Vol.2022-CVIM-230 No.23
2022/5/12

深層指差し方向推定

中村 周[1]　野中 聡馬[1]　川西 康友[2]　延原 章平[1]　西野 恒[1]

概要：人間が興味や注意を表すジェスチャである指差しの方向を自動で推定することは，人間の意図の理解において重要な課題である．既存の研究では，深度カメラや複数台のカメラで近距離から撮影された画像に対し推定を行っているが，指差し推定を実用化するためには，監視カメラのような遠方から撮影された低解像度な単眼 RGB 画像に対する推定を実現する必要がある．このためには，特徴量をデータから直接学習することができる畳み込みニューラルネットワークが有効である．本研究では，写実的な指差しの画像とその方向の真値を持つデータセットを自動で作成する手法を 3 種類提案し，作成したデータセットでネットワークを訓練・評価した．評価実験の結果，複数種の訓練データを使用することで性能が向上することを確認した．

1. 指差し方向推定

物体に向かって指を差すという行為は，人間の持つ最も基本的なコミュニケーション形式の一つである．我々は指を差すことによって日常的に他者に興味の対象を伝えているため，その方向を自動で認識する技術は，人間とコミュニケーションを行うロボットやユーザーインターフェースの実現に必要不可欠である．

特に，画像からの指差し方向推定には応用上大きな意義がある．画像を使用することなく指差しを認識する場合，Bolt の研究 [2] のようにポインティング・デバイスを使用する必要があるが，この方法ではユーザーがデバイスを常に持つ必要があり，特に日常生活における実用性が低い．指示者の画像のみから指差し方向を推定できるようになれば，より簡便にコンピュータとコミュニケーションを取ることが可能になり，実用的なインターフェースの実現につながる．

画像からの指差し方向推定の手法は既存研究においても数多く提案されているものの，特殊なデバイスを使用しているため適用できるシーンが限定されているものが多い．例えば Fernández らの研究 [9] では，天井に設置した複数台の Kinect から得られた深度情報を用いて指差し方向を推定しているため，2.5×3m の操作領域内でしか指差しを認識することができない．Shukla らは同じく Kinect カメラを用いて指差し方向を高精度に推定できる手法を提案している [21] が，この手法は高解像度の RGB-D 画像を使用するため，Kinect のごく近くでしか推定を行えない．

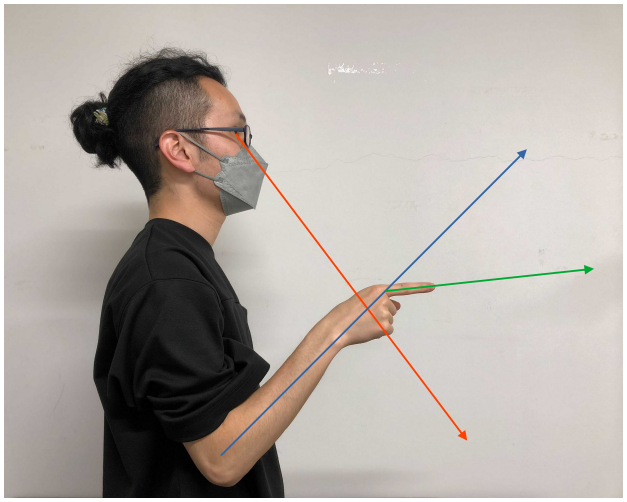

図 1　既存研究で用いられていた指差しの定義が適用できない例．指差しの方向を「顔と手を結ぶ半直線」と定義すると赤矢印が，「肘と手を結ぶ半直線」と定義すると青矢印が指差しの方向となるが，指示者が実際に意図しているのは緑矢印の方向である．

低解像度な RGB 画像からの指差し方向推定を行った既存研究自体は存在し，例えば Jaiswal らによるもの [13] や Watanabe らによるもの [25] が挙げられる．これらの研究ではそれぞれ，指差しの方向を「肘から手に向かう半直線」，「顔と手を結ぶ半直線」と定義している．しかし，人間が自然に指差しを行うとき，図 1 のようにこれらの前提が成り立たないような姿勢を取る場合がある．そのため，本研究では，指差しの方向を人差し指の付け根から先端に伸びる半直線の方向と定義する．

本研究では，この指差しの定義に基づき，指示者の姿勢に依存しない，低解像度の単眼 RGB 画像からの指差し方

[1] 京都大学大学院情報学研究科
[2] 理化学研究所

向推定を行う．畳み込みニューラルネットワークを使用することで，遠くから撮影され高解像度ではない単眼 RGB カメラの画像に対しても頑健な推定が行える．また本研究では，ネットワークの訓練および評価のために必要となる指差し画像のデータセットの作成手法を三種類提案する．いずれの手法も，人間の手による指差し方向の真値のアノテーションが不要であり，大量のデータを簡単に生成することができる．本研究の提案する推定手法は，既存の監視カメラや，搭載デバイスが限られている移動型ロボットの画像からの指差し方向の推定が可能であるため，日常の幅広いシーンに応用が可能である．

2. 関連研究

本節では，コンピュータにおける指差し認識に関する先行研究について説明することで，本研究のその中における立ち位置を示す．まず指差し認識の研究全体の歴史について概観し，次に，その中でも近年多く用いられている統計的手法に必要となるデータセットの作成手法について詳述する．

