2025 年度（令和 7 年度）
創造工学セミナー II Final Report

単眼カメラからの人物分布推定に関する研究

研究メンバー
S522007 大城 寛剛
S522055 米原 碧海
S523401 佐藤 真之介

指導教員
金丸 隆志 教授

所属研究室
知能機械研究室

目次

第 1 章 緒論（大城）

1.1 研究背景

本節では、本研究の背景について説明する。

1.1.1 近年における混雑問題

近年、新型コロナウイルス感染症の流行の収束などから訪日する外国人旅行者が急増している。図 1 は日本政府観光局が出典している 2003 年から 2025 年までの訪日外国人旅行者と出国日本人数の推移を表したグラフであり、2023 年以降訪日する外国人旅行者の数が増えているのがわかる[1]。そのため特定のエリアが過度に混雑する問題などが起こっている。例をあげると、東京都の豊島区では 2024 年の年間の観光客数は約 2,367 万人であった[2]。それに加え、豊島区はもともと人の集中度合が極めて高く、防災上の危険性や災害時の混乱が懸念されている[3]。

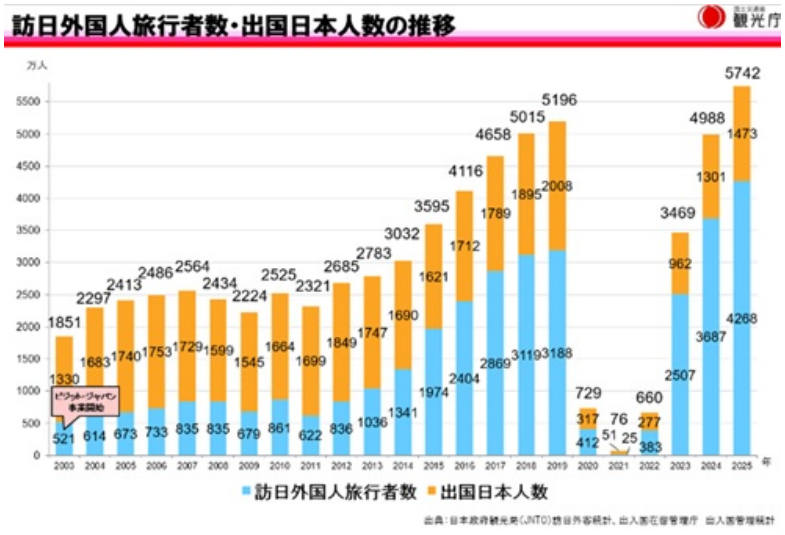

図 1　訪日外国人旅行者数・出国日本人数の推移[1]

1.1.2 群衆行動解析技術の注目

群集行動解析技術とは、複数の人間が集団として形成する動きや分布、密度の変化などを解析することにより、混雑状況を把握する技術である。従来の映像解析技術では、映像から人物を一人ずつ切り出して行動解析を行っていた。そのため人物同士の重なりに弱いという課題があった。そこで、群衆画像を「何人かの集団」として画像を部分領域に分割し、各

部分領域が何人から成る集合であるかを解析することによって、群衆画像から個別に人物を切り出すことなく、大まかな人数を自動推定することが可能になった。実際に、豊島区では、群衆行動解析技術を活用した総合防災システムが導入されている。豊島区に設置してある監視カメラの映像からコンピューターにより、混雑度、群衆の流れなどを自動で解析し、リアルタイムでの検知を行っている。今後の展望として、多くの人が集まる公共空間や大型施設（ターミナル駅・空港・テーマパーク・イベント会場など）での適用範囲の拡大を目指している[3]。

以上より、カメラ映像から群衆行動解析技術を用いて人物分布推定を行うことに需要があると考えた。

1.2 先行研究

本節では、本研究に関する先行研究として複数カメラを用いた人数分布推定に関する検討を紹介する。

田渕ら(2013)は、群集行動解析技術における課題の一つである場所・領域ごとの人数分布推定に着目し、複数カメラを用いた人数分布推定手法を提案した[5]。従来の人数推定手法は、カメラ視野内全体の人数推定に主眼を置いたものが多く、人数の空間的な分布を考慮していないという課題がある。また、単一カメラによる解析では人物同士の遮蔽の影響により正確な人数把握が困難である点が指摘される。

田渕らが提案したシステムは、複数台のカメラを用いることで遮蔽の影響を低減し、床面を複数の小領域に分割した上でそれぞれの領域に存在する人数を回帰モデルにより推定する手法を提案した（図 2)。具体的には、前景画素数やエッジ画素数などの画像特徴量を用い、撮影面分割領域と床面分割領域の対応関係に基づいて人数分布を推定している。

実験では 300cm×300 cm の床面を 3×3 の 9 つの領域に分割し，2 台のカメラを約 90 度の位置に設置し両カメラの注視点が同じになるように調整し、人数の分布の推定を行なった。

実験結果から、田渕らの手法は一定の精度で人数分布を推定できることが示されている。その一方で、カメラから遠い領域や高密度な領域では遮蔽の影響が依然として大きく、推定精度が低下する傾向が報告されている。また、学習データの偏りや、周囲領域の人物が推定結果に影響を与える点が課題として挙げられている。

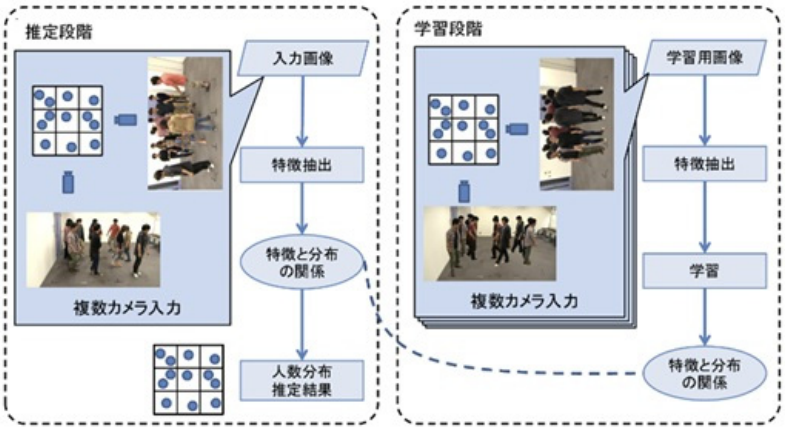

図 2　人数推定を用いた人数分布推定手法の処理の流れ[5]

1.3 研究の目的（佐藤）

1.1.2 項より、監視カメラからの群集映像を対象とした自動解析技術の需要が高まっているなかで、1.2 節よりカメラからの映像入力で、より正確な人数の分布推定を行おうとする先行研究もある。

先行研究では指定された領域を 2 台のカメラを用いて複数視点で映すことで、人物の重なりによる遮蔽の影響を低減した。しかし、実際の環境下において同一領域を複数カメラから捉えられている状況は限られている。

そこで我々は、単眼カメラを用いて重なりに強い人物分布推定を行うことを目標に人物分布推定システムの構築を行った。

第 2 章 構築を目指す人物分布推定システムについて（佐藤）

2.1 構築を目指す人物分布推定システムの概要（米原）

1.3 節で先述したように、単眼カメラを用いて重なりに強い人物分布推定を行うシステムの構築を目的とする。

対象とする領域は、300cm×300cm の正方形領域とし、これを縦横ともに 3 分割した 9 つのエリアに分ける（図 3)。各エリア内に存在する人数を推定することで、領域内の人物分布を把握することを目指す。

図 4 は、我々が構築を目指すシステムの完成イメージであり、左側が、3×3 の領域内に人物を配置した実際の映像で、右側が、領域内にどのように人物が配置されているかをシステムで推定した結果のイメージとなっている。

本システムを実現するためには、各人物が領域内のどの位置に存在しているかを把握する必要がある。そこで我々は、足元位置を用いる。足元位置を用いる理由は、人物が実際に立っている位置に最も近い点であるからである。

本研究では、人物の立っている実世界の座標を(X, Y) [m]と定義し、人物検出モデルで検出された人物の足元位置のピクセル座標を(px, py) [pixel]と定義する。さらに、我々が構築を目指すシステムで推定された人物の立っている座標を(X', Y') [m]と定義する。

まず、カメラで複数人を検出する必要があるため、人物検出モデルを用いる。そして、検出された人物領域から足元位置の画素座標(px, py)を算出し、これを基に人物の位置を推定する。しかし、カメラ映像上の画素座標である(px, py)はカメラ座標系に基づくものであり、実際の空間上の位置を直接表すものではない。そこで、足元の画素座標(px, py)を実世界の座標(X, Y)へ変換する処理が必要となる。

カメラの映像から実世界の座標を推定する手法として、本研究では以下の 3 つの方法を検討する(図 5)。

1. 射影変換
2. 深度センサ
3. 深度推定 AI

以上の 3 つの手法を用いて得られた実世界の座標(X, Y)をもとに、人物分布を推定する。

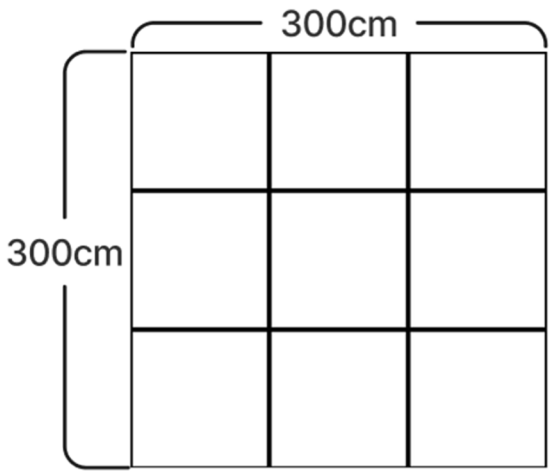

図 3　実験領域のイメージ

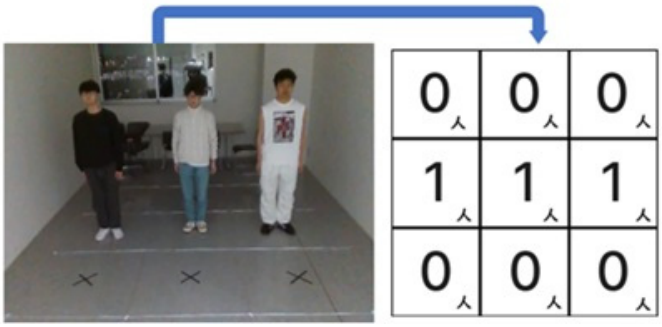

図 4　人数分布推定システムの完成イメージ

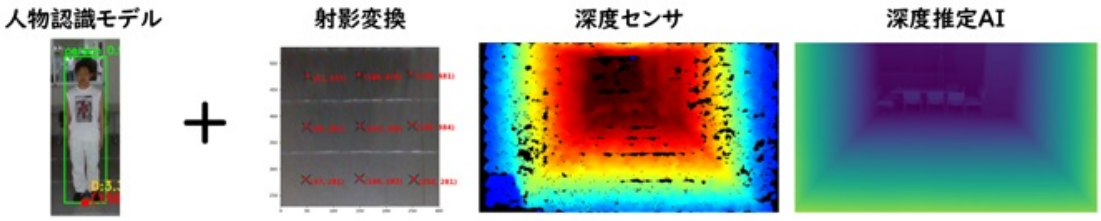

図 5　深度推定システムの流れ

2.2 複数の人物を検出するモデル（大城）

我々は、複数人物を検出する方法として、物体検出モデル RF-DETR および骨格推定モデル Lightweight OpenPose の 2 つを検討した。

RF-DETR は roboflow により開発された、リアルタイムに動作する物体検出モデルである(図 6)。Lightweight OpenPose は 2016 年に Daniil Osokin により開発された、ディープランニングを用いて、動画や静止画に映る人の Keypoints (図 8)を検出し、ポーズを推定する骨格推定モデルである(図 7)。

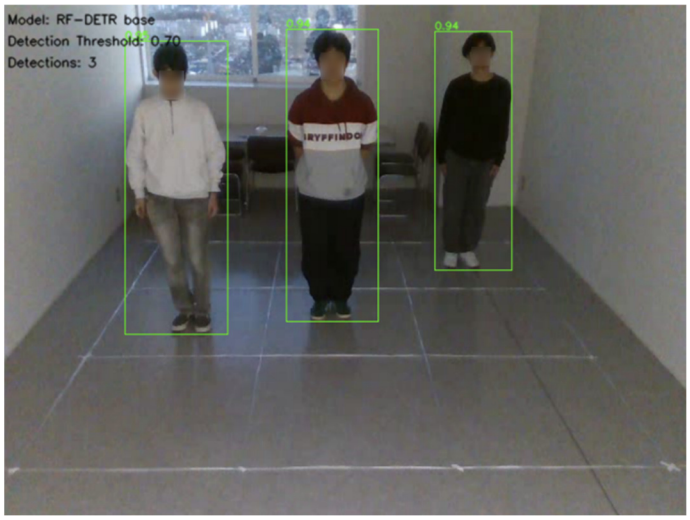

図 6 RF-DETR による人物検出

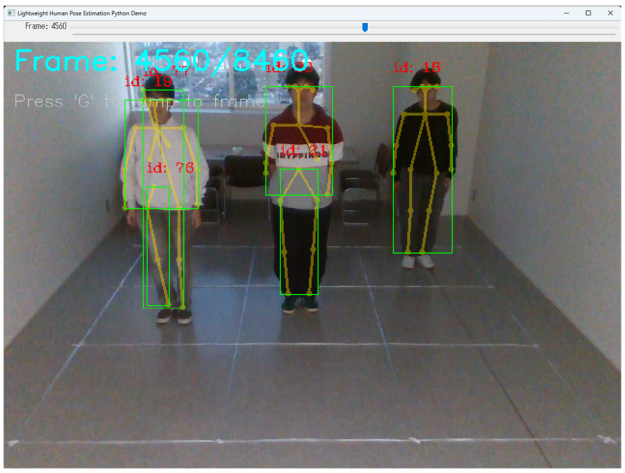

図 7　Lightweight OpenPose による骨格推定

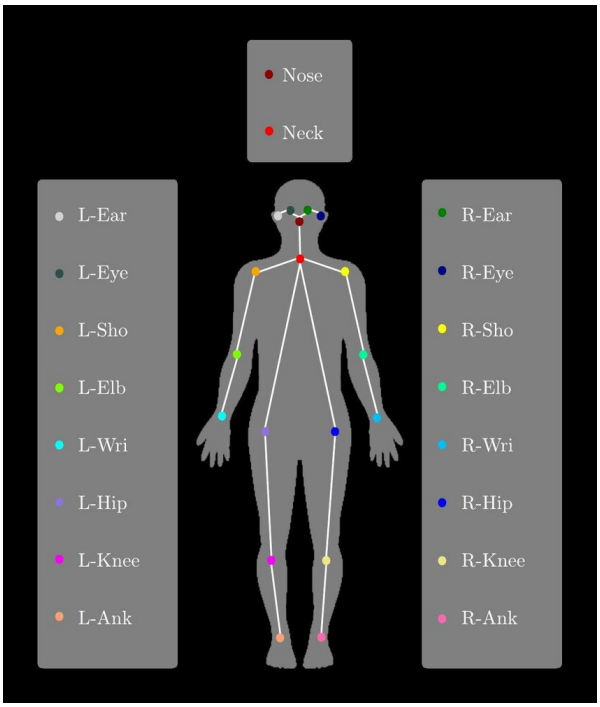

図 8 Lightweight OpenPose で取得される Keypoints

Lightweight OpenPose を用いた場合、図 8 の L-Ank と R-Ank の Keypoint を基準に深度を取得することになる。しかし、対象人物の遮蔽状況によって、指定した Keypoint が正確に検出されない場合があり、深度取得のための、次節で説明する足元位置の定義が困難となる。

これに対し、物体検出モデルである RF-DETR ではバウンディングボックス(以下 BBox)を検出結果として出力する。物体検出モデルの特徴として、体の一部しか写っていなくてもその一部を含む BBox を出力できることが挙げられる。

以下、RF-DETR と Lightweight OpenPose とによる検出の特徴をいくつか紹介する。

図 9 と図 10 では重なりのある 2 人の人物の検出を行っている。RF-DETR の結果を示す図 9 では BBox（図中の人物を囲う、緑の四角）が概ね正しく表示されているが、Lightweight OpenPose の結果を示す図 10 では、奥の人物の BBox の横幅が実際より長く表示されており、図 8 中の L-Ank と R-Ank に相当する Keypoint の高さが異なって検出されている。

図 11 と図 12 では重なりのある 3 人の人物の検出を行っている。RF-DETR の結果を示す図 11 では、手前の人物と奥の人物に挟まれている 2 番目の人物の BBox が表示されておらず、3 番目の人物の BBox は胸の高さまでの範囲しかない。Lightweight OpenPose の結

