前者の値が小さければ 1 次近似式 f(t)を，それ以外の場合は 2 次近似式 g(t)の値を difference とする．

2-c)　delta が$\frac{difference}{end-start+1}$より小さければ，その区間の近似式を確定する．その上で start を end の値に，end を end+2 に更新する．それ以外の場合は，end を end+1 に更新し，2-b)を再度行う．end が 0 次近似を行う区間に含まれる場合は，end にその 0 次近似を行なった区間における t 座標が最小の点のフレーム番号を設定し，処理 2-b)を行い，最後の区間の近似式を確定する．

図 8 に，腕の上下運動を 2 回行う動作が収録されている動画から生成した右上腕(RUA)部に関する時系列データ及び，(1)，(2)の方式で定めた多項式近似を t-θ に関してグラフ化したものを示す．

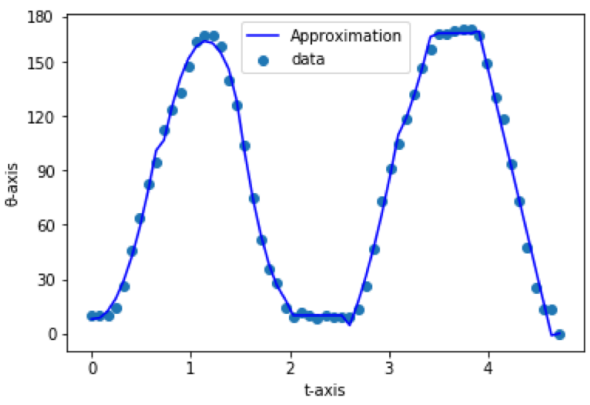

図 8　時系列データと近似式

4.4. コード生成

本プログラムでは，文献[3]と同様にプログラムコードを URI パラメタに指定して，ピクトグラミングの Web アプリケーションの URI にアクセスすることで，プログラムコードをピクトグラミング上で実行する．4.3 節にて確定した区間の近似式ごとにピクトグラミングのコードを生成する．生成したコードの例を図 9 に示す．

```
//2 次近似
SET :T 0
SET :OMEGA -22.0
REPEAT 6
SET :OMEGA [:OMEGA-3.0]
R RUA :OMEGA 0.08 [2.16+:T*0.08]
SET :T [:T+1]
END
R RUA 51.0 0.48 2.16 // 補完

//0 次近似
R RUA 0 1.76 2.64
R RUA -12.0 0 4.4  // 補完

//1 次近似
R RUA 150.0 1.2 4.4
R RUA -10.0 1.2 4.4  // 補完
```

図 9 生成したピクトグラミングのコード例

0 次近似を行なった区間は，区間の動作時間，部位を動かさない動作のコード化を行う．1 次近似を行なった区間は，区間の動作時間で，その区間における近似式の始点と終点の θ の差を回転する動作のコード化を行う．2 次近似を行なった区間は，単位時間あたりの回転量 OMEGA と動作開始時間に用いる T の 2 変数を用いてコードを表す．OMEGA の初期値には，近似式の始点における微分値を代入し，繰り返し処理内では，OMEGA に 2×a×time×time を加算する．ここで a は近似式における 2 次の項の係数であり，time は 4.2 節における動作単位時間である．

また，ある区間における近似式の終点の θ と次区間における近似式の始点の θ が異なるため，人型ピクトグラムが滑らかな動きをするように補完を行う．補完コードの開始時間と動作時間は，前者の区間と同じである．回転角度は，次区間における近似式の始点の θ から前者の区間における近似式の終点の θ を引いた値となる．なお，前者が 0 次近似した区間の場合，動作開始時間は，次区間の動作開始時間であり，動作時間

は次区間の回転時間とする．図 9 のコードにてコメントに「補完」と記載されている行の命令が補完コードの例である．

5. 評価と考察

5.1. 実装の評価

4 章で提案した実装プログラムの検証を 6 種類の動画用いて行った．いずれの動画も，単一の被験者が終始映っており，被験者には，カメラに向かって正面に立った状態で，自由に動いてもらう設定とした． 動画から生成した時系列データのうち，右上腕(RUA)部に関する時系列データ及び，多項式近似を t-θ に関してグラフ化したものを示し，それを元に生成したピクトグラミングのコードと共に図 10 から図 15 に示す．図 10，12，14 は，それぞれ 0 次近似，1 次近似，2 次近似を行なった例である．また，図 11，13，15 は，それぞれ図 10,12,14 に対して，近似を行なった結果から生成したピクトグラミングのコードである．