2.1 指差し認識

指差しの検知および方向推定を用いたユーザーインターフェースの構築は古くから研究されており，コンピュータビジョンに限らず様々な手法が提案されている．

初期の研究には，認識に画像を使わずに，専用のデバイスを用いて手の姿勢を得ているものがある．Bolt の研究 [2] は指差しをコンピュータに認識させた最初期の研究であり，手首や人差し指に装着したデバイスで三次元の位置および方向を測定した．これと音声認識を組み合わせることで，壁に投影した映像を指差しと発声で操作できるユーザーインターフェースを構築した．

次に，複数台のカメラを用いた手法が登場した．Fukumoto らは，天井と壁に設置されたカメラそれぞれで指先を検知することで指先の三次元座標を求め，指差しの方向を推定したり親指の曲げ伸ばしを判定することができるシステムを作成した [11]．Cipolla らは，30 度ほど離れた 2 台のカメラからの推定を統合して，40cm × 40cm のワークテーブルのどこを指差しているかを推定する手法を提案した [4]．Watanabe らは，指示者を囲うように等間隔に設置された 8 台のカメラを利用することで，その中央にいる指示者の指差しを 360 度どちら向きでも認識できる手法を提案した．Nickel と Stiefelhagen は，ステレオカメラからの動画に対して隠れマルコフモデルを適用することで顔と手の領域をトラッキングし，得られた 3 次元軌道を用いて指差しジェスチャの発生を検知した [17]．Fujita と Komuro は，2 台のカメラからの画像それぞれに対してパターンマッチングを適用した結果を統合することにより，指差しの精度を向上させた [10]．

画像と同時に深度を取得できる RGB-D カメラの普及に伴い，深度情報を利用する研究，特に，民生用 RGB-D カメラの代表的なものである Kinect を用いた研究も多く見られるようになった．Shukla らは，Kinect によって得られた手の領域の画像および深度情報に対して機械学習を適用することで，指差しの 3 次元方向を推定している [21]．Dhingra らは，畳み込みニューラルネットワークを用いて肘から手首のベクトルを求めることで指差しの三次元方向を推定し，その性能を詳細に評価した [7]．Fernández らは，スマートスペースにおいて指差しによる操作を実現するために，天井に設置した 2 台の Kinect を用いた指差し認識を行った [9]．深度情報のみを用いた指差し方向推定の研究として，Das によるものが挙げられる [5]．この研究では，正確に指差しを行おうとするとき同じユーザーは毎回同じ指差しの形をとるはずだという仮説に基づき，そのポーズの点群をユーザーごとに先に測定しておくことでマッチングを行っている．Das は RGB 情報を使わないことのメリットとして，照明条件や肌の色による推定への影響を避けられることを挙げている [5]．

近年では，単眼 RGB カメラからの画像のみを使って指差し認識を行った研究もみられる．Shiratori と Onoguchi は，手を伸ばした姿勢のみに指差し姿勢を限定することによって，全包囲カメラで部屋全体を撮影した低解像度な画像を使った指差し方向推定を行っている [20]．Mukherjee らは Web カメラの映像から人差し指の先端をリアルタイムで検知する手法を提案し，これをもとに空中に文字を書くことができるシステムを作成している [16]．Huang らもまた，一人称視点からの画像に対し，指先の座標をリアルタイムで検知し，空中に指で文字を書くことができるシステムを作成している [12]．

2.2 データセット

統計的手法による指差し方向推定のためには，手の画像と，その画像における指差し方向の真値を持つ教師データセットが必要となる．データセットは主に，CG によって合成しているものと，実際に撮影しているものの 2 つに分けられる．実際に撮影する場合，指差し方向の真値を同時に測定することが主な課題となる．

深度情報のみから指差しの認識を行った Das の研究 [6] では，実撮影によりデータセットを作成している．9 軸 IMU（慣性計測装置）を人差し指に装着することで指差し方向の真値を，また人差し指の先端に白色 LED を装着し複数方向から撮影することで三角測量によりその座標を測定する．その状態で指差しを行い，Kinect で深度情報を測定することで，手の（深度）画像と指差し方向真値のペアを持つデータセットを作成した．

一方で，CG によって様々なジェスチャをとった手の画像をレンダリングできるライブラリの一つに LibHand [24]

がある．例えば Shukla は，LibHand を用いて生成された合成データを使って指差し方向推定を行った [21]．

ハンドポーズ推定全体の文脈においては，CG でレンダリングされた高品質なデータセットがいくつか存在する．例えば，Mueller らによる SynthHands は Unity で手のモデルをレンダリングすることによって作成されている [15]．RGB-D カメラで撮影された手の画像から手のキーポイントを検出し，Unity 内での手のモデルをキーポイントに合わせて動かしたうえでレンダリングを行う．しかし，これらのデータセットを含め，ハンドポーズ推定全体を対象とした既存のデータセットにおける指差し画像の割合は非常に少なく，指差し方向推定モデルの訓練に十分な枚数が確保できない．

CG と実撮影の両方を利用してネットワークを訓練している研究に Shiratori と Onoguchi によるものがある [20]．彼らは，Unity を使って生成した大量の CG データで事前学習を行い，少量の実撮影データでファインチューニングを行った．実撮影データセットを作成する際には，指示者の手にレーザーポインタを持たせ，壁の光点を指差している場所を指差し方向の真値とした．

本研究ではマーカーを使用することなく実撮影データセットを作成しているが，その際行ったものと似た工夫を行っている研究に Simon らによるもの [22] がある．この研究のなかで彼らは，多視点のカメラから撮影した手の画像それぞれに対して既存のキーポイント推定機を適用し，得られたキーポイントを三角測量している．しかし，このようにして得られたキーポイントは誤っていることも多く，ここから直接三角測量を行うと誤差が大きくなる．そのため，三角測量の際には，キーポイントの確信度が高いカメラの視点のみを採用するなどといった複数の処理によって誤った測量結果を厳格に除去している．