果を示す図 12 では、手前の人物に対して、体全体の BBox と右足のみを含む BBox の 2 つが表示されてしまっている。

図 13 と図 14 でも重なりのある 3 人の人物の検出を行っている。RF-DETR の結果を示す図 13 では正しく 3 人の人物が正しく検出されているが、Lightweight OpenPose の結果を示す図 14 では推定された骨格が乱れており、BBox が 4 つ表示されてしまっている。

このように、骨格検出モデルは人物同士の重なりの影響を受けやすく、骨格点の検出や BBox の表示が不安定となる問題があるため、我々は、本研究において、複数人物検出モデルに RF-DETR を用いることにした。

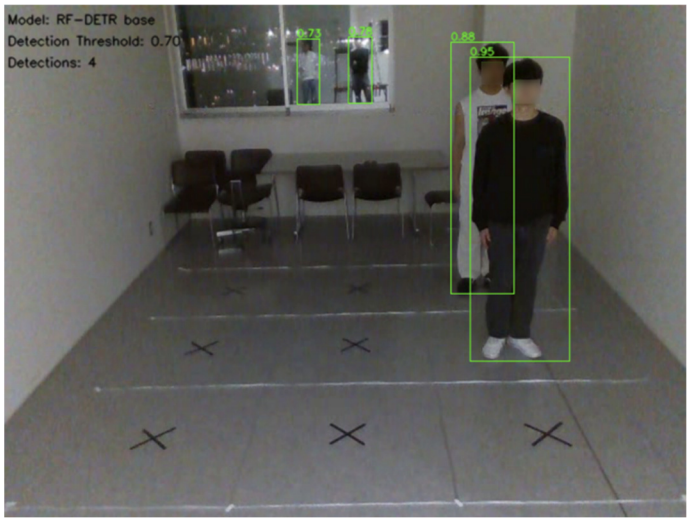

図 9　RF-DETR　検出例 1

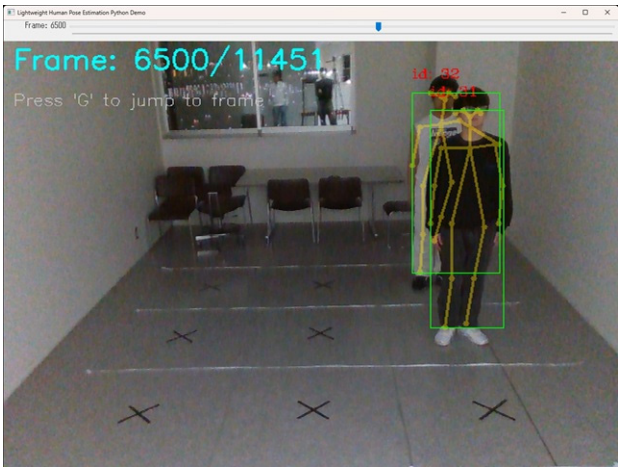

図 10　Lightweight OpenPose　検出例 1

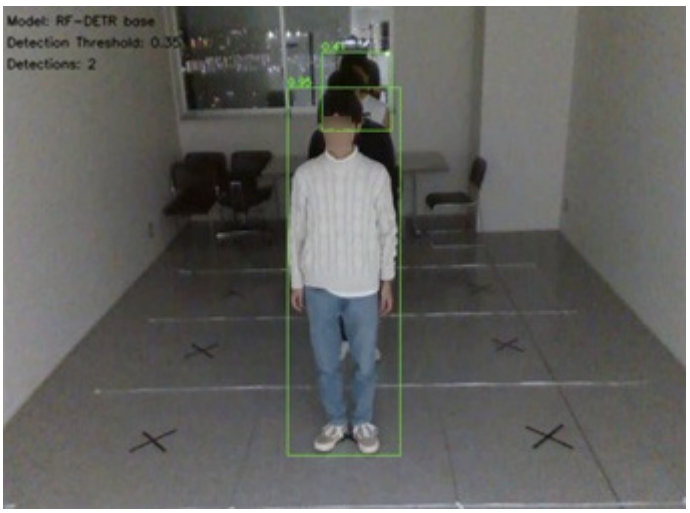

図 11　RF-DETR　検出例 2

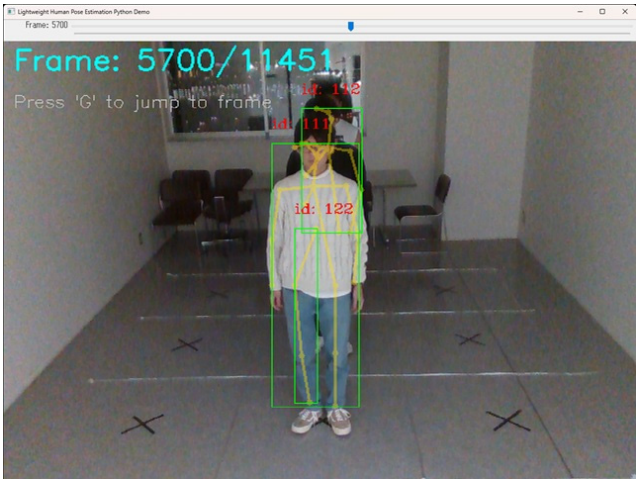

図 12　Lightweight OpenPose　検出例 2

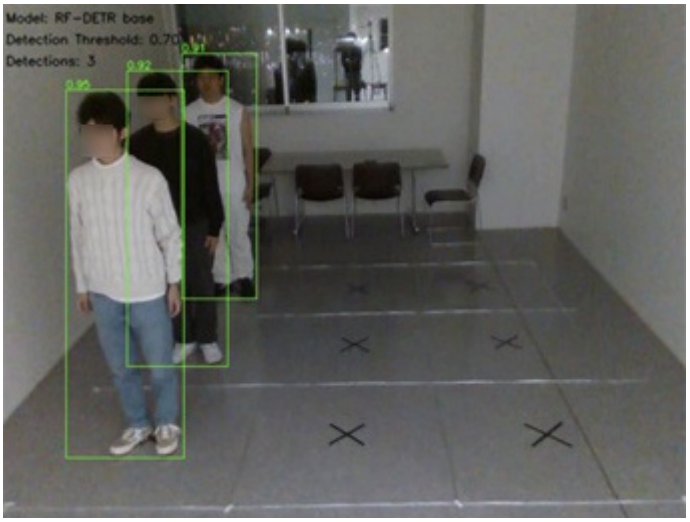

図 13　RF-DETR　検出例 3

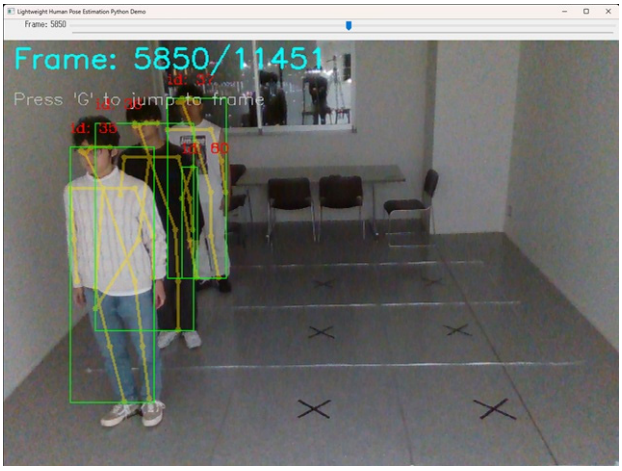

図 14　Lightweight OpenPose　検出例 3

2.3 足元位置の定義

本研究では、足元位置の定義をその人が立っている位置と定義する。足元位置の点を RF-DETR で検出された人物の BBox の下辺の中心とし、そのピクセル座標(px, py)を得る（図 15)。ピクセル座標(px, py)を実世界の座標(X', Y')に変換する際、px から X'への変換には、後述する射影変換を用いる。py から Y'の変換には射影変換、深度センサ、深度推定 AI を用い、それぞれの精度を比較する。その詳細は 2.4 節で説明する。

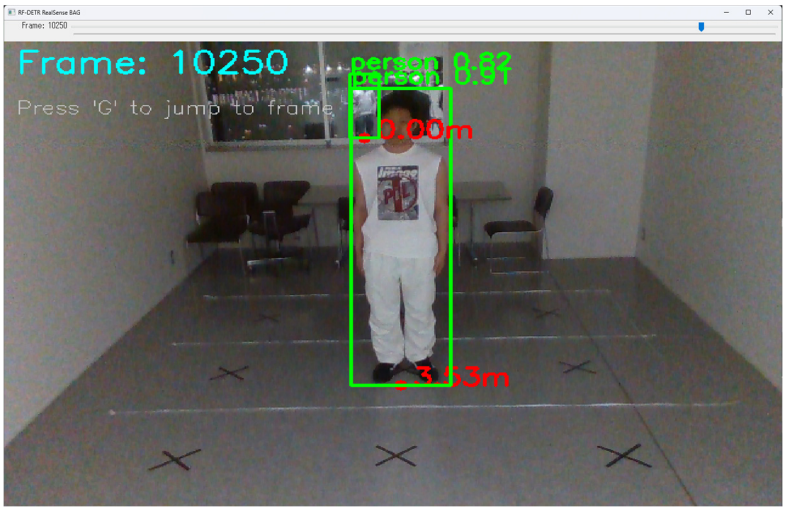

図 15　足元位置の例

2.4 深度推定手法

本節では、3 つの深度推定手法(射影変換、深度センサ、深度推定 AI)について説明する。

2.4.1 射影変換

本研究で用いる射影変換とは、カメラで撮影した画像上の位置を、実際の平面上の位置に対応させるための変換である。

射影変換を用いることで、画像中の人物の位置を、実世界の床面上の位置に変換することができる(図 16)。これにより、カメラの設置位置や角度が異なっていても人物の位置を同じ基準で比較できるようになる。また、事前に領域の寸法が分かっていれば、射影変換後の座標に実寸法を割り当てておくことで、実環境の値を出すことができる。

ピクセル座標(px, py)に対応する実世界の座標を(X, Y)とする。(px, py)から(X, Y)への変換は以下の射影変換行列の計算によって求めることができる。H は射影変換行列を表している。

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = H \begin{bmatrix} px \\ py \\ 1 \end{bmatrix} = \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} px \\ py \\ 1 \end{bmatrix}$$

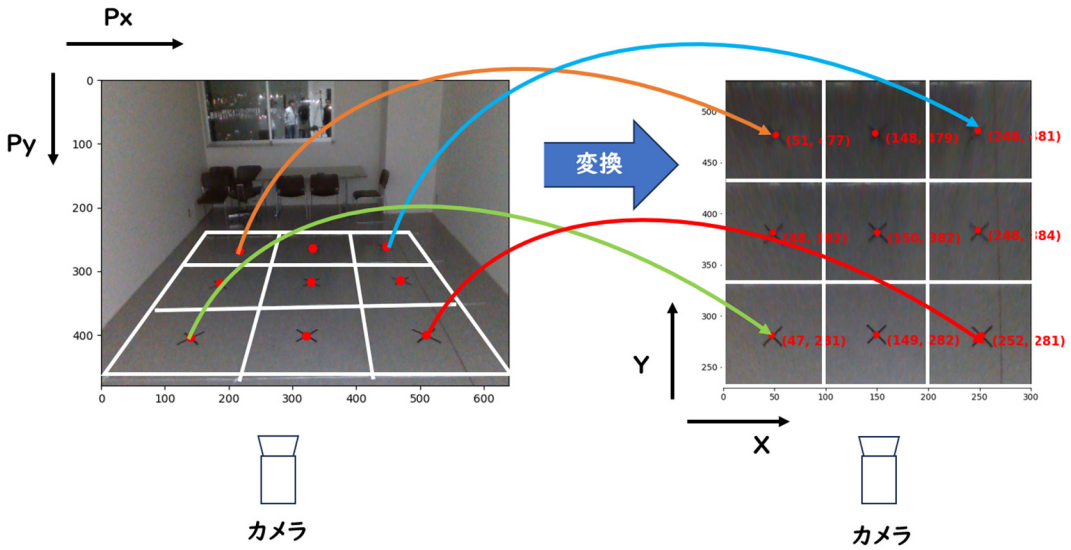

図 16　射影変換

2.4.2 深度センサ

本研究では、深度センサとして、RealSense デプスカメラ D435（図 17）を使用する。

深度センサとは、ステレオカメラ方式で距離(奥行き)を測ることができるセンサである。

ステレオカメラ方式とは 2 つのカメラを用いて対象物を複数の異なる方向から同時に撮影し、カメラの画素の位置情報から奥行き方向の情報も計測することが可能なカメラである(図 18）。

図 18 は深度センサを使用した時のイメージ図である。右に写っている 2 つの画面は写し

ている景色は一緒だが、右側の画像は RGB 画像という最も基本的な形式で表したデジタル画像である。左に写っているのが Depth 画像というカメラから被写体までの距離情報を画素置として可視化した画像である。深度が小さいと寒色であらわされ、深度が大きいと暖色であらわされる。

後の 3.4 節で説明する実験環境で、照明をつけている場合とつけていない場合で撮影を行った結果、図 19 と図 20 より、照明をつけている場合において、図 19 左側の Depth 画像において、深度が取得できていない(図 19 の黒い部分)箇所が多く見られた。これは、蛍光灯の人工照明に含まれる赤外線成分や、床面での反射光が、RealSense の深度計測に影響を与え、深度が正しく取得できなかった可能性がある。そのため、本研究では、照明をつけていない状況において 3 章と 5 章の実験を行っている。

なお、本研究で用いる深度センサの仕様を表 1 に示す[7]。

図 17　RealSense デプスカメラ D435

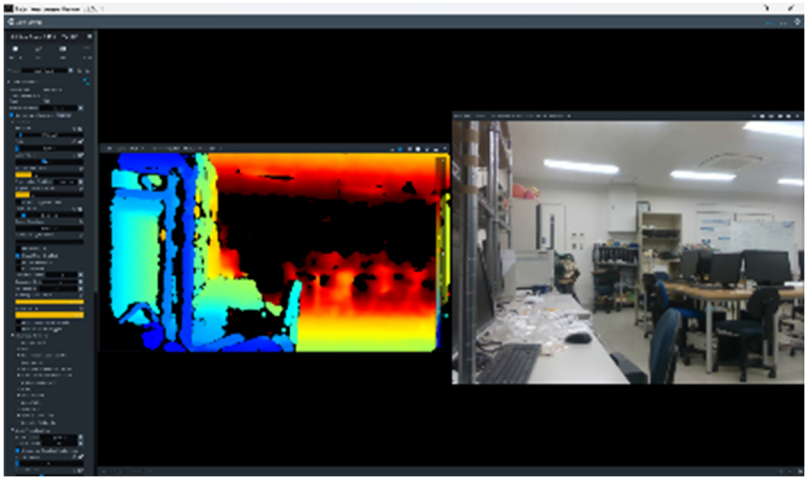

図 18　深度センサの使用イメージ

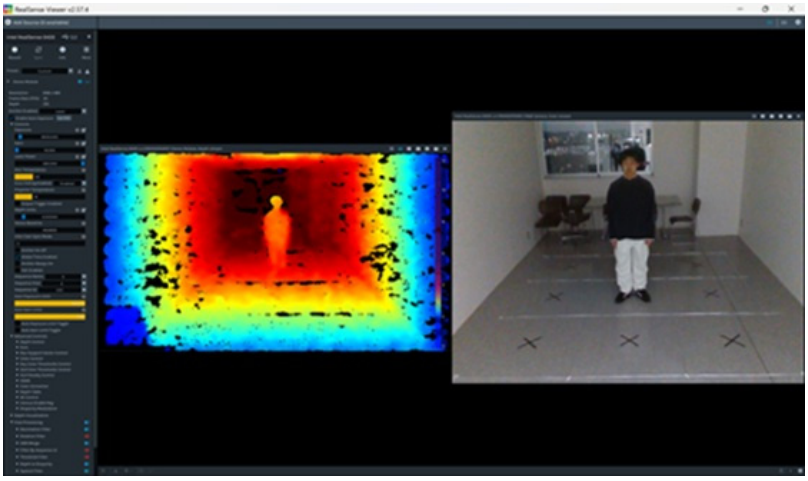

図 19　照明をつけている場合

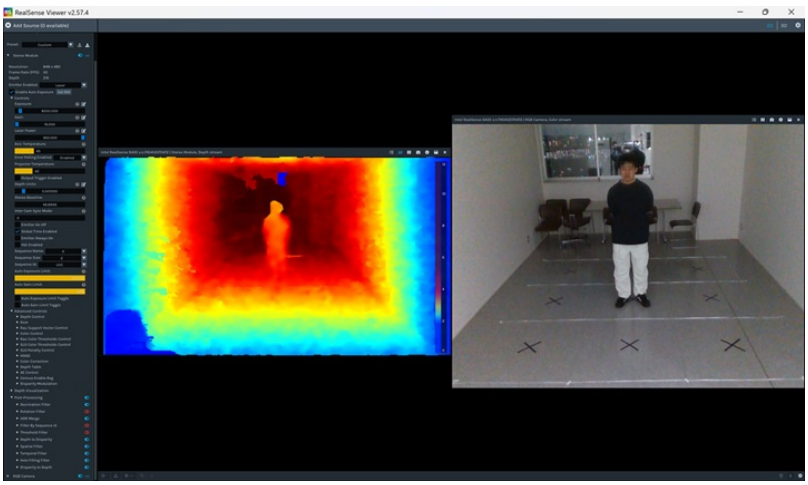

図 20　照明をつけていない場合

表 1　深度センサの仕様

名称	REALSESE™ DEPTH CAMERA D435
方式	ステレオビジョン方式
推奨動作範囲	0.3m~3.0m

2.4.3 深度推定 AI（佐藤）

本研究では深度推定 AI として、Apple の単眼深度推定モデルである Depth Pro を使用する。Depth Pro は、RGB 画像・映像を入力として、入力データの深度を推定する事前学習済みモデルである(図 21)。