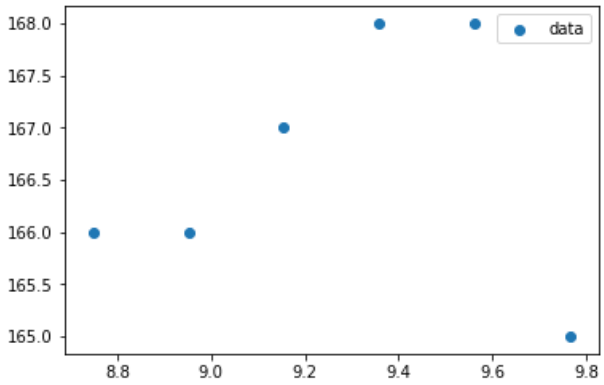

図 10　0 次近似を行なった区間における
時系列データの例

```
R RUA 0 1.0 8.6
```

図 11　図 10 の時系列データから生成される
ピクトグラミングのコード

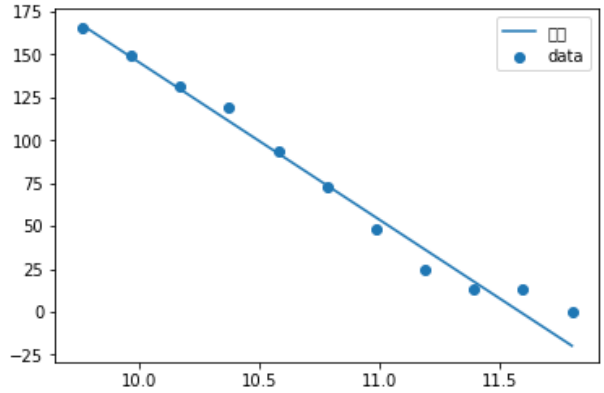

図 12　1 次近似を行なった区間における
時系列データと近似式の例

```
R RUA -183.7 2.0
```

図 13　図 12 の 1 次近似式で表現した
ピクトグラミングのコード

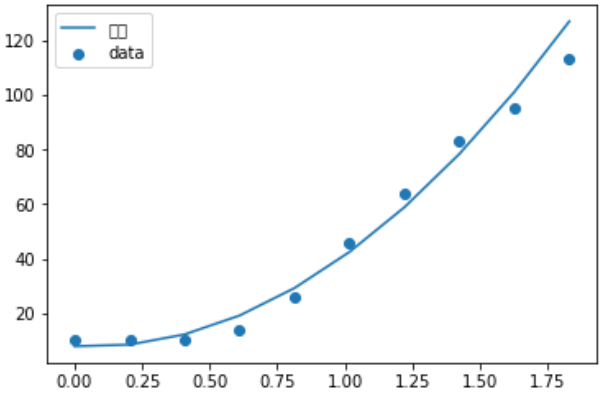

図 14　2 次近似を行なった区間における
時系列データと近似式の例

```
SET :T 0
SET :OMEGA -0.97
REPEAT 9
SET :OMEGA [:OMEGA+3.05]
R RUA :OMEGA 0.2 [0.0+:T*0.2]
SET :T [:T+1]
END
```