3. 指差し画像データセット

RGB 画像から指差し方向の推定を行うニューラルネットワークを訓練するためには，写実的な手の画像とその指差し方向の真値のペアを持つデータセットが必要となる．既存研究では，人差し指に LED，IMU，指サックなどのマーカーを装着することで指差し方向の真値を取得してきた [5], [6] が，このような方法は手の見え方を変化させてしまう．また，CG によってデータを生成する試みも行われてきた [24] が，前腕までレンダリングされるものが存在しない．

そこで，本研究では，前腕を含む自然な手の画像に対して指差し方向をアノテーションした 3 種類のデータセットを作成する．1 つ目は 3DCG ソフトを用いて手の画像をレンダリングすることによって得たデータセットであり，これを合成データセットと命名する．2 つ目は指差しを行っている手を撮影することで得たデータセットであり，これを半合成データセットと命名する．3 つ目は住環境を模した空間で実際に指差しを行っている人の様子を撮影したデータセットであり，これをリビングデータセットと命名する．

図 2 合成データセットの画像例．腕の方向・回転，手首の曲げ具合，伸ばしている人差し指以外の曲げ具合，照明条件，前腕のテクスチャ，背景，カメラから手のモデルまでの距離を，それぞれ自然な範囲で画像ごとに変えることでデータセットの多様性を上げている．前腕のテクスチャを変えることで様々な服を模擬している．

3.1 合成データセット

合成データセットは，手の写実的な 3D モデルをレンダリングした画像に対し，その指差し方向の真値をアノテーションしたデータセットである．手の 3D モデルを利用する手法は，指の曲がり方や照明の向きなどのバリエーションを様々に変化させることで，自動的に大量のデータを作成できるという利点がある．

3D モデルを用いた手の画像の見え方を多様にするためには，使用する 3D モデルの向き，肌の色味，指の関節の曲げ方，光源の位置・強度，背景画像などを様々に変化させた画像を作成することが望ましい．そこで，豊富なレンダリング機能を持つ 3D ソフトである Blender と，手首や指の関節の曲がり方を自由に調整できるリグ付きモデル [19] を用い，これらのパラメータをランダムに変化させることで豊富なバリエーションの手の画像を作成する．レンダリングされた手以外に人間が写っていないことが望ましいため，背景の画像は人を含まない大規模データセットである PASS[1] から無作為に選ぶ．作成される画像例を図 2 に示す．

3.2 半合成データセット

第 3.1 節で導入した合成データセットは，大量の画像を自動で生成できるという利点がある一方，動かせる関節の自由度が低く，手の形状も変更できないので，生成される画像のパターンが限られているという欠点がある．そのため，指差しを実際に行っている手を撮影したデータセットで，かつ背景画像も自由に変更できるものを作成する必要がある．手の領域を切り抜き，背景画像を変更できるようにするために，撮影は白い紙で内張りされた箱の中で行う．

既存の指差し方向研究における実撮影データセット作成

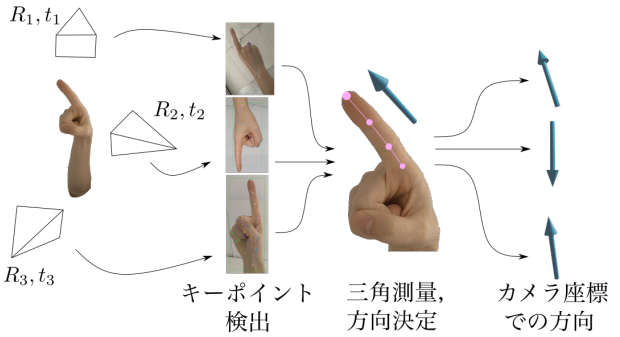

図 3 半合成データセットの作成における指差し方向の測定手順．三角測量を行うことで指差しの 3 次元方向を得て，それを各カメラ座標系に変換する．

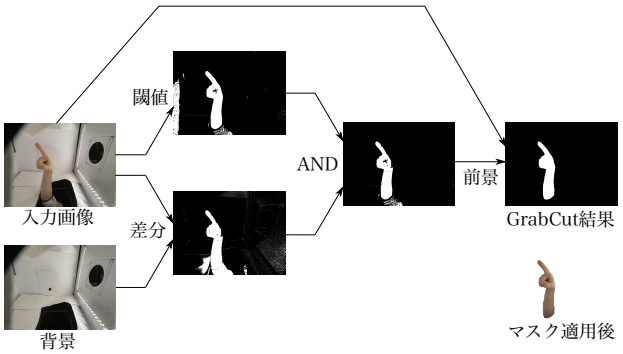

図 4 半合成データセットの作成において，撮影された画像から手の領域を切り抜く方法．HSV 空間での閾値と差分の両方の条件を満たす領域を前景として GrabCut を行う．

手法では，指差し方向の真値を得るためのマーカーにより撮影される画像が不自然なものとなってしまう問題があった．これを回避するため，本手法ではマーカーを使用せず，代わりに，図 3 に示すように複数の動画カメラを使って手を周囲から同期撮影し，既存の骨格推定モデル [3], [26] を用いて求めた手のキーポイントの三角測量によって指差し方向を測定する．三角測量にはカメラのパラメータが必要となるため，このデータセットの作成は，カメラパラメータを求めるためのキャリブレーション，手のキーポイントの三角測量，手の領域の切り出しの 3 つのステップからなる．このうち，キャリブレーションは球体を同期撮影して得た対応点を用いて行った．