最適な深度推定には、RGB 画像・映像に加えて、撮影に用いたカメラのピクセル単位の焦点距離f_{px}をパラメータとして与える必要がある。f_{px}は実距離とピクセルの対応を決めるスケーリングのための係数であり、この値によって推定される深度の精度が変わる。設定値が実際のカメラのf_{px}から離れるほど、推定深度と実際の深度の差は大きくなる。ただし、カメラの持つ実際のf_{px}を入力した場合でも、推定値が実測値と完全に一致するわけではなく、モデル本来の性能や特性による誤差は残る。

f_{px}を意図的に与えない場合でも、入力映像のメタデータにf_{px}が含まれていればその値が読み込まれるが、そこにもない場合には Depth Pro が内部的にf_{px}も推定し、その値を用いて深度推定まで行われる。ただし、それによって得られる深度は、推定に次ぐ推定によって算出された値になってしまうため、純粋な Depth Pro の性能を評価することができない。のちに 4 章で述べる重なり対応のシステムでは、深度取得において必ずしも真値に近い深度値を得られることを求めないため、そこにおいてはピクセル単位の焦点距離の値は重要ではない。しかし、本章では Depth Pro 自体の性能に、精度の結果がそのまま影響するため、誤差要因を減らすためにも実際のピクセル単位の焦点距離を用いるのが適している。

本研究ではカメラに RealSense D435 を用いた。RealSense SDK 2.0 の tools ディレクトリで”rs-enumerate-devices -c”を実行し、出力されるデバイス情報からこのカメラのピクセル単位の焦点距離 616[px]を取得して用いた。RealSense D435 は研究用途で使われることが多く、SDK などを通じて内部パラメータを確認できる。一方、一般的なカメラでは内部パラメータを確認できない場合がある。その場合、焦点距離f_{px}、解像度H_{px}、イメージセンサの実寸法S_{mm}から算出する必要がある。例えば高さ方向を基準にする場合、次式で近似できる。

$$f_{px} \approx f_{mm} \times \frac{H_{px}}{S_{mm}}$$

ここで、f_{mm}は焦点距離[mm]、H_{px}は画像の高さ[px]、S_{mm}はセンサ高さ[mm]である。RealSense D435 のハードウェアパラメータをこの式に当てはめると、RGB カメラ解像度の高さが 480[px]、焦点距離が 1.93[mm]、センサ高さが 1.5498[mm]なので[6]、

$$f_{\mathrm{px}} \approx 1.93 \times \frac{480}{1.5498} = 597.75 \approx 598\ [\mathrm{px}]$$

となる。このように算出できる一方で、この値は推定値であり、実際の内部パラメータとは一致していない。実際に本研究で SDK から取得した 616[px]と比べると差が 18[px]あり、内部パラメータを確認できないカメラを使用する際、推論値のピクセル単位の焦点距離を使用する場合は、この差が深度推定の誤差要因になる点に留意する必要がある。

図 21　深度推定 AI の使用イメージ

2.5 取得された深度の補正

2.4.2 項と 2.4.3 項で取得された深度は、我々が必要な深度と座標系が異なる。

2.4.2 項と 2.4.3 項で取得される深度は、カメラから対象物までの直線距離であり、図 22 より、Zを表している。本研究で必要な深度は、カメラ設置点から、対象物が立っている位置までの水平距離は、図 22 よりdである。よって、取得された深度の補正が必要となる。

我々が必要としている深度をd、カメラの高さをh、取得される深度をZ、カメラの俯角をθとすると、以下の式によって我々が必要としている深度を求めることができる(図 22)。

$$d = \frac{Z - h\sin\theta}{\cos\theta}$$

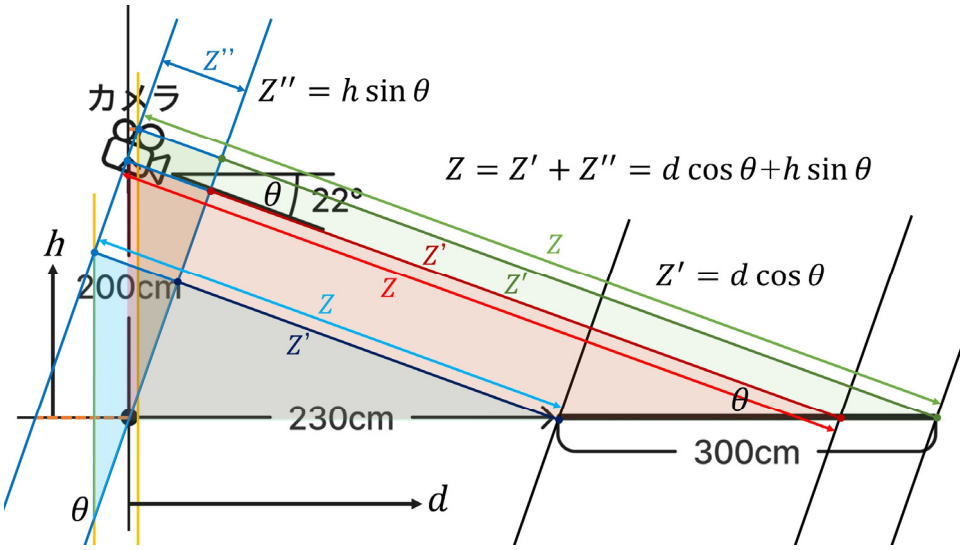

図 22　深度の補正式

第 3 章 深度取得方法の性能比較実験（米原）

3.1 実験目的

2 章で先述したように、本研究では、実世界における人物の位置を(X, Y)で表すこととし、Y を深度として扱う。深度の算出には、射影変換、深度センサ、深度推定 AI の 3 つの手法を用いる。

本章では、これらの 3 つの手法について、深度取得精度の比較を行い、どの手法がより正確な深度推定が可能であるかを検討する。また、精度に加えて、深度の取得にかかる時間も評価する。

3.2 実験方法

深度取得方法の性能比較実験では、床面上の各測定点に人物を配置し、各測定点における実際の奥行き方向の座標を実測値 Y と定義する。人物の配置には 17 種類のパターンを設定し、人物同士の重なりの有無による条件を含めた。なお、本研究における「重なり」とは、図 29 や、図 30 のように、同一縦列上に 2 人以上が配置されている状態と定義する。各パターンにおいて、射影変換、深度センサ、深度推定 AI の 3 つの手法により深度 Y'を算出し、Y と Y'の関係をグラフ化することで、深度取得の精度を評価する。

人物の深度取得精度および深度の取得にかかる時間を評価するため、あらかじめ決められた人物の配置パターンを含む動画を撮影し、取得した映像の全フレームに対して解析処理を行った。ここで、「フレーム」とは、映像を構成する 1 枚 1 枚の静止画像を指す。また、ここでは、RGB 画像と深度画像の両方を取得可能な深度センサをカメラ機能の代用とし、実験映像の撮影を行った。撮影された映像は BAG ファイルとして保存されており、このファイルには、RGB 画像および深度情報の両方が含まれている。

各深度取得方法において参照する画像情報は異なり、射影変換および深度推定 AI では BAG ファイルから抽出した RGB 画像を用いて深度の取得を行った。一方、深度センサによる深度取得では、深度情報を直接参照した。また、人物検出においては、すべての手法において RGB 画像を用いて行った。

実験領域は 300cm×300cm の正方形領域とし、これを縦横ともに 3 分割した 9 つのエリアに分ける。各エリアの中心点に印をつけ、人物をこれらの測定点に配置し、領域内に 2 人

または 3 人が立つ状況を想定した。図 23 が実験領域のイメージ図であり、300cm×300cm の実験領域を表し、縦横ともに 3 分割した 9 つのエリアの中心点を青の×印で示しており、この点に人物を配置する。すべてのパターンを連続して撮影した動画を取得し、その中からあらかじめ定めた 17 パターンに対応するフレームを評価に使用した。各配置パターンを図 24～図 40 に示す。

配置条件は、人物同士の重なりが生じない場合（図 24～図 28）と、人物同士の重なりが生じる場合（図 29～図 40）の 2 種類に分類した。

取得した動画の各フレームに対して、2 章で選定した RF-DETR を用いて人物検出を行い、各人物の足元のピクセル座標(px, py)を取得する。それをもとに射影変換、深度センサ、深度推定 AI を用いて深度 Y'を取得した。得られた深度 Y'を図 24～図 40 までの各配置パターンに対応付けて整理し、事前に測定した実測値 Y と比較した。さらに、Y を横軸、Y'を縦軸とした散布図を作成する。このとき、測定値 Y'が実測値 Y と一致する理想的な状態では、すべての点が Y=Y'の直線上に分布する。したがって、各手法による推定結果がこの直線に近いほど、深度取得の精度が高いことを示す。図 41 は、深度取得精度を評価するために描くグラフの例であり、横軸が Y、縦軸が Y'、グラフ上の青で描かれている直線が Y=Y'である。また、本研究では、実験領域内における人物の配置位置による深度取得精度への影響を検討するため、実験領域の左列で重なった場合（図 31、図 32、図 37)、中央列で重なった場合（図 30、図 33、図 36)、右列で重なった場合（図 29、図 34、図 35）における深度取得精度の違いについても比較を行う。さらに、人物検出モデルで足元座標を取得し、各手法によって深度 Y'を取得する処理を動画の全フレームに行ったときにかかった時間を計測し、総フレーム数で割ることで、1 フレームあたりの平均処理時間を算出し、これも評価に用いる。

実験手順を以下に示す。

＜実験手順＞

1. すべての配置パターン（図 24～図 40）を含む動画を撮影する。
2. 取得した動画の全フレームに対して人物検出および深度取得を行う。
3. 各フレームの結果を配置パターンごとに整理する。
4. 取得した深度 Y'を実測値（事前に測定した図 23 中の青色×印の Y の値）と比較し、深度取得の精度を評価する。
5. 総処理時間を総フレーム数で割り、1 フレームあたりの平均処理時間を算出する。