図 15　図 14 の 2 次近似式を表現した
ピクトグラミングのコード

5.2. 考察

5.1 節で示した通り，近似式に従いピクトグラミングのコードを生成することができた．

しかし，近似を行った結果，終点における時系列データ上の点と近似式上の点に大きな差が生じているケースが散見された．例を図 16 に示す．

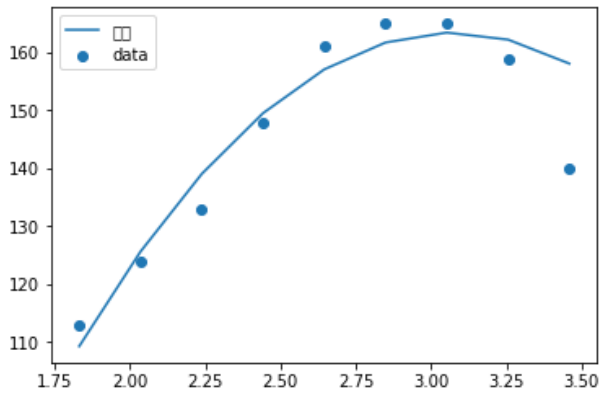

図 16　近似式と時系列データとの誤差が大きい例

また，被験者が各部位を激しく上下運動する動作をした際の右上腕(RUA)の時系列データを図 17 に示す．

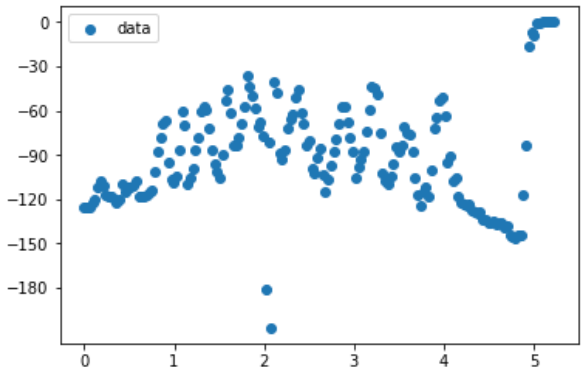

図 17　激しい上下運動の時系列データの例

動画内の人間が部位を素速く動かした結果，時系列データの点が散乱した．これに対して近似を行いピクトグラミングコードの生成を行なった．その結果，生成した人型ピクトグラムの部位の動きが，動画内の人間の動作と明らかに異なっていた．これは，4.3 節において，近似を行う最小区間を 2 点ではなく 3 点としていたことが原因であると推測できる．また，人間の動作の速度に対して，動画のフレームレートが小さすぎていたために，本来の動作とは異なる動作として観測されてしまうことも原因であると推測できる．前処理やグラフ生成アルゴリズム，近似手法の改良が検討課題である．

6. 関連研究

熊崎らによると，従来の対話型のインタラクティブシステムの研究では，人が対人的な反応を取ってしまう人工物と人の間に，人と人のような自然な関係性が構築されておらず，人と人工物の間に自然な関係性を構築するためには，人から人工物への自発的な関与を引き出す必要があると述べている．その上で，人は人間の姿勢から棒人間のような単純な物体でも，同じコンテキストのもとで，自分の振る舞いをデフォルメした棒人間にデザインすることで，その振る舞いから感情を推測し，共感できるという仮説を立て，実験によって検証している[5]．坂本らは両腕の上げ下げを命令できる，ひよこのキャクターを用いたプログラミング学習ツールを提案している．しかし，いずれも本件のような人間が姿勢をとることで直接姿勢を入力できるインターフェースは備えていない[6]．

鏡あるいは仮想的な鏡の前で，人が映った自分を見ながら動きを学習することは，日常行為だけではなく，スポーツや作業手順の動き習得などの分野でも幅広く行われている．小林らは，ドリトルから Kinect を利用し，関節の座標によって画面上のタートルを操作する環境の構築を行った[7]．しかし，Web カメラに比べ高価な Kinect を使用すること，ユーザが動作と機器の操作の対応関係を把握する必要があることから，金銭的なコストや学習コストの高さは，初学者向けの教育や学校での学習には不向きである．

人型ピクトグラムは，ユーザ自身との同一視を重視する点からも親和性が高く，ピクトグラミングを用いた本機能は，高い学習効果が期待できる．また，本稿で実装したプログラムでは，単眼カメラを使用しており，カメラを用意するコストも比較的低く抑えることができる．

7. まとめと今後の展望

本稿では，ヒューマンピクトグラミングにおいて，入力を画像だけではなく動画に拡張し，アニメーション人型ピクトグラムを自動生成する手法の提案を行なった．その結果，動画を入力として人間の姿勢を推定し，ピクトグラミングのコードを自動生成することができた．今後は，初中等教育機関での授業実践を踏まえ，手法の改良とともに評価を行う予定である．

参考文献

[1] 太田幸夫：国際安全標識のピクトグラムでデザインの研究，入手先〈http://www.tamabi.ac.jp/soumu/gai/hojo/seika/2003/kyoudou-ota1.pdf〉(参照 2019-2-01)

[2] 伊藤一成:ピクトグラミング - 人型ピクトグラムを用いたプログラミング学習環境 - 情報処理学会論文誌 教育とコンピュータ，Vol. 4，No. 2，pp. .47-61 (2018)

[3] 高橋将樹，伊藤一成：ヒューマンピクトグラミング，2018 年度 情報処理学会関西支部 支部大会 (2018)

[4] Cao，Z.，Simon，T.，Wei，S.E.，and Sheikh，Y.: Realtime multi-person 2D pose estimation using part affinity fields，Computer Vision and Pattern Recognition，arXiv:1611.08050 (2017)

[5] 熊崎周作，竹内勇剛: 他者性の知覚と共感を誘発する自己投影像，日本認知科学会第 31 回大会論文集，P3-10，pp. 724-730 (2014)

[6] 坂本一憲，本田澄，音森一輝，山崎頌平，鷲崎弘宜，深澤良彰: まねっこダンス：真似て覚えるプログラミング学習ツール，コンピュータソフトウェア，Vol.32，No.4，pp. 74-92 (2015)

[7] 小林史弥，本多佑希，島袋舞子，兼宗進: 生徒のプログラムから Kinect を利用するプログラミング環境の提案，情報処理学会第 79 回全国大会講演論文集，Vol.2017，No.1，pp. 637-638 (2017)