3.2.1 キーポイント測量

手のキーポイント検出には既存の骨格推定手法の実装を使用する．物体検出モデル Cascade R-CNN [3] を用いて手の領域を切り出し，その領域に対して OneHand10k[27] で訓練されたキーポイント検出モデル Simple Baseline [26] を適用して各関節のキーポイントおよび推定の確信度を求める．Simple Baseline モデルにおいて，カメラ $i \in \{1,2,3,4\}$ のキーポイント $j \in \{1,2,\ldots,21\}$ に対する推定は $\left\{x_i^j, y_i^j, p_i^j\right\}$ の形式で出力される．$x_i^j \in [0,1920]$, $y_i^j \in [0,1440]$ はキーポイントの座標，$p_i^j \in [0,1]$ はキーポイント推定の確信度である．

ここで，精度良く三角測量を行うため，キーポイント検出に失敗しているカメラを除いて測量を行う．検出に失敗した際は確信度 p_i^j が低くなることを利用し，人差し指を構成する 4 つのキーポイント（$j = 6,7,8,9$）の確信度の積 $\prod_{j=6,7,8,9} p_i^j$ がある閾値 λ_{kp} より高かったカメラの集合 $S \subset \{1,2,3,4\}$ から得たキーポイントのみを三角測量に用いるようにする．選ばれたカメラの人差し指を構成するキーポイントに対して DLT 法による三角測量を行い，三次元座標 $\boldsymbol{X}^j := \left(X^j, Y^j, Z^j\right)^\top$ $(j = 6,7,8,9)$ を得る．なお，選ばれたカメラの台数 $|S|$ が 1 以下であった場合，三角測量が行えないためそのフレームは使用しない．

得られた 4 つの三次元座標から，再投影誤差が最も小さいものを 2 つ選び，その差分を指差し方向の真値とする．再投影誤差とは，人差し指を構成する 4 つのキーポイント $j \in \{6,7,8,9\}$ の座標と再投影座標の平均誤差

$$E(j) = \frac{1}{|S|} \sum_{i \in S} \sqrt{\left(x_i^j - \hat{x}_i^j\right)^2 + \left(y_i^j - \hat{y}_i^j\right)^2} \qquad (1)$$

のことである．ここで再投影座標 $\left(\hat{x}_i^j, \hat{y}_i^j\right)$ とは，三角測量したキーポイントをカメラ i に再投影したときの座標のことであり，$\left(\hat{x}_i^j, \hat{y}_i^j, 1\right)^\top \sim K\left(R_i|t_i\right)\tilde{\boldsymbol{X}}^j$ という関係から算出できる（ただし，$\sim$ は比例を，R_i, t_i はカメラ i の回転行列および並進ベクトルを，$\tilde{\boldsymbol{X}}^j = \left(\boldsymbol{X}^{j\top}|1\right)^\top$ は $\boldsymbol{X}^j$ の共役座標を表す）．再投影誤差が小さいほうからカメラを 2 つ選び，$j_1, j_2\,(j_1 < j_2)$ とする．この 2 つの平均再投影誤差の合計 $E(j_1) + E(j_2)$ がある閾値 λ_{rp} を超えていた場合は，三角測量の精度に問題があると考えられるためこのフレームを使用しない．

こうして得た三次元座標は，$\boldsymbol{X}_i^j := \left(R_i|t_i\right)\tilde{\boldsymbol{X}}^j$ のように，それぞれのカメラの外部パラメータをかけることでカメラ座標系に変換することができる．カメラ座標系での三次元座標の差分 $\boldsymbol{X}_i^{j_2} - \boldsymbol{X}_i^{j_1}$ を長さ 1 に正規化したものを指差し方向の真値とする．

3.2.2 手領域の切り出し

本データセットでは，撮影された画像から手の領域を切り出し，PASS データセットから無作為に選んだ画像に貼り付けることで背景の多様性を高めている．図 4 に示すように，手の領域の切り出しは，まずピクセルごとの輝度の条件から手の領域を推定し，それを GrabCut[18] で修正するという 2 段階で行う．

手の領域の推定には，HSV 空間での閾値と画像差分を使用する．閾値は撮影映像にあわせて手動で設定した．更にそのうち，手が写っていない最初のフレームの輝度との差分が 5%以上ある画素のみを選択する．

推定された領域を前景として GrabCut を適用する．GrabCut は，指定された前景（選び取るべき領域）と背景（捨てるべき領域）に基づいて，グラフカットを適用して画

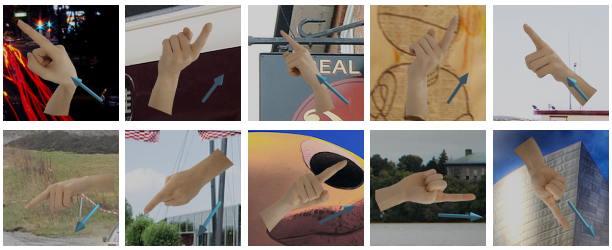

図 5 半合成データセットの画像例．参考のため，それぞれの画像の右下にシアン色の矢印で測定された指差し方向を表示している．指差し方向の測定が成功していることがわかる．

図 6 リビングデータセットの画像の一つ．壁に貼られたマーカーを指差す被験者が写っている．

像の切り出しを行うアルゴリズムである．前段落で求めた画素を前景に指定することで，手の領域を決定する．合成データセットと同様，PASS データセットから無作為に選んだ画像に対して手の領域を貼り付ける．