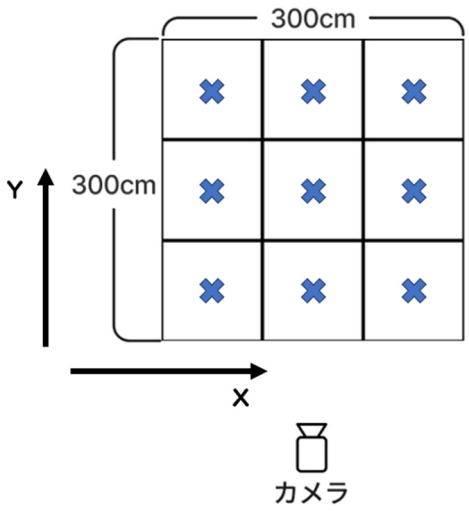

図 23　実験領域のイメージ

図 24　パターン 1

図 25　パターン 2

図 26　パターン 3

図 27　パターン 4

図 28　パターン 5

図 29　パターン 6

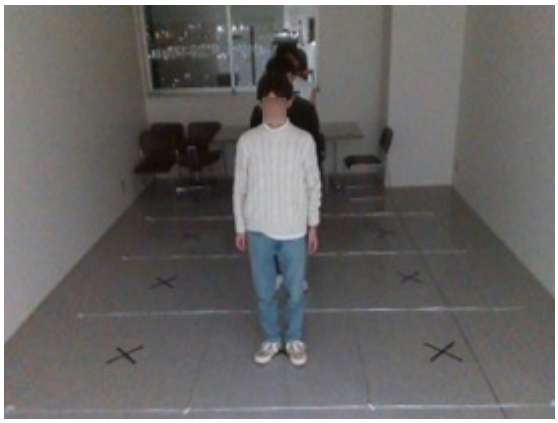

図 30　パターン 7

図 31　パターン 8

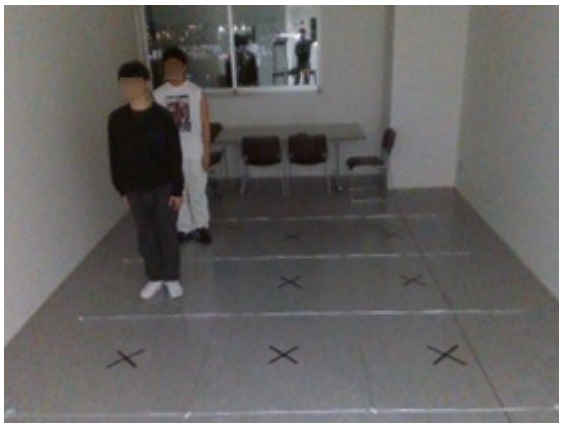

図 32　パターン 9

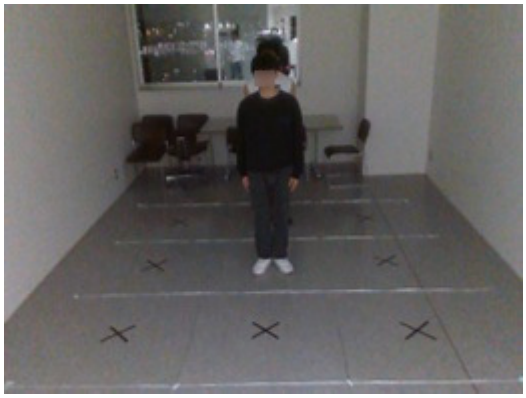

図 33　パターン 10

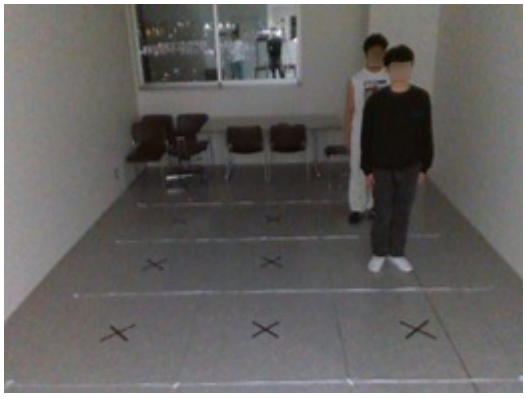

図 34　パターン 11

図 35　パターン 12

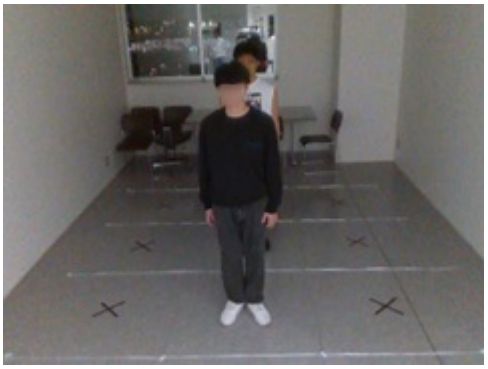

図 36　パターン 13

図 37　パターン 14

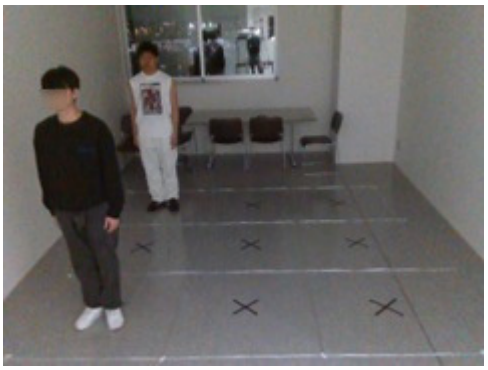

図 38　パターン 15

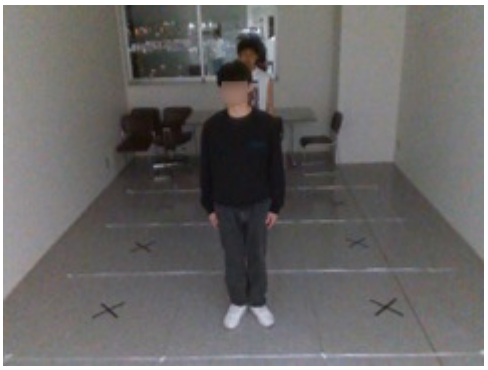

図 39　パターン 16

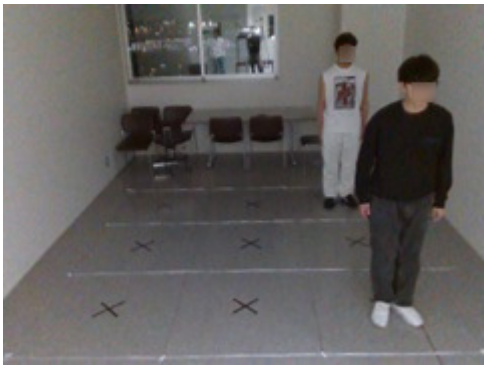

図 40　パターン 17

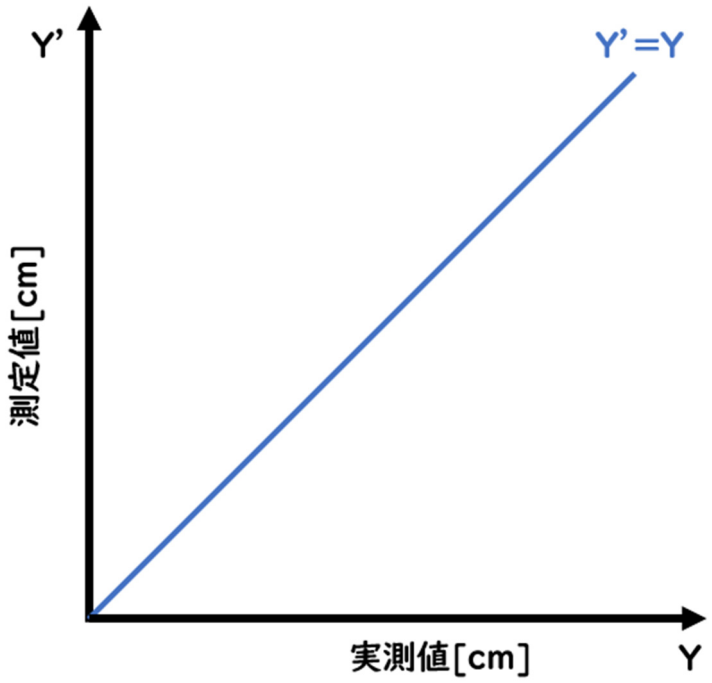

図 41　深度取得精度の評価に用いるグラフの例

3.3 使用機器

本節では、実験で使用した機器の説明をする。

3.3.1 実験映像録画用 PC

実験映像を録画するパソコンとして、iiyama 社製 PC を使用した（図 42)。詳しい仕様を表 2 に示す。

図 42　使用した PC

表 2　録画用 PC の仕様

製品名	Iiyama IStNEs-15FR171-i7-UASXB
OS	Microsoft Windows 11 Pro
CPU	12th Gen Intel Core i7 – 12700H
GPU	NVIDIA GeForce RTX 3070 Ti
メモリ	16GB

3.3.2 カメラ機能として代用した深度センサ

図 43 は、本実験に使用したカメラとして代用した深度センサである。深度情報と RGB 画像の両方を取得する機能がある。そのうち、RGB 画像を通常のカメラとして代用する。

仕様については、2.4.2 項にて先述している。

図 43　カメラ機能として代用した深度センサ

3.3.3 深度取得システム実行用 GPU 搭載 PC

映像に対して人物検出を行い、取得された足元座標(px, py)に射影変換、深度センサ、深度推定 AI を用いて深度 Y'を取得する処理を、表 3 の GPU 搭載 PC 上で実行する。

表 3　プログラム実行用 PC の仕様

OS	Microsoft Windows 11 Pro
CPU	12th Gen intel Core i7 – 12700
GPU	NVIDIA GeForce RTX 3060
メモリ	12GB

3.4 実験環境

3.2 節で解説した 300cm×300cm の実験領域から 230cm の位置にカメラを設置する（図 44）。

図の青い×印の位置に人物を配置するため、正しく深度 Y'の取得が行われている場合、カメラに近い側から、一列目であれば 280cm、2 列目であれば 380cm、3 列目であれば 480cm が得られる。

実験は、八王子キャンパスの 4 号館 8 階で行う（図 45）。カメラの設置条件は、高さ 200cm、俯角 22° とする（図 46）。

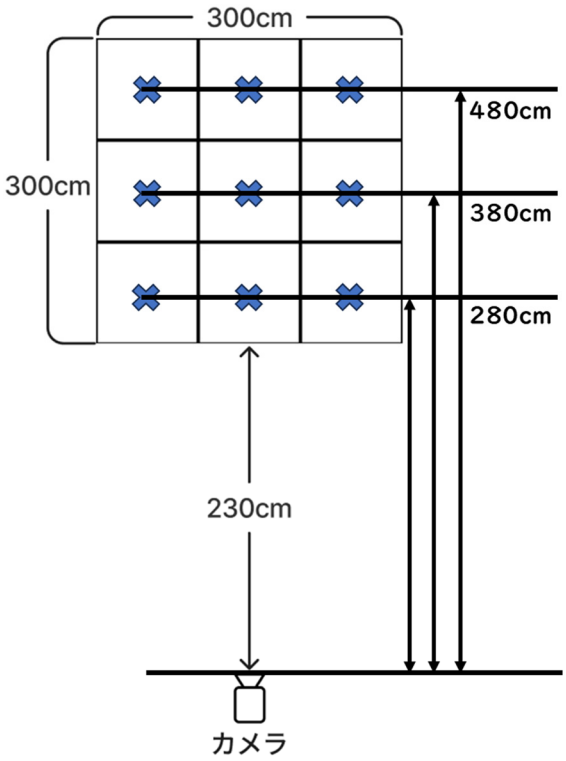

図 44　実験環境を上から見た図

図 45　八王子キャンパス 8 階

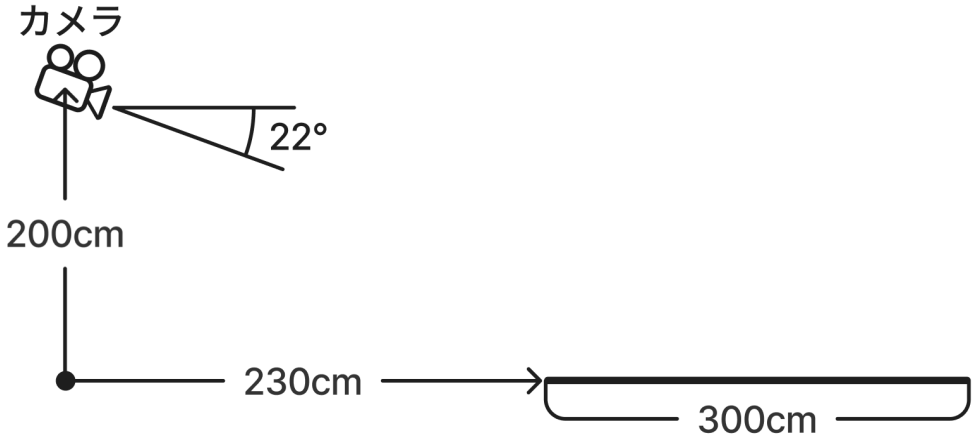

図 46　実験環境を横から見た図

3.5 評価方法

本節では、本実験の評価方法について説明する。

3.5.1 深度取得精度

深度取得精度とは、どの程度正しく人物の足元位置を推定できているかを表す。

3.4 節において、テープを貼った位置を実際の深度 Y とし、2 章で紹介した 3 つの深度取得方法で取得された深度 Y'が実際の足元位置とどの程度離れているのかを検証する。また、カメラからの距離が離れるほど、取得結果にどのような傾向がみられるのかも確認する。さ

らに、3.2 節でも先述したように、人物が配置された列ごとに推定結果を分類し、領域内の左列で重なった場合（図 31、図 32、図 37)、中央列で重なった場合（図 30、図 33、図 36)、右列で重なった場合（図 29、図 34、図 35）の深度取得精度に与える影響を検証する。

3.5.2 平均処理時間

平均処理時間とは、1 フレームあたりにかかる処理時間のことを表す。

人物検出を行い、取得された足元座標(px, py)を射影変換、深度センサ、深度推定 AI を用いて深度 Y'を取得するまでにかかった処理時間を総フレーム数で割ることによって、平均処理時間を導く。

3.6 実験結果

本節では、3 つの深度取得方法の深度取得精度と平均処理時間を比較実験した結果について説明する。

3.6.1 重なりがない場合の深度取得精度

本項では、重なりがない場合における各深度取得方法の深度取得精度を比較する。

図 47 は、領域内で人物の重なりがない場合における各深度取得方法で得た深度 Y'の実測値 Y との比較を示している。

図 47 より、重なりがない場合では、射影変換が実測値との誤差が最も少なく、深度取得精度が高いことがわかった。さらに、射影変換であれば、カメラからの距離に関係なく安定して深度の取得ができることがわかった。一方、深度センサと深度推定 AI に関しては、カメラからの距離が遠くなるほど実測値からの誤差が大きくなっていることがわかる。これは、深度センサの場合、動作推奨範囲（2.4.2 項にて先述）を大きく逸脱していることが原因であり、深度推定 AI の場合、深度推定 AI のカメラから離れれば離れるほど測定値が実測値から遠ざかってしまう特性が原因である。

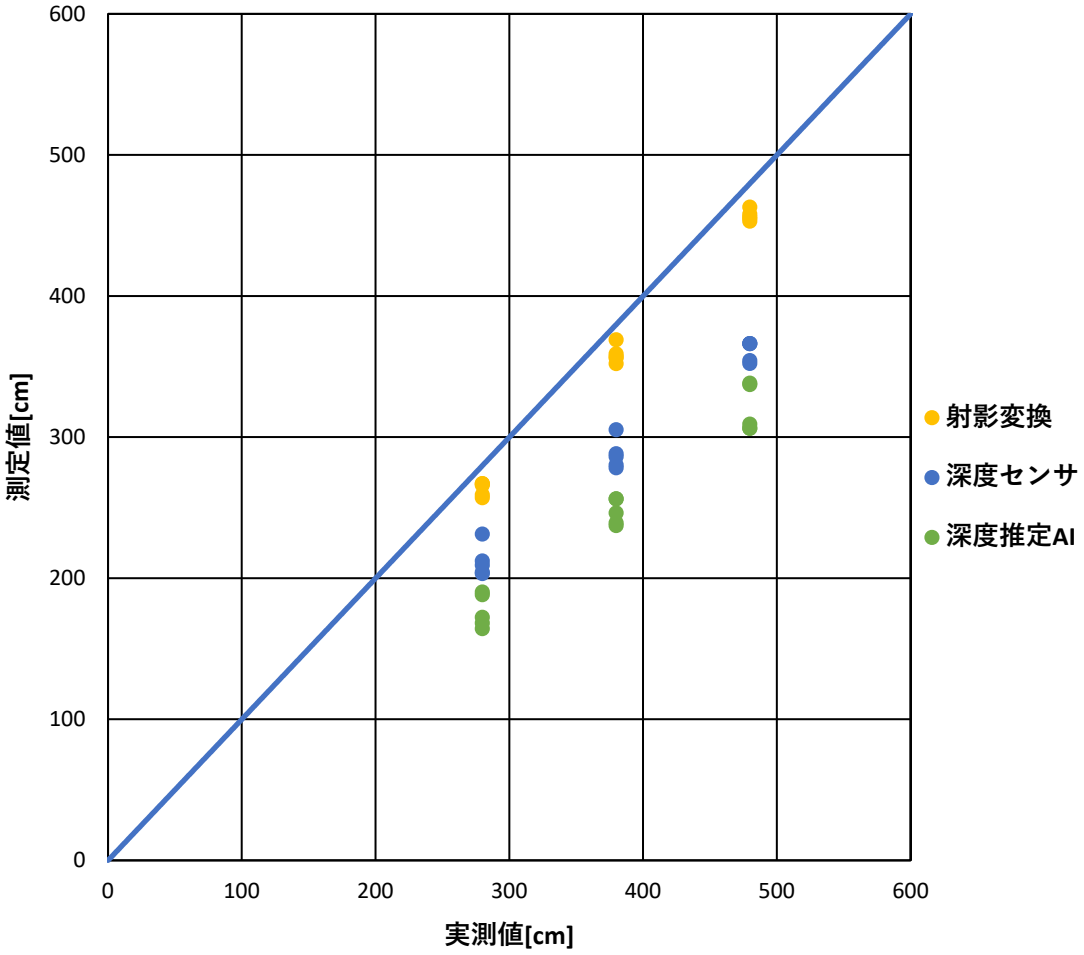

図 47　重なりがない場合

3.6.2 複数の人物に重なりがある場合の深度取得精度

本項では、複数の人物に重なりがある場合における各深度取得方法の深度取得精度を比較する。

図 48 は、領域内で複数の人物に重なりがある場合における各深度取得方法で得た深度 Y' の実測値 Y との比較を示している。

図 48 より、射影変換では、実測値が 280cm の地点においては、取得された深度が、重なりがない場合と比較してわずかにばらつきが大きくなっていることがわかった。実測値が 380cm の地点においては、取得された深度が、重なりがない場合と比較して、多くの測定値においてばらつきが抑えられているように見える。しかし、取得された深度が、実測値と大きく離れてしまう場合があることがわかった。実測値が 480cm の地点においては、取得された深度が、重なりがない場合と比較して、多くの測定値についてはあまり変わらない結果であったが、380cm の地点と同様に、取得された深度が、実測値と大きく離れてしまう場合があることがわかった。

図 48 より、深度センサでは、実測値が 280cm と 380cm の地点では、重なりがない場合の深度取得精度とあまり変わらない結果であったが、480cm の地点でばらつきが大きくなっていることがわかる。これは、足元位置を 2.3 節にて固定しているため、人物検出自体は正しく行えていても、重なっていることで、より手前の人物の深度情報を参照してしまうことが原因である。

図 48 より、深度推定 AI では、実測値が 280cm の地点では、重なりがない場合の深度取得精度とほぼ変わらない結果であったが、380cm の地点では、ばらつきが大きくなっており、480cm の地点では、取得された深度が、実測値と大きく離れてしまう場合があることがわかった。

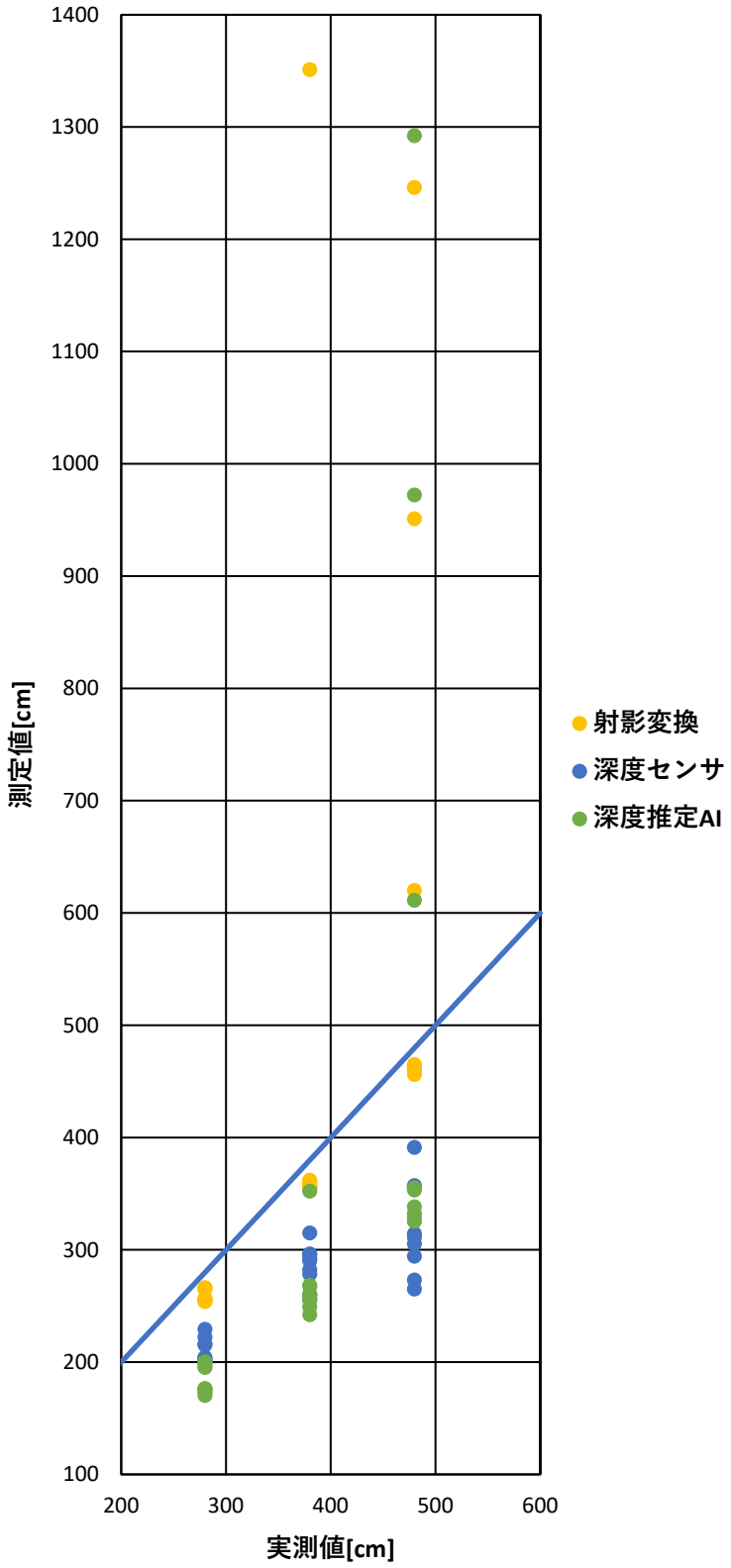

図 48　重なりがある場合

3.6.3 複数の人物の位置による深度取得精度の比較

本項では、複数の人物を左列、中央列、右列に配置したパターンの深度取得精度を比較する。

実験結果を図 49、図 50、図 51 に示す。これらの図より、中央列に人物を配置した際に、射影変換と深度推定 AI において、取得された深度が実測値と大きく離れてしまうことがわかる。これは、図 52、図 53、図 54 に示したように、RF-DETR で人物検出を行った際に、中央列に複数の人物を配置した場合のみ BBox が正しく表示されていない（図 53 中の最も奥の人物）ことが原因であると考えられる。