3.3 リビングデータセット

現実の人の指差しの様子を撮影したデータで提案手法を評価するため，住環境を模した環境で行動する人物を定点カメラから撮影したデータセットを作成する．このデータセットでは，理化学研究所情報統合本部に設置された，リビングルームを模した室内で自由に歩き回る人物が実際に指差しを行う様子を複数台のカメラから撮影し，三角測量によって指差し方向を測定している．図 6 にデータセットの画像例を示す．

撮影が行われた部屋の壁や床には 75 枚程度の AR マーカーが貼り付けられている．被験者がこの中から無作為に選んだマーカーを指差しながら 5 分間程度室内を自由に動き回る様子を，部屋の天井四隅に設置されたカメラ（解像度 1224px × 1024px）から撮影した．

このデータセットでは，AR マーカーを活用して指差し方向を測定している．まず，AR マーカーの座標を対応点としてカメラの外部パラメータ及び AR マーカー自体の三次元座標を求める．既存の骨格推定手法で得た関節点についても三角測量を行い，被験者の手の三次元座標を求める．

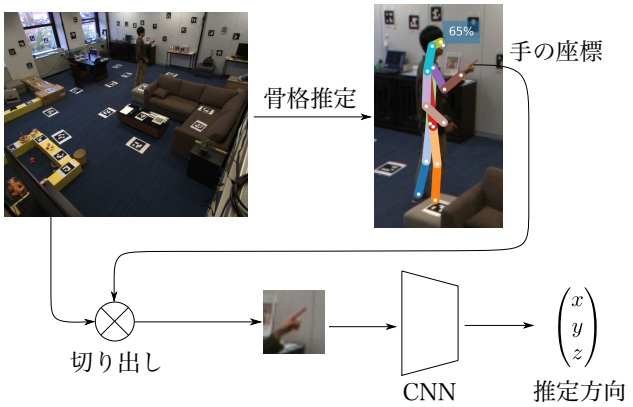

図 7 本論文で提案する指差し推定器の構造．入力は被験者の体全体を写す程度の画角を持つ RGB カメラによる画像であり，出力は三次元の指差し方向である．骨格推定には OpenPifPaf [14] を用いる．

AR マーカーには一意な番号が振られており，被験者は指差しているマーカーの番号を声に出して指定しながら指差しを行うため，被験者の指差しているマーカーの座標を求めることができる．手の三次元座標は指差しの起点，指定されたマーカーの三次元座標は指差しの対象と考えることができるので，この 2 つの座標の差分を取ることで指差しの三次元方向を求めることができる．

骨格推定には Kreiss らによって提案された OpenPifPaf [14] を用いる．COCO Keypoint Detection データセットで訓練された OpenPifPaf モデルを使い，全身の骨格からキーポイント 17 点 $\{\boldsymbol{x}_i\}$ を推定する．肘のキーポイントから手首のキーポイントへの線分を 1.5 倍に延長した点 $\boldsymbol{x}_{\text{right/left wrist}} + 0.5 \times \left(\boldsymbol{x}_{\text{right/left wrist}} - \boldsymbol{x}_{\text{right/left elbow}}\right)$ を手の座標であると考え，この点を DLT 法によって三角測量することで指差しの起点の座標とする．使用するキーポイントの確信度がある値を下回っていた場合はそのフレームを使用しない．

4. 深層指差し方向推定

本節では，本論文で提案する深層指差し方向推定の手法と，その学習について説明する．

4.1 アーキテクチャ

図 7 に提案手法の概略を示す．提案手法は 2 次元骨格推定，手領域の切り出し，指差し方向推定の 3 つのステップからなる．

まず二次元骨格を推定して手の座標を求める．二次元骨格の推定にはリビングデータセットの節と同じく COCO Keypoint Detection タスクで訓練された OpenPifPaf [14] を用い，全身の骨格からキーポイント 17 点を推定する．手の座標もリビングデータセットと同様，肘から手首への線分を 1.5 倍に延長した地点とする．

得られた手の座標を中心として正方形の領域を切り出す．切り出す大きさは指示者の画像中の大きさに依存せず一定とすることが望ましいため，胴体の大きさの 0.6 倍とする．胴体の大きさは腰から肩までの長さ，すなわち

$$\left\| \frac{\boldsymbol{x}_{\text{left shoulder}} + \boldsymbol{x}_{\text{right shoulder}}}{2} - \frac{\boldsymbol{x}_{\text{left hip}} + \boldsymbol{x}_{\text{right hip}}}{2} \right\|_2 \tag{2}$$

として求める．これらのキーポイントの確信度の積が 0.2 を下回った場合は，骨格推定に失敗している場合が多いため，推定を行わない．

切り出した画像を畳み込みニューラルネットワークに入力し，指差し方向を推定させる．使用するニューラルネットワークは，128×128 の RGB 画像を入力とし，EfficientNet-B0 [23] と同じコンボリューション処理を行ったあとに全結合層による処理を行うものである．平坦化されたコンボリューション層の出力に対し，128, 128, 3 次元の全結合層をおく．活性化関数にはすべて SiLU [8]：

$$\mathrm{SiLU}(x) = x \cdot \sigma(x) = \frac{x}{1 + \mathrm{e}^{-x}} \tag{3}$$

を用いる．ニューラルネットワークが出力した 3 次元のベクトルを長さ 1 に正規化し指差し方向とする．

4.2 学習方法

提案するアーキテクチャの方向推定ニューラルネットワークを，合成データセットと半合成データセットを用いて訓練する．

訓練に際し，予測対象と訓練データセットの画質に差が存在するため，これを解消するための前処理を行う．画質の差の例として，解像度の違いが挙げられる．例えば，本研究では低解像度な手の画像からの推定を目標としており，リビングデータセットの手の領域の画像サイズは 40×40 から 100×100 程度である．一方で合成データセットをレンダリングする際，解像度は 256×256 としており，半合成データセットの解像度も 500×500 から 1100×1100 程度と大きいため，この違いを解消する必要がある．

画質の差を解消するために，訓練画像に対し，画像リサイズ，画像をぼかすガウシアンブラー，画像全体の色調を変更するカラージッターの 3 つの前処理を行う．以下 $U(a,b)$ は，画像ごと，処理ごとにランダムに決定される $[a,b]$ の範囲の一様分布関数とする．まず，画像サイズを 150×150 に縮小し，そこから 128×128 の大きさになるようにランダムな場所を切り取る[*1]．次に，カーネルサイズ 17 px, 標準偏差 $\sigma = U(5,10)$ のガウシアンブラーをかける．最後に，画像の明度を $U(0.7, 1.3)$ 倍し，色相を $U(-3,3)$% 変化させ，彩度を $U(0.4,1)$ 倍する．これらの処理を行った入力画像の比較を図 8 に示す．

[*1] すなわち，左上の頂点が $(a,b) = (U(0,22), U(0,22))$ で，右下の頂点が $(a+128, b+128)$ である正方形領域を切り取る

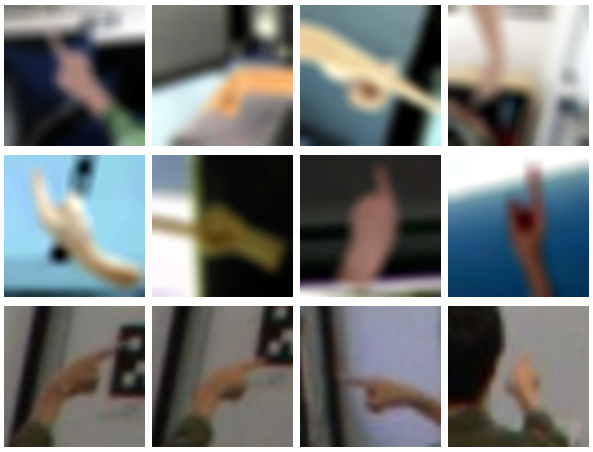

図 8 訓練時，ニューラルネットに入力するための前処理を行ったあとのそれぞれのデータセットの画像．上段から順に，合成データセット，半合成データセット，リビングデータセット．合成データセットと半合成データセットについて，第 4.2 節で述べた方法でランダムな前処理を行っている．

学習時の設定を説明する．損失関数は $L = L_\theta + \lambda L_2$ とする．ここで L_θ は角度誤差であり，推定値 $\tilde{\boldsymbol{x}}$，真値 $\boldsymbol{x}$ に対し，

$$L_\theta = \cos^{-1}\left(\mathrm{clamp}\left(\frac{\tilde{\boldsymbol{x}} \cdot \boldsymbol{x}}{\|\tilde{\boldsymbol{x}}\| \, \|\boldsymbol{x}\|}, -1+\epsilon, 1-\epsilon\right)\right), \tag{4}$$

$$\epsilon = 1 \times 10^{-7} \tag{5}$$

と定義する．ただし

$$\mathrm{clamp}(x,a,b) \coloneqq \begin{cases} a & (x < a) \\ x & (a \le x \le b) \\ b & (b < x) \end{cases} \tag{6}$$

であり，これにより $\cos^{-1}$ の引数が 1 に近いとき微分が発散してしまう問題を回避する．L_2 はニューラルネットの重み全体に対する L2 正則化項であり，係数 $\lambda = 0.4$ とした．全結合層の隠れ層では確率 0.1 のドロップアウトおよびバッチ正則化を行った．最適化アルゴリズムとして SGD を使用し，初期学習率は 4×10^{-4} とした．10 エポックに渡って訓練時の平均損失が減少しなかったとき学習率を 0.9 倍するよう設定し，バッチサイズを 128 として 2000 エポック学習させた．

合成データセット 25000 枚，半合成データセット 9005 枚のうちそれぞれ 500 枚（合計 1000 枚）を訓練に使用せず，250 枚ずつをバリデーションデータとテストデータとした．各エポックの最後にバリデーションデータに対する損失を計算して，バリデーションデータに対する損失が一番小さいモデルを採用した．テストデータは評価実験で使用した．

5. 評価実験

本節では，半合成データセットを作成した際の結果を詳

述し，提案手法の精度を評価する．提案手法の未知ドメインへの汎化性能を調べるため，訓練データのうち 250 枚のテストデータに加え，396 枚のリビングデータセットで評価を行う．

5.1 半合成データセットの作成結果

撮影は，SyncBac を用いて同期された GoPro 7 カメラ 4 台で行った．撮影時の設定は，Linear モード[*2]，解像度 1920×1440，60FPS とした．

データセットを作成するために，指差しを行う様子を約 4 分 40 秒，16820 フレームに渡って撮影した．これに対し，三角測量の正確性に関する閾値 $\lambda_{\mathrm{kp}} = 0.4$, $\lambda_{\mathrm{rp}} = 10\,\mathrm{px}$ とし，そのうち 3748 フレームを選択した．4 台のカメラで同時に撮影しているので，三角測量に成功している画像は $3748 \times 4 = 14992$ 枚存在する．GrabCut に切り出された領域が画面の端に触れているものを除くことで手が画角の外に出てしまっている画像を排除し，最終的に 9005 枚の画像を得た．