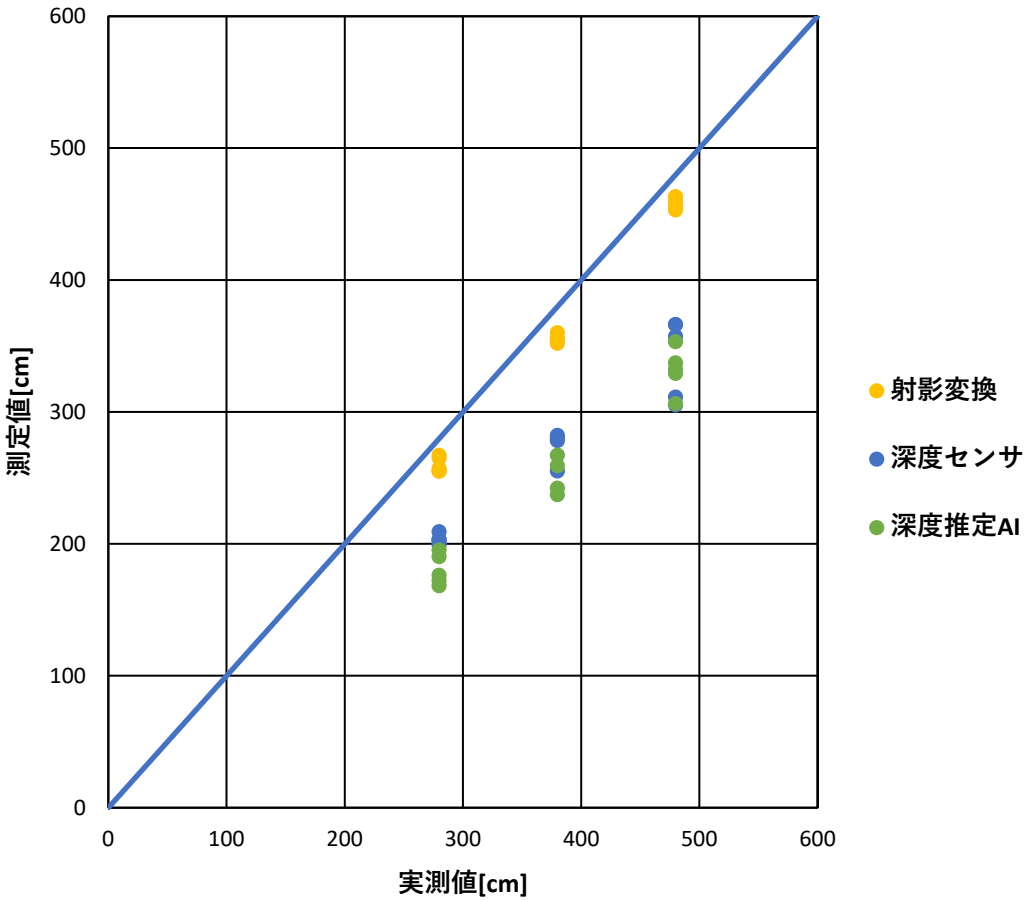

図 49　複数の人物が実験領域の左列で重なった場合の深度の比較

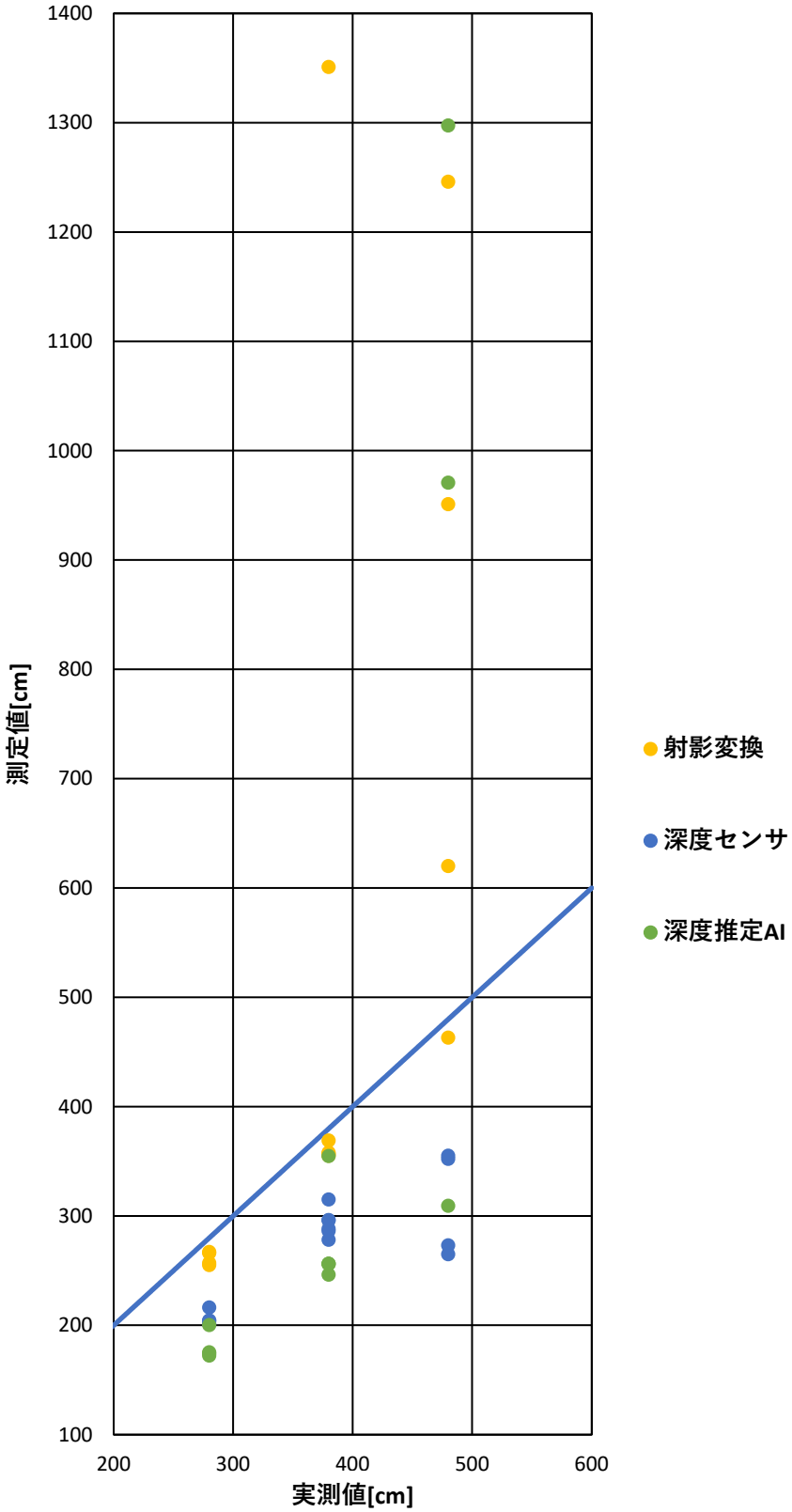

図 50　複数の人物が実験領域の中央列で重なった場合の深度の比較

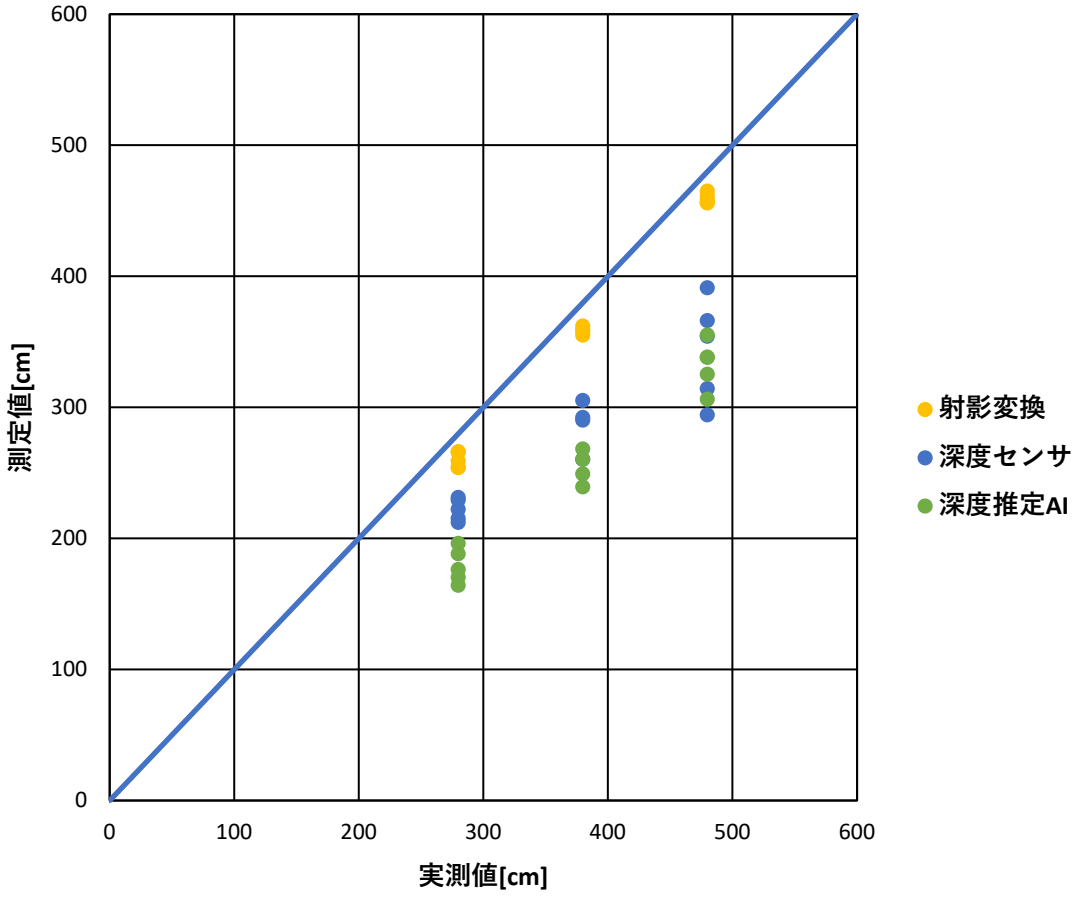

図 51　複数の人物が実験領域の右列で重なった場合の深度の比較

図 52　複数の人物を左列に配置したパターン

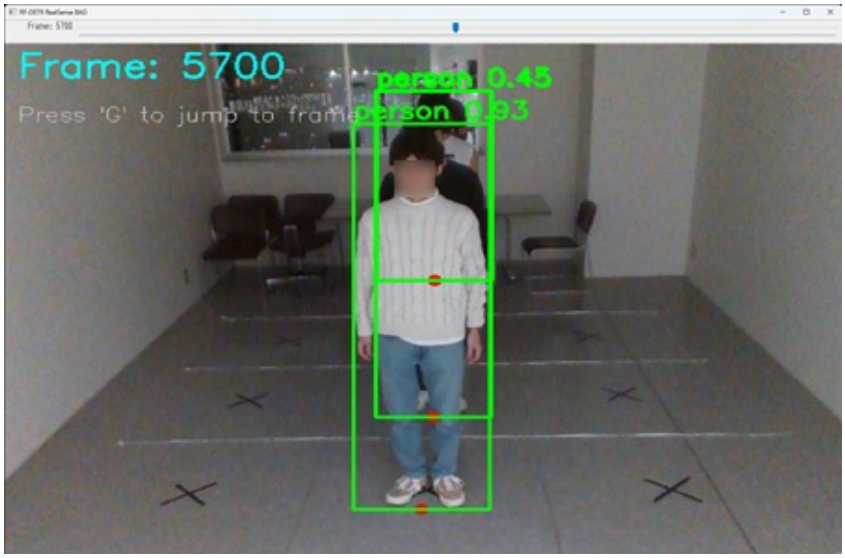

図 53　複数の人物を中央列に配置したパターン

図 54　複数の人物を右列に配置したパターン

3.6.4 各深度取得方法の平均処理時間

本項では、各深度取得方法の平均処理時間について比較する。

表 4 より、平均処理時間については、射影変換と深度センサがほぼ同じであり、深度推定 AI が他 2 つの深度取得方法と比べて約 40 倍の時間がかかることがわかった。

なお、表 4 では、複数人物検出モデルの推論時間の 25.2ms/Frame を除いた時間を記した。

表 4　平均処理時間

射影変換	3.6ms/Frame
深度センサ	3.3ms/Frame
深度推定 AI	119.8ms/Frame

第 4 章 重なりに対応するシステムの構築（佐藤）

第 3 章でわかった重なりの問題は、深度取得点を BBox の下辺の中点に固定したことが原因であると考えた。それに対し本章では、頭や上半身など人物の見えている部分の深度情報と背景深度とをもとに、人物の隠れた足元位置の推定を行うことで重なりの問題に対処する。

4.1 重なり対応システムの概要

2.3 節の深度取得点を固定する方法では、人物が重なった際に正しく床上の足元位置を取得できず、正しい深度を取得できない場合があるという問題があった(図 55)。これは図 53 の問題と同様である。

そこで本章では、図 81 のように BBox の推定が不完全な状況を改善した重なり対応システムを提案する。改善したシステムの概要は以下の通りである(図 56)。

まず、不完全な BBox(図 56 の緑枠)内で見えている人物の一部から代表深度を 1 つ決める。BBox 内には人物以外の領域も含まれるため、この代表深度を取得するためにセグメンテーション AI が必要である。

次に、不完全な BBox から得られた足元座標(px, py)の py を「足元位置補正 y 座標 py'」に変更する(図 56 の青矢印)。py'は、x 座標が px と等しく、かつ人物の代表深度と同じ深度を持つ床の y 座標とする。この py'を求めるために、人物を含まない背景のみの深度推定画像を深度推定 AI によりあらかじめ用意しておく必要がある。

最後に、新たな足元座標(px, py')に対して射影変換を適用し、人物位置(X, Y)を得る。

以上により、重なりによって人物が完全に推定されなかった場合でも、射影変換と同程度の精度(図 47)で人物位置を推定できる改善システムを実現できると考えた。

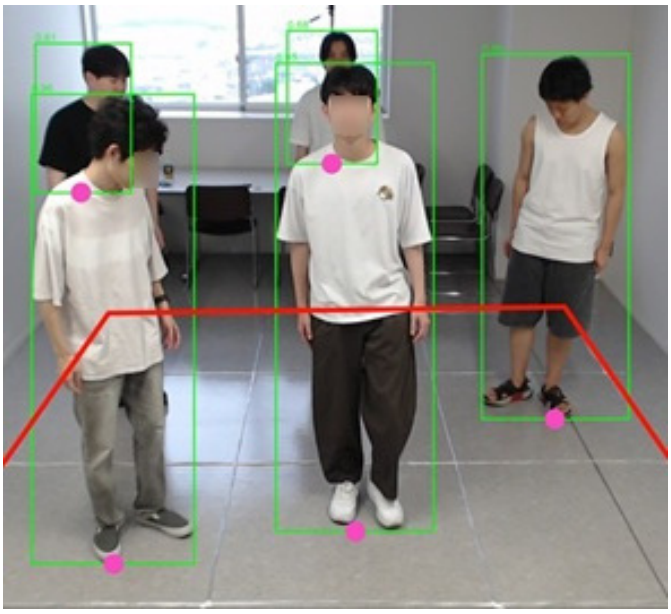

図 55　複数人物の重なりにより人物の正しい深度値を取得できない例

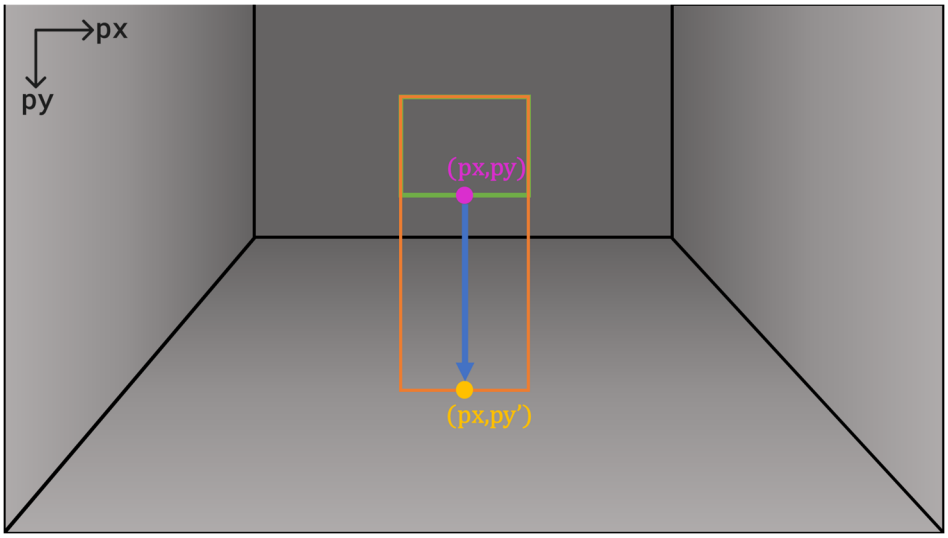

図 56　重なり対応システムの概要

本システムを構築する上で作成したモジュールとその役割を以下に示す。なお、各モジュール同士の関係概略図を図 57 に示す。

4.2 重なり対応システムの構成

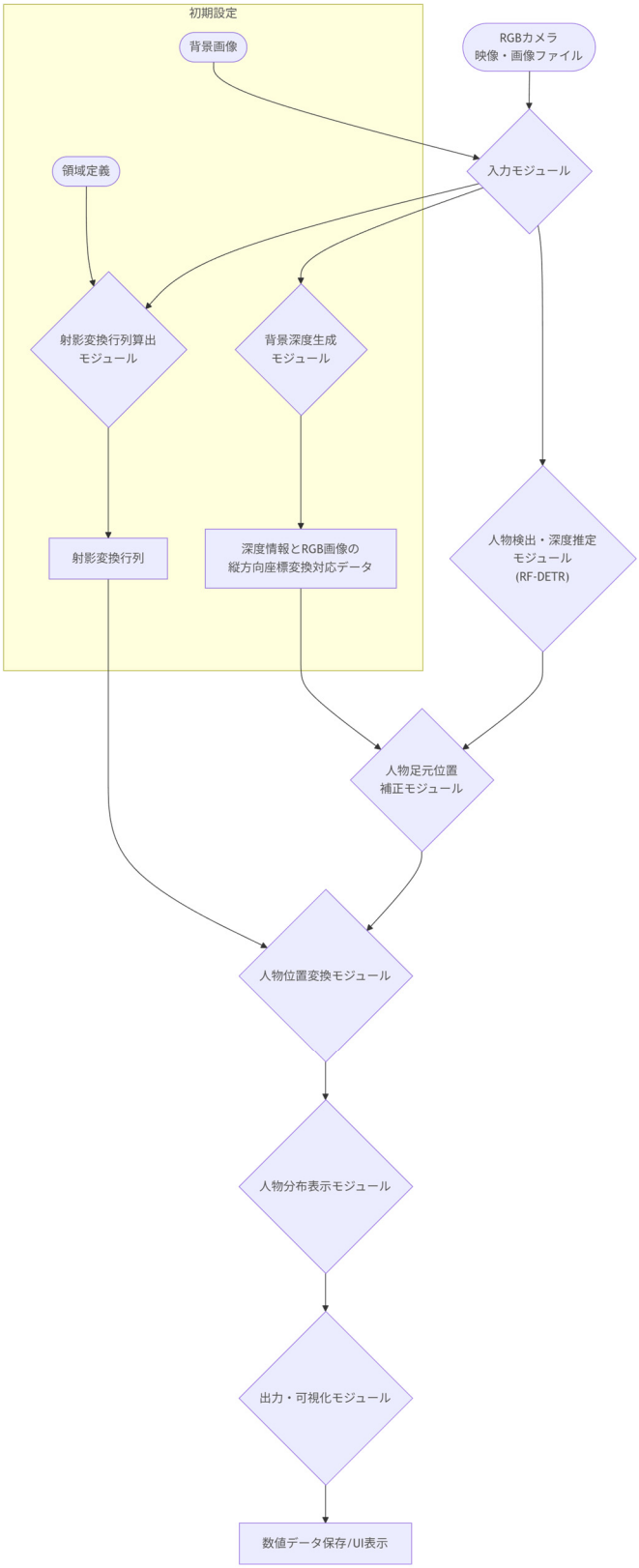

図 57　重なり対応システムの概要図

4.2.1 入力モジュール

入力モジュールでは、RGB カメラ、映像ファイル、画像ファイルからのフレームデータの取得と以降のモジュールへデータの受け渡しを行う(図 58)。

ここで三種類の入力方法があるのは、これらからのデータを同時に受け取るということではなく、カメラからのリアルタイム映像の入力、事前に収録された映像、そして画像としての入力にそれぞれ対応しているということである。

本システムでは、BAG ファイルから抽出した RGB 映像から任意のフレームを選択し、画像ファイルとして入力に使用した。

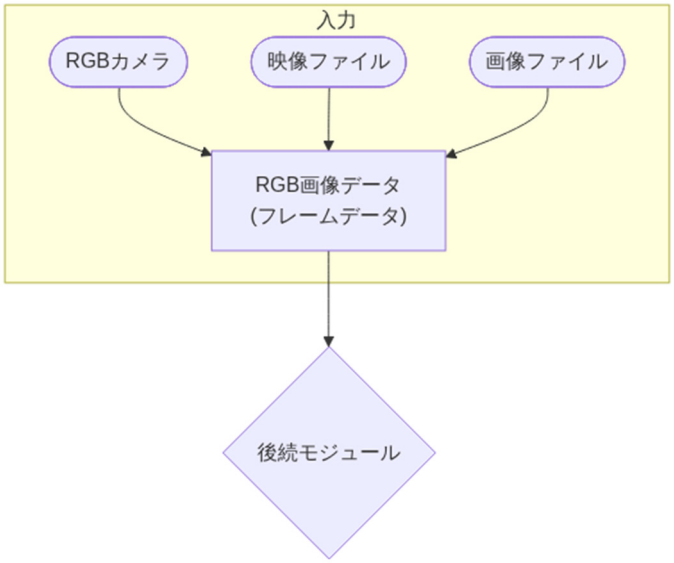

図 58　入力モジュール

4.2.2 背景深度生成モジュール

背景深度生成モジュールでは、入力された背景画像から深度推定 AI を用いて深度を取得し(図 59)、推定された深度情報(図 60 の Y'方向)と、その深度値のある RGB 画像の床の中心線上(図 60 の赤線位置)の縦方向座標(図 60 の py 方向)との対応テーブルを作成する(図 61)。ここでいう対応テーブルとは、「深度情報を入力すると、その深度情報が位置する床の中心線上の縦方向座標、つまり足元位置補正 y 座標を返す」ようにするものである。この対応テーブルを保持しておくことで、人物の代表深度値を入力した際に、その人物の足元位置としてみなせる床の中心線上の足元位置補正 y 座標が返され、足元が見えていない人物であっても足元位置を推定することが可能になる。

また本手法は、人物が床面上に立っている状況では、人物の足元深度と頭部付近の深度が近似できるという前提のもと、人物の可視領域から取得した代表深度値を足元深度として扱えるという考え方に基づいている。さらに、事前に作成した背景深度と人物を含む深度とで、同一の深度スケールが維持され、深度値の範囲が整合している状態を理想とする(図 62)。

なお、深度取得において深度センサを用いた場合でも深度推定 AI を用いた場合でも、それぞれの特性に起因する誤差が避けられず、前述の実験でもそれが原因となり、重なりがない場合であっても真値とは異なる結果が生じていた (図 47)。しかし本手法では、対応テーブルを介して縦方向座標を返すという構成上、深度取得段階で真の深度値を厳密に取得することを要しない。なぜなら、同一の深度推定 AI によって推定された人物の代表深度値と背景深度には同様の特性が現れるため、それら同士を比較することで当該特性の影響が相殺されるためである。

対照実験のため、深度推定 AI に入力する RGB 画像は RealSense で撮影したデータから抽出したものを用いた。しかし、深度センサについては 2.4.2 項で述べた通り、照明をつけた状態では深度が取得できない箇所が発生した。そこで本実験は照明をつけない環境に限定して実施し、それに合わせて深度推定 AI への入力 RGB 画像についても、照明を点灯しない条件で取得したデータに限定されている。 なお、深度推定 AI に与えられるパラメータは、ピクセル単位の焦点距離f_{px}のみであり、部屋の明るさなど環境条件に応じて変更できる設定は存在しない。

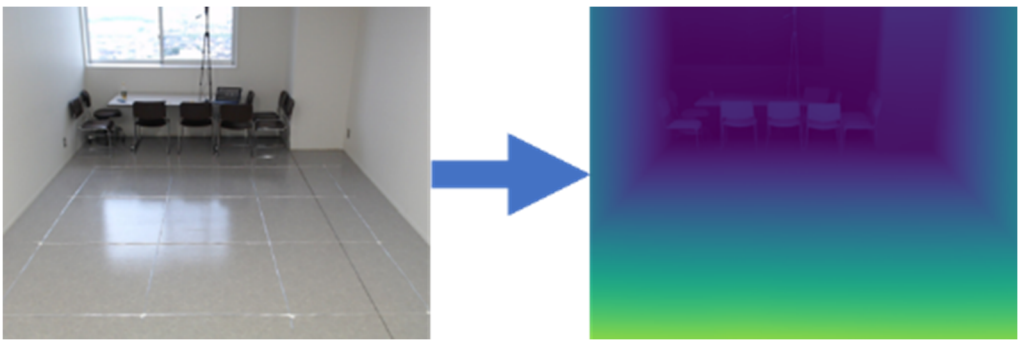

図 59　深度推定 AI による背景深度推定

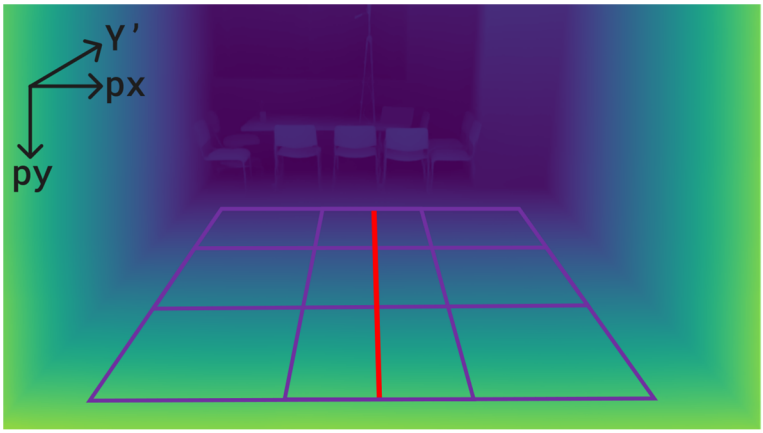

図 60　深度値と床の位置における縦方向座標ピクセル(赤線)との対応テーブルの作成

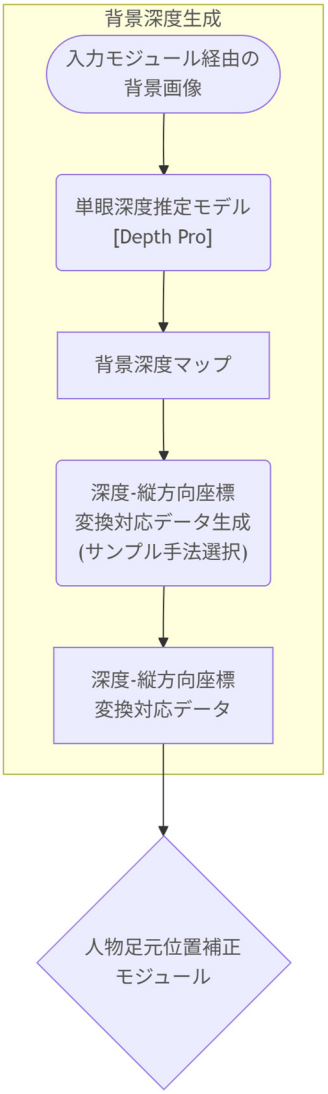

図 61　背景深度生成モジュール

図 62　背景のみの深度情報と人物を含む場合の深度情報

4.2.3 射影変換モジュール

射影変換モジュールでは、事前に入力した背景画像内で人物が立つ範囲の実験領域(3×3)を定義し、その領域を構成する頂点および交点に対応する点を設定して、そこから射影変換行列を算出する(図 63)。なお、特定の画角から写した歪んだ状態の実験領域を、正方形の領域に変換するための射影変換行列は、その対において一意なため、カメラの画角を変更しない限りは再設定しなおす必要はない。

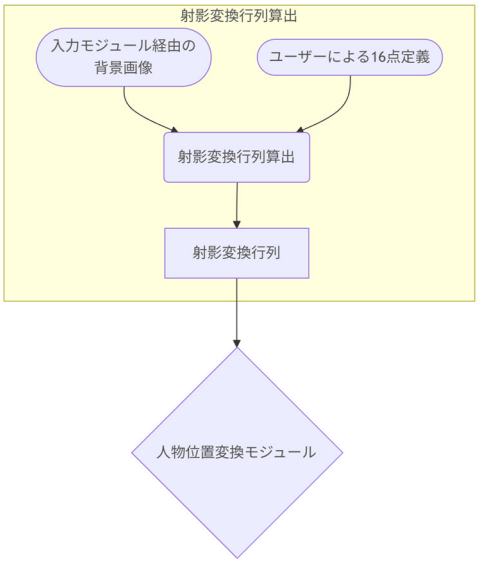

図 63　射影変換行列算出モジュール

4.2.4 人物検出・深度推定モジュール

人物検出，深度推定モジュールでは、人物を含む RGB 画像データに対し、人物検出と人物ありの深度の推定を行う(図 64・図 65)。

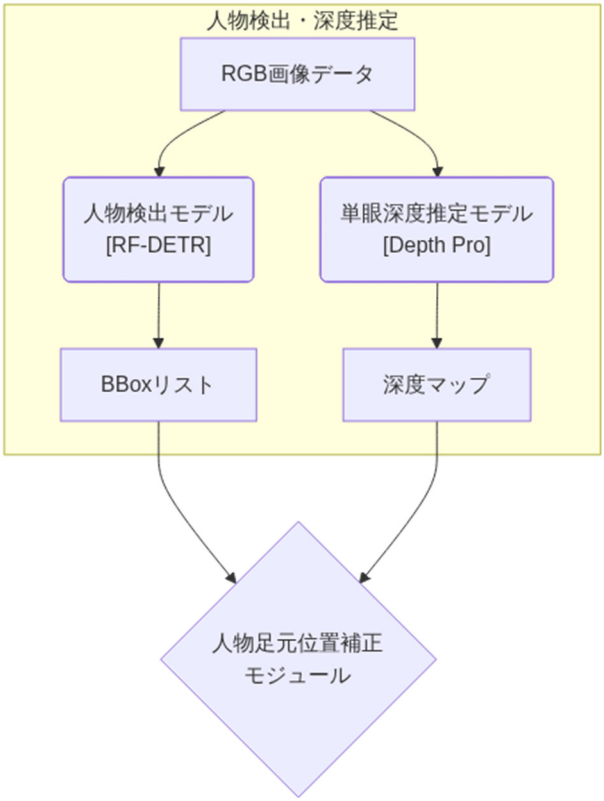

図 64　人物検出・深度推定モジュール

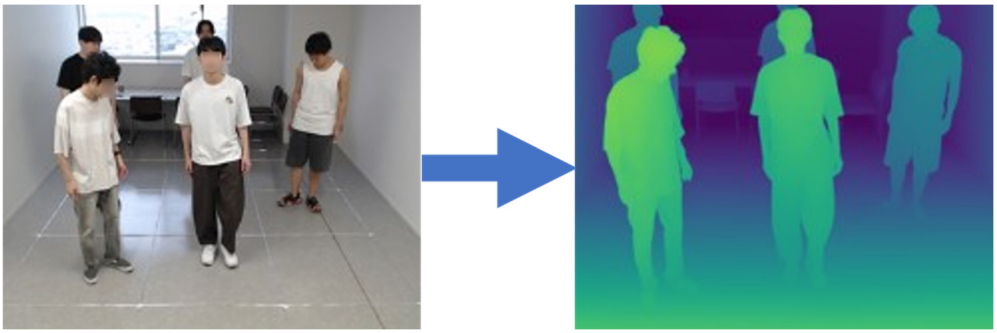

図 65　深度推定 AI による人物を含む深度推定

4.2.5 人物足元位置補正モジュール

人物足元位置補正モジュールでは、人物検出・深度推定モジュール(4.2.4 項)で検出された BBox の縦座標 py を、背景深度生成モジュール(4.2.2 項)で算出した深度-足元位置補正 y 座標変換対応テーブルから取得した足元位置補正 y 座標を用いて、py'に変更することで補正を行う(図 66)。