三次元座標から計算される指差しの方向は，遠近法のために画像から感じられる方向と異なっている．この問題を解決するため，得られた三次元座標に対して次の変換を行ったのちに指差し方向を決定する：

$$\boldsymbol{X} = \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \to \hat{\boldsymbol{X}} = \begin{pmatrix} f\frac{X}{Z} + u_0 \\ f\frac{Y}{Z} + v_0 \\ \alpha Z \end{pmatrix} = \begin{pmatrix} \hat{x} \\ \hat{y} \\ \alpha Z \end{pmatrix}. \quad (7)$$

x 座標と y 座標はカメラ平面に投影することで世界座標系から画像座標へと変換しており，z 座標（奥行き方向）はスケールを世界座標系から画像座標系に変換するため定数 α を掛けている．スケールを決定するために，伸ばした人差し指のデータセット内における平均的な長さを測ったところ，世界座標系では 0.209，画像座標系では 250 px であったため，$\alpha = 250/0.209 = 1.20 \times 10^3\,\mathrm{px}$ とした．

5.2 精度評価

本節では，提案するモデルの方向推定の精度を評価する．評価の指標には，真の方向 $\boldsymbol{x} = (x, y, z)^\top$ と推定方向 $\hat{\boldsymbol{x}} = (\hat{x}, \hat{y}, \hat{z})^\top$ の間の誤差角度 $\theta = \cos^{-1}(\boldsymbol{x} \cdot \hat{\boldsymbol{x}})$ を用いる．合成データセットのみで訓練したモデルと，合成データセットと半合成データセットの両方で訓練したモデルの 2 つをそれぞれ評価した．評価するためのデータとして，合成データセットと半合成データセットのテストデータ，およびリビングデータセットを用いた．

表 1 に比較の結果を示す．合成データセットのみで訓練されたモデルは，両方で訓練したモデルよりも合成データセットにおける精度が良いものの，未知のデータである半合成データセットおよびリビングデータセットの推定誤差が大きくなってしまっている．一方で，合成データセットに半合成データセットを混合したデータセットで訓練されたモデルは，加えた半合成データセットだけではなく，未知のリビングデータセットにおける推定誤差がより小さくなっており，現実のデータに対する性能が向上している．

[*2] 歪みパラメータが 0 になるように GoPro 内で事前処理するモード

	合成データ	半合成データ	リビングデータ
合成	4.54°	28.56°	53.66°
合成＋半合成	4.73°	5.50°	46.78°

表 1 それぞれのデータセットに対する推定誤差 θ の平均値．合成データと半合成データについては，訓練に使用していないテストデータを用いて評価している．一行目が合成データセットのみで訓練されたモデルの，二行目が合成データセットと半合成データセットの両方で訓練されたモデルの評価結果である．

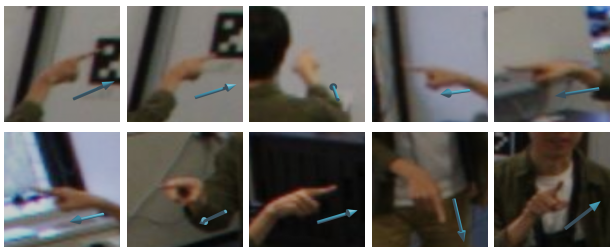

図 9 合成データセット，半合成データセットの両方を用いて訓練されたモデルを用いた，リビングデータセットの画像に対する推定の結果．とくに，上段中央の画像は図 6 の画像に対して推定を行った結果である．それぞれの画像の右下にシアン色の矢印で推定された指差し方向を表示している．画像平面内の二次元的な指の方向は概ね正しく推定できている一方，奥行き方向の推定が正確ではないケースが存在することがわかる．

しかし，混合データセットで訓練されたモデルであっても，リビングデータセットにおける誤差は 46.8 度と比較的大きい．この原因を確かめるため，図 9 に，合成データセットと半合成データセットの両方で訓練されたモデルをリビングデータセットに適用した例を示している．多くの場合概ね正しい方向を推定できているが，画面奥行き方向の推定誤差が大きい場合が存在することが分かる．このことによって推定精度が低下していると考えられる．

6. 結論

本研究では，単眼 RGB カメラの低解像度な画像に対して深層指差し推定を行うためのアーキテクチャおよび，データセットの作成手法を 3 つ提案した．データセットのうち 2 つで訓練を行い，訓練されたネットワークの性能をもう一つのデータセットを使って評価した．提案するデータセット作成手法はいずれも人間による指差し方向真値のアノテーションが不要であり，大量のデータを容易に生成することができる．作成した 2 つの訓練用データセット両方を使って訓練することにより，片方だけで訓練した際よりも現実のデータに対する性能が向上することが確認された．今後解決すべき課題として，画面奥行き方向の推定

精度が高くないことが挙げられる．指示者の胴体や頭部といった，より視覚的変化に富んでいて，方向推定の行いやすい部分の情報も利用することで，精度を向上させることができると考えられる．