処理の流れとしては、人物ありの深度情報から各々の人物領域のみの深度情報を取得し、そこから代表深度値を決定する。次に、その代表深度値を深度-足元位置補正 y 座標変換対応テーブルと比較し、返ってきた人物の足元縦座標を、人物検出・深度推定モジュール(4.2.4 項)で検出された人物の BBox 下辺の縦座標に適用することで、人物の足元位置を補正する(図 67)。

人物の代表深度値を決めるためには、まず画像領域内のオブジェクトをセグメントする事前学習済みモデルである Meta の SAM2(Segment Anything Model 2)を用いる。4.2.4 項で解説した人物検出・深度推定モジュールで得られた人物の BBox を SAM2 に入力し、画像中で人物が存在する範囲を指定する。SAM2 はその範囲を手がかりとして、RGB 画像または深度画像から人物部分のみをセグメントし、各 BBox に対応するセグメントマスクを生成する(図 68)。セグメントマスクとは、特定領域を選択的に取り出すことを可能にするものであり、図 68 では画像中の人物部分だけがピクセル単位でセグメントされ、人物領域と背景領域が分けられた状態を表している。そのセグメントマスクを深度画像に適用し、人物領域の深度値を取得する。セグメンテーションを適用する入力は RGB 画像または深度画像のいずれでもよく、人物領域をより良好にセグメントできる方を選択して用いる。本論文では、RGB 画像に対してセグメンテーションを適用した。

なお、人物の代表深度は人物領域内の深度値の 95 パーセンタイル値を使用した。95 パーセンタイル値は、人物領域内の深度値を小さい順に並べたときに下位 95%が収まる境界の値であり、深い側(足元に近い深度)の値を代表値として採用するための指標である。平均値のように外れ値の影響を受けやすい方法に比べ、セグメンテーションの境界付近の満ち欠けによる背景混入などの影響を抑える効果がある。図 69 は、例として取り出した人物 1 人分の人物領域における深度値の分布を示しており、95パーセンタイル値は 2.51[m]である。また、その値を含む範囲をオレンジで示した。

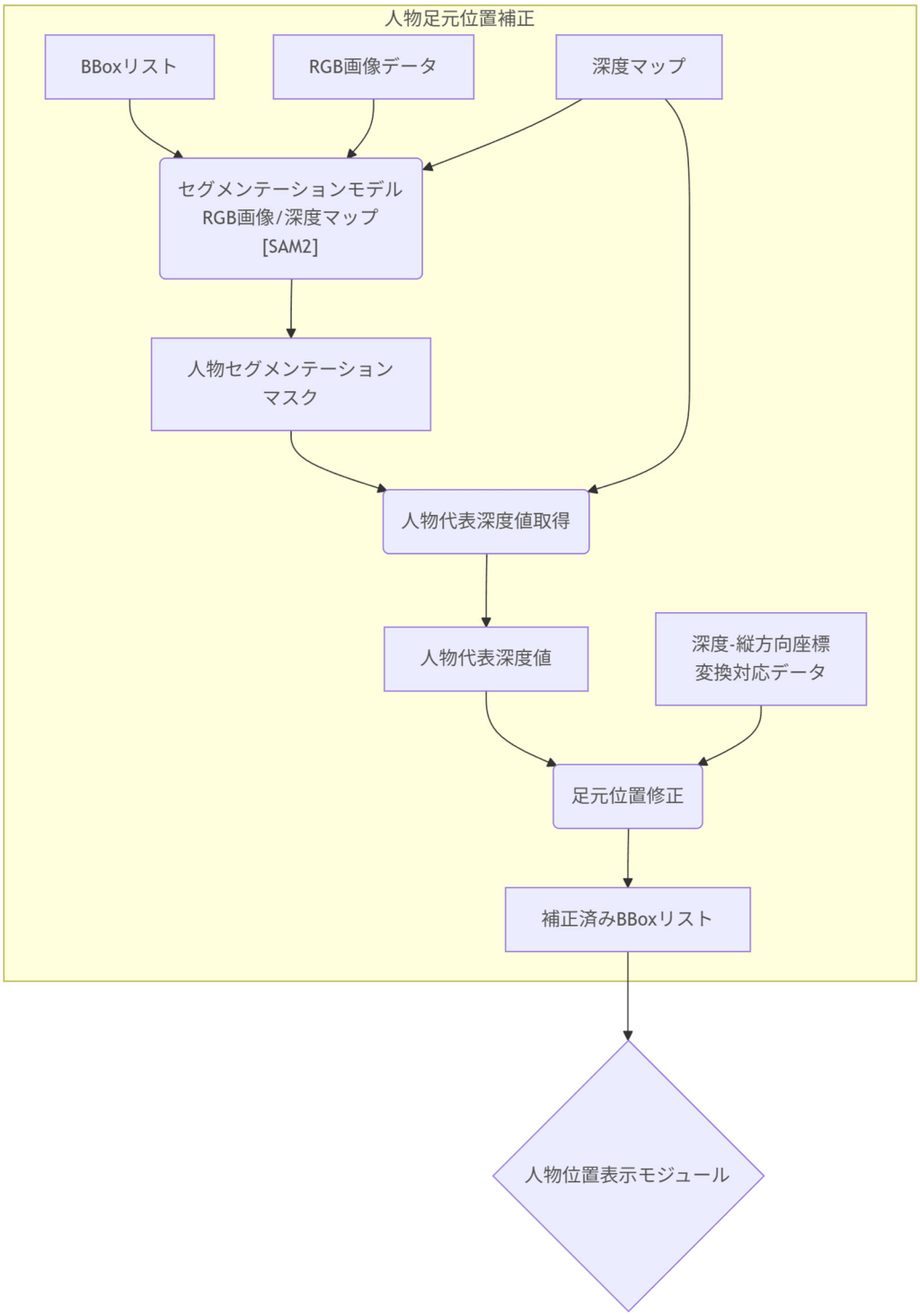

図 66　人物足元位置補正モジュール

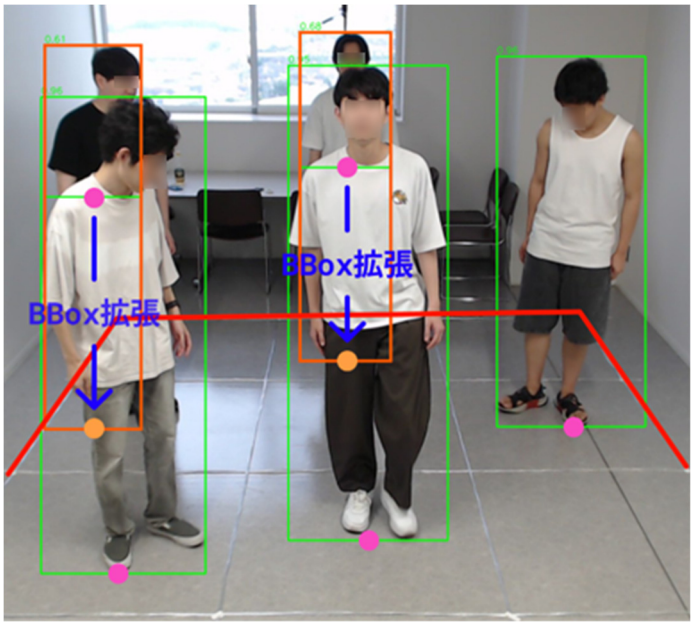

図 67　BBox 下辺の拡張

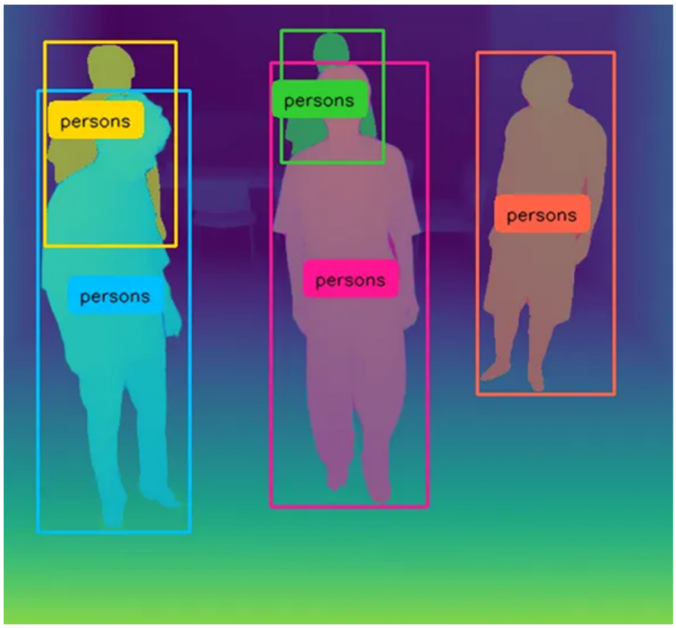

図 68　SAM2 による人物領域のセグメンテーション

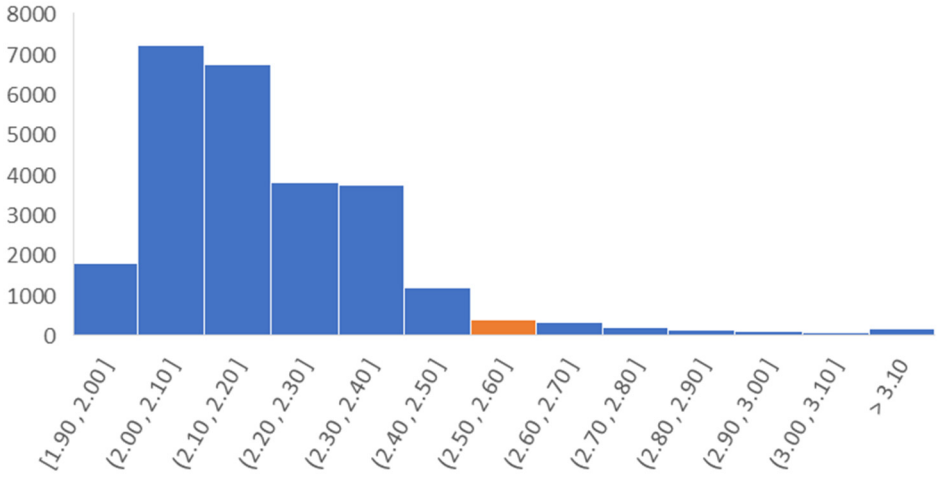

図 69　人物領域内の深度値分布と 95 パーセンタイル値の値を含む範囲(オレンジ)

4.2.6 人物位置変換モジュール

人物位置変換モジュールでは、人物足元位置補正モジュール(4.2.5 項)で補正された BBox から下辺の中点を代表点として取り出し、それにより得られる足元位置(px, py')を、射影変換モジュール(4.2.3 項)で算出した射影変換行列を用いて、定義された 2 次元床面上の座標(X', Y')へと変換する(図 70、図 71)。

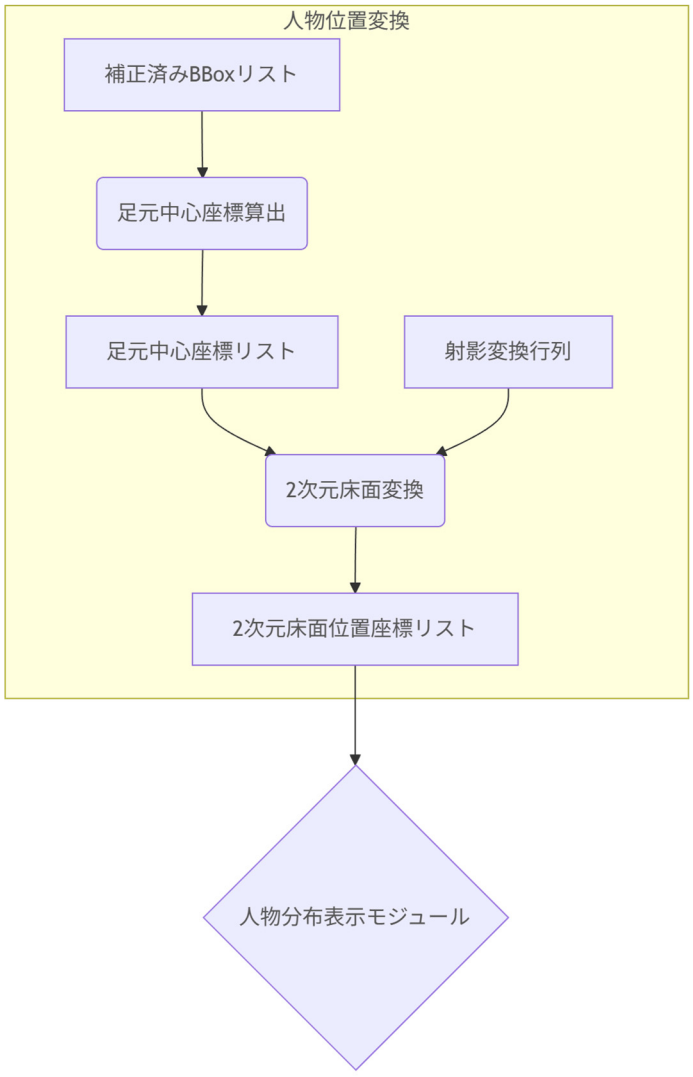

図 70　人物位置変換モジュール

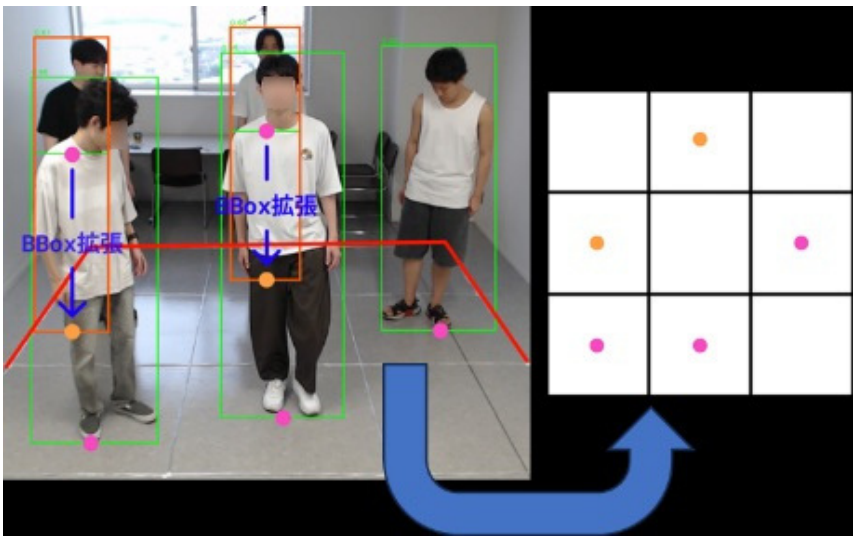

図 71　人物の足元位置の 2 次元床面上への投影

4.2.7 人物分布表示モジュール

人物分布表示モジュールでは、人物位置変換モジュール(4.2.6 項)で得られた人物位置(X', Y')をもとに、実験領域内における各グリッド内の人数の表示を行う(図 72)。