謝辞

この研究の一部は JSPS 20H05951, 21H04893, JST JPMJCR20G7, 理研 GRP の助成を受けて行ったものです．

参考文献

[1] Asano, Y. M., Rupprecht, C., Zisserman, A. and Vedaldi, A.: PASS: An ImageNet replacement for self-supervised pretraining without humans, *NeurIPS Track on Datasets and Benchmarks* (2021).
[2] Bolt, R. A.: “Put-That-There”: Voice and Gesture at the Graphics Interface, *Proc. SIGGRAPH*, (online), DOI: 10.1145/800250.807503 (1980).
[3] Cai, Z. and Vasconcelos, N.: Cascade R-CNN: High Quality Object Detection and Instance Segmentation, *IEEE TPAMI*, Vol. 43, No. 5, pp. 1483–1498 (2021).
[4] Cipolla, R., Hadfield, P. A. and Hollinghurst, N. J.: Uncalibrated stereo vision with pointing for a man–machine interface, *IAPR Workshop on Machine Vision Applications*, pp. 163–166 (1994).
[5] Das, S. S.: Precise Pointing Direction Estimation using Depth Data, *International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pp. 202–207 (2018).
[6] Das, S. S.: A data-set and a method for pointing direction estimation from depth images for human-robot interaction and VR applications, *Proc. ICRA*, pp. 11485–11491 (2021).
[7] Dhingra, N., Valli, E. and Kunz, A.: Recognition and Localisation of Pointing Gestures Using a RGB-D Camera, *HCI International*, pp. 205–212 (2020).
[8] Elfwing, S., Uchibe, E. and Doya, K.: Sigmoid-weighted linear units for neural network function approximation in reinforcement learning, *Neural Networks*, Vol. 107, pp. 3–11 (2018).
[9] Fernández, A., Bergesio, L., Bernardos, A. M., Besada, J. A. and Casar, J. R.: A Kinect-based system to enable interaction by pointing in smart spaces, *IEEE Sensors Applications Symposium* (2015).
[10] Fujita, D. and Komuro, T.: Three-Dimensional Hand Pointing Recognition Using Two Cameras by Interpolation and Integration of Classification Scores, *ECCV WS*, pp. 713–726 (2015).
[11] Fukumoto, M., Mase, K. and Suenaga, Y.: Real-time detection of pointing actions for a glove-free interface, *In IAPR Workshop on Machine Vision Applications*, pp. 473–476 (1992).
[12] Huang, Y., Liu, X., Zhang, X. and Jin, L.: A Pointing Gesture Based Egocentric Interaction System: Dataset, Approach and Application, *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 370–377 (online), DOI: 10.1109/CVPRW.2016.53 (2016).
[13] Jaiswal, S., Mishra, P. and Nandi, G.: Deep Learning based Command Pointing direction estimation using a single RGB Camera, *2018 5th IEEE Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON)*, pp. 1–6 (online), DOI: 10.1109/UPCON.2018.8596762 (2018).
[14] Kreiss, S., Bertoni, L. and Alahi, A.: OpenPifPaf: Composite Fields for Semantic Keypoint Detection and Spatio-Temporal Association, *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–14 (2021).
[15] Mueller, F., Mehta, D., Sotnychenko, O., Sridhar, S., Casas, D. and Theobalt, C.: Real-time Hand Tracking under Occlusion from an Egocentric RGB-D Sensor, *Proc. ICCV* (2017).
[16] Mukherjee, S., Ahmed, S. A., Dogra, D. P., Kar, S. and Roy, P. P.: Fingertip detection and tracking for recognition of air-writing in videos, *Expert Systems with Applications*, Vol. 136, pp. 217–229 (2019).
[17] Nickel, K. and Stiefelhagen, R.: Pointing Gesture Recognition Based on 3D-Tracking of Face, Hands and Head Orientation, *Proceedings of the 5th International Conference on Multimodal Interfaces*, pp. 140–146 (2003).
[18] Rother, C., Kolmogorov, V. and Blake, A.: “GrabCut”: Interactive Foreground Extraction Using Iterated Graph Cuts, *ACM TOG* (2004).
[19] sarojit: 3D Female Right Hand Rigged model (2020).
[20] Shiratori, Y. and Onoguchi, K.: Detection of Pointing Position by Omnidirectional Camera, *Intelligent Computing Theories and Application*, pp. 774–785 (2021).
[21] Shukla, D., Erkent, O. and Piater, J.: Probabilistic Detection of Pointing Directions for Human-Robot Interaction, *International Conference on Digital Image Computing: Techniques and Applications (DICTA)* (2015).
[22] Simon, T., Joo, H., Matthews, I. and Sheikh, Y.: Hand Keypoint Detection in Single Images Using Multiview Bootstrapping, *Proc. CVPR* (2017).
[23] Tan, M. and Le, Q.: EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks, *Proc. ICML*, pp. 6105–6114 (2019).
[24] Šarić, M.: LibHand: A Library for Hand Articulation, http://www.libhand.org/ (2011).
[25] Watanabe, H., Hongo, H., Yasumoto, M. and Yamamoto, K.: Detection and Estimation of Omni-Directional Pointing Gestures Using Multiple Cameras., *IAPR Workshop on Machine Vision Applications*, pp. 345–348 (2000).
[26] Xiao, B., Wu, H. and Wei, Y.: Simple Baselines for Human Pose Estimation and Tracking, *Proc. ECCV* (2018).
[27] Yangang Wang, C. P. and Liu, Y.: Mask-pose Cascaded CNN for 2D Hand Pose Estimation from Single Color Images, *IEEE TCSVT*, Vol. 29, No. 11, pp. 3258 – 3268 (online), DOI: 10.1109/TCSVT.2018.2879980 (2019).