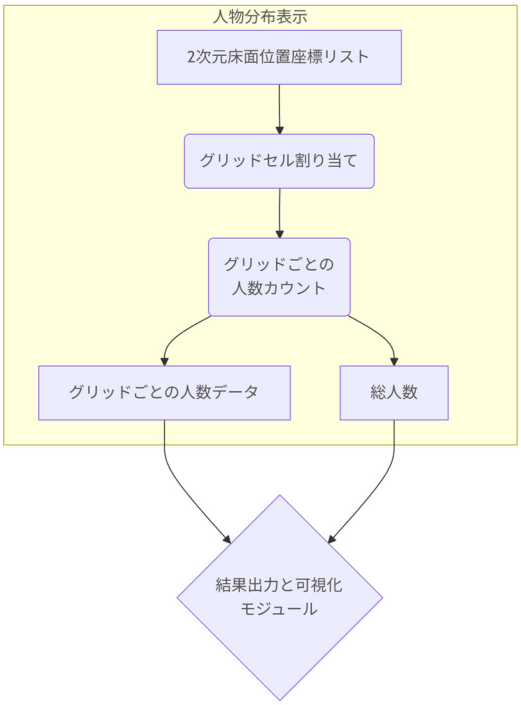

図 72　人物分布表示モジュール

4.2.8 出力・可視化モジュール

出力・可視化モジュールでは、処理結果を GUI に表示する役割を担う(図 73)。

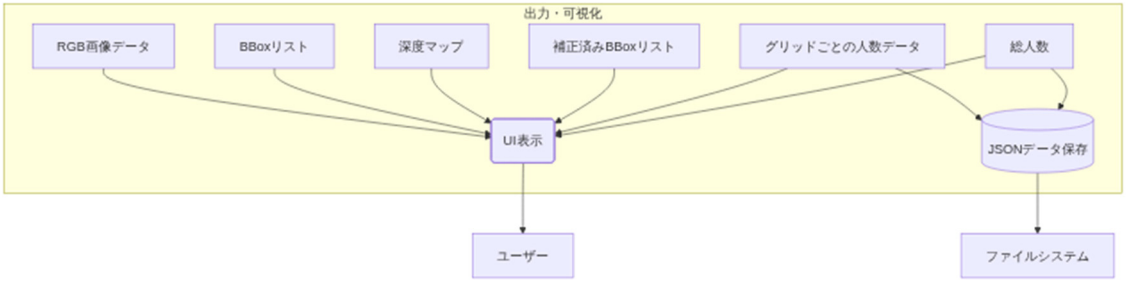

図 73　出力・可視化モジュール

4.2.9 構築した重なり対応システム

完成した重なり対応システムを示したのが図 74 である。図 74 の左上は、人物検出のみを行った画面で、検出結果を緑の BBox で示している。右上は推定された深度画像、左下は補正処理後の BBox、右下は領域内における補正後の人数分布図である。

深度画像の配色は図 62 と異なるが、深度値を色に変換して表示する際に用いるカラーマップを、視認性の観点から viridis から magma へ変更したためである。具体的には、viridis では浅い領域が緑系、深い領域が青紫系で表現されるのに対し、magma では浅い領域が黒系、赤系に近い色で表現される。この変更は可視化のための表示上の差にとどまり、推定された深度値自体や後段の処理結果には影響しないため、どのカラーマップを用いるかは最終的には見やすさの好みに依存する。

また、viridis および magma はいずれも入力値に対して明度が線形に変化するため、深度の変化がグラデーションとして均等に知覚される。これにより、深度推定結果の確認の際に、特定の値域が不自然に強調されたり、明暗の反転によって遠近関係の解釈を取り違えたりするリスクを抑えつつ、奥行きの大小関係を直感的に読み取りやすいため採用した。

図 75 は、セグメンテーションされた人物を画像上に重ね描きしたものである。

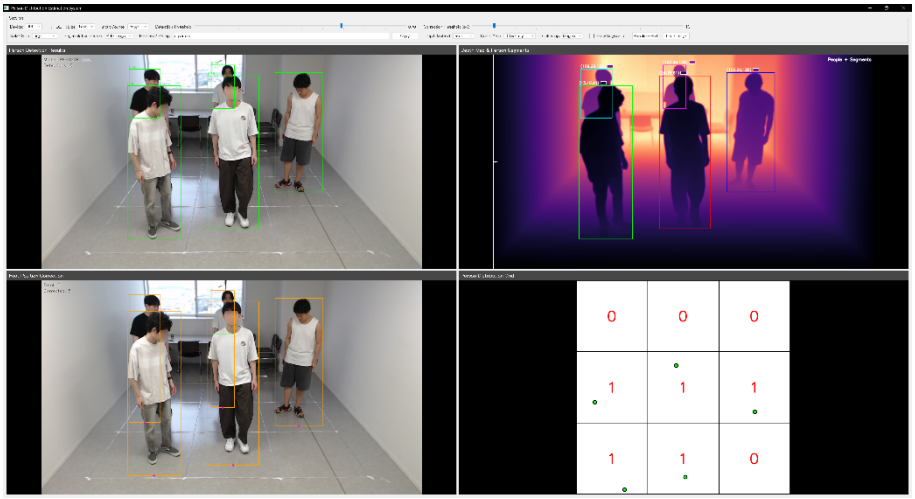

図 74　重なり対応システムメイン画面

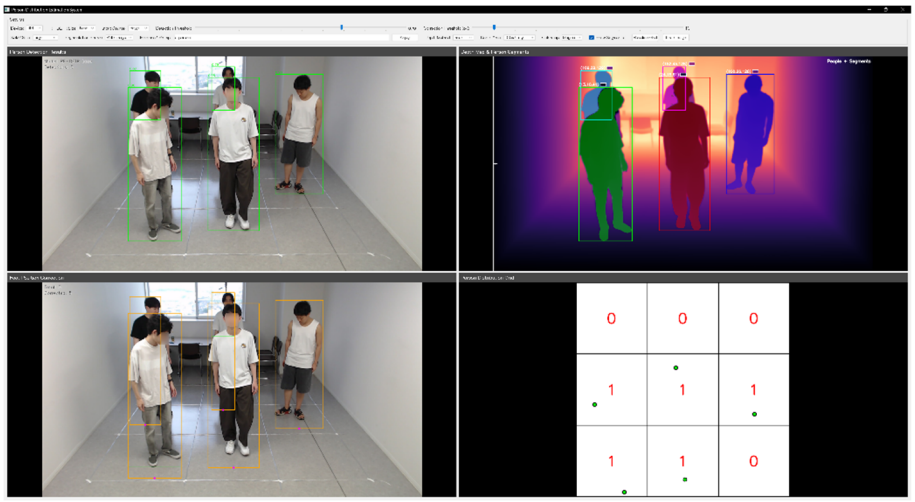

図 75　人物領域を重ね書きした画像(右上)

第 5 章 重なり対応システムの性能評価実験（米原）

本章では、重なり対応システムと 3 章で紹介した深度取得方法の比較評価実験を行う。

5.1 実験目的

本実験の目的は、4 章で我々が構築した重なり対応システムと 3 章で使用した 3 つの深度取得方法とで深度取得精度と平均処理時間を比較し、人物分布システムの評価を行うことである。

5.2 実験方法

本実験の使用機器、実験環境、および評価方法は 3 章と同様とする。

3.2 節で示した計 17 パターンの画像に対し、重なり対応システムで深度の取得を行い、3 章の結果との比較を行う。

5.3 実験結果

5.3.1 重なりがない場合の深度取得精度

図 76 は重なりがない場合の深度を比較したグラフである。実測値が 280cm、380cm の地点では、構築した重なり対応システムで取得された深度は、射影変換とほぼ同様の値を取っている。ただし、ばらつきは射影変換に比べてやや大きい。また、実測値が 480cm の地点では、重なり対応システムで取得された深度より射影変換で取得された深度の方が誤差が小さいことがわかった。さらに、深度推定 AI と深度センサよりも重なり対応システムの方が、取得された深度が実測値と近くなる傾向がみられるが、これは、4.1 節で先述したように、補正後のピクセル座標(px, py')から(X', Y')への変換に射影変換を用いているためである。

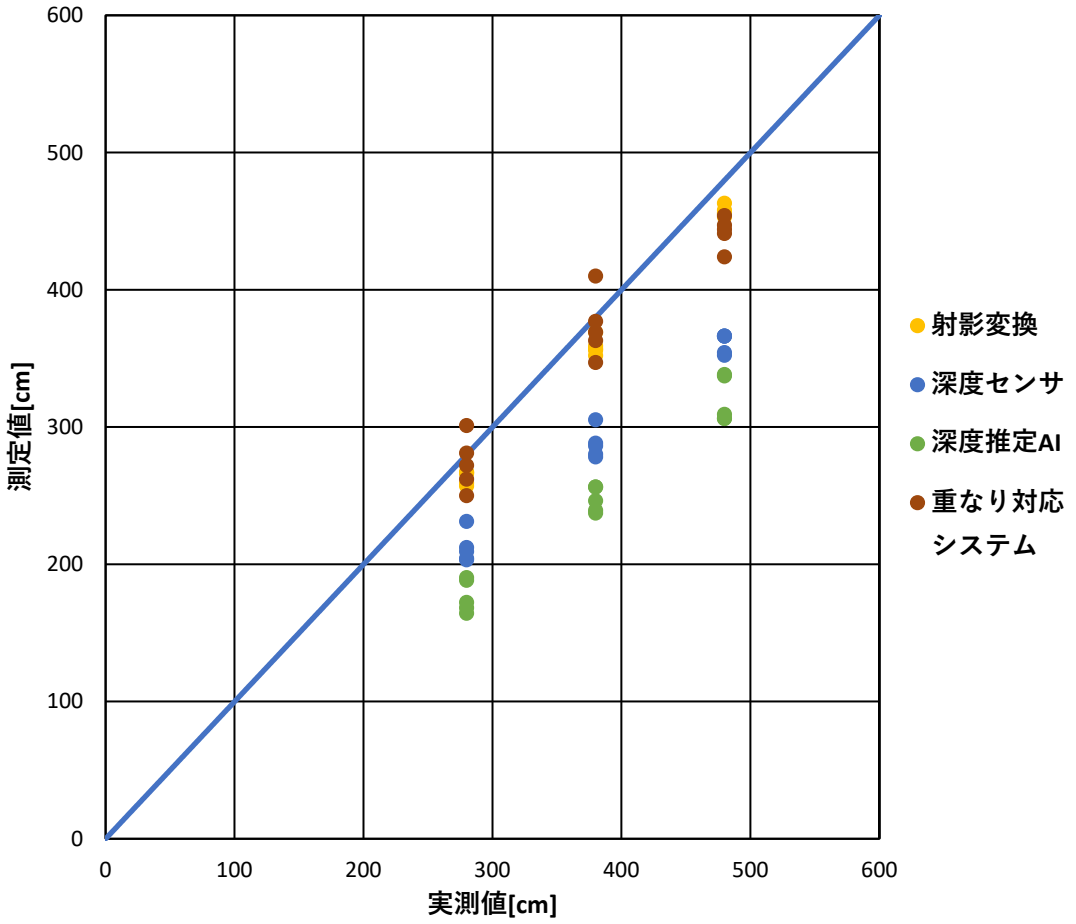

図 76　重なりがない場合

5.3.2 重なりがある場合の深度取得精度

図 77 は、重なりがある場合の深度を比較したグラフである。実測値が 380cm、480cm の地点で、射影変換と深度推定 AI において誤差が大きくなっているが、重なり対応システムではそれが抑えられていることがわかる。また、誤差が大きくなっていた場合を除くと、深度センサと深度推定 AI に比べ、深度取得精度が向上していることがわかった。これは、5.3.1 項と同様に、補正後のピクセル座標(px, py')から(X', Y')への変換に射影変換を用いているためである。

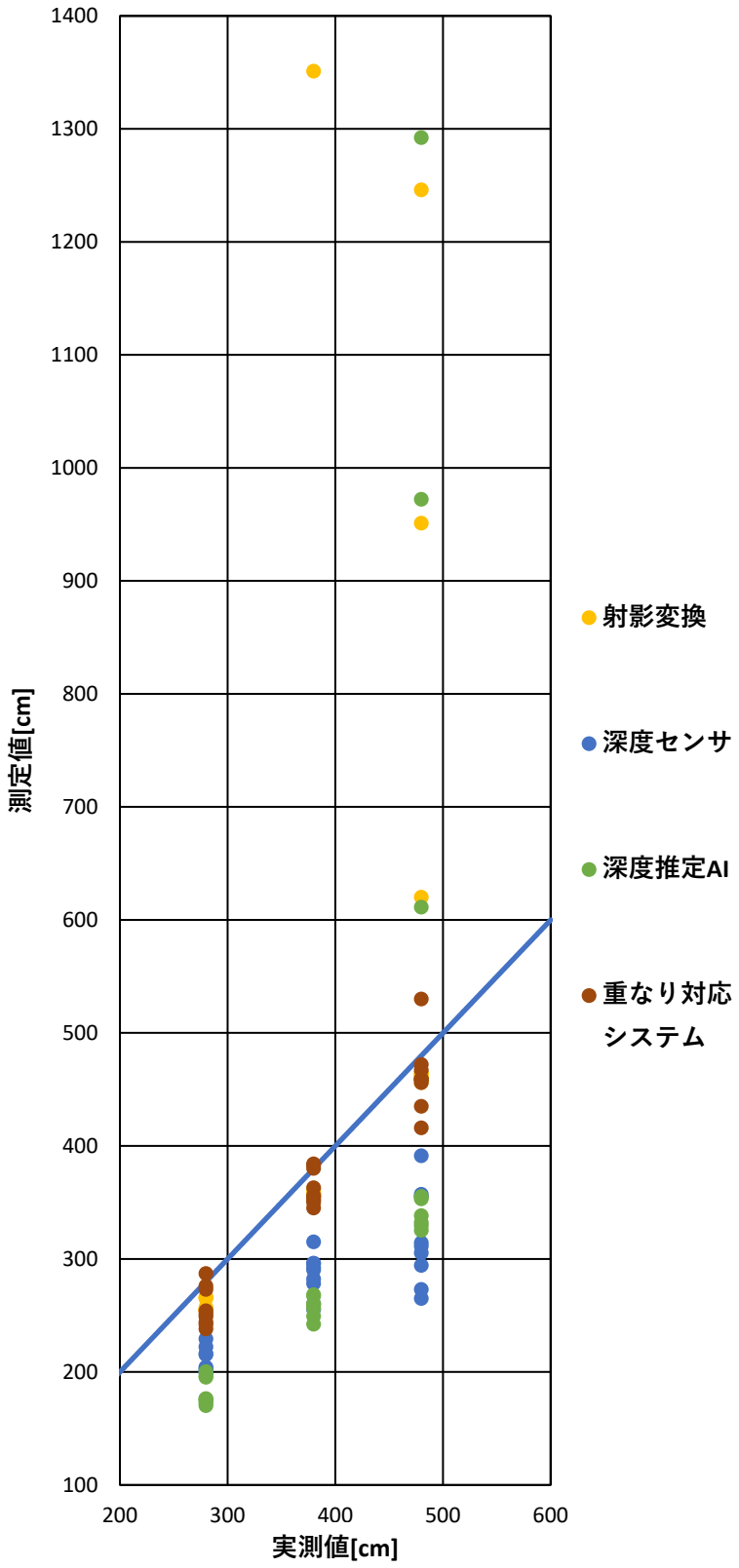

図 77　重なりがある場合

5.3.3 重なり環境下における左右位置と深度誤差の関係

本項では、複数の人物を左列、中央列、右列に配置したパターンの深度取得精度を比較する。5.3.2 ではこの 3 つのパターンをまとめた結果を示したが、本節ではそれを個別に検証するということである。

図 78、図 79、図 80 はそれぞれ人物が「左列で重なった場合」、「中央列で重なった場合」、「右列で重なった場合」の深度を比較したグラフである。

図 78、図 80 からわかるように、人物が「左列で重なった場合」および「右列で重なった場合」では、重なり対応システムの性能は射影変換と同等程度であることがわかる。これは、図 52、図 53 に示されているように、人物の足元座標(px, py)が補正なしでも正しく推定できているからである。

一方、図 79 からわかるように、人物が「中央列で重なった場合」は重なり対応システムを用いることで大きく深度の精度が向上することがわかる。これは、図 53 に示されているように、人物の足元座標(px, py)が正しい結果を表しておらず、重なり対応システムで導入した補正が有効に働くためである。

この効果をより詳しく検証する目的で、複数の人物を中央列に配置したパターンの重なり対応システムによる推定結果を表示したのが図 81、図 82、図 83、図 84 である。

まず、どの結果でも X 方向の推定は安定していることがわかる。Y 方向の推定は図 81 ではおおむね正しいと言える。ただし、図 82、図 84 の奥の人物のように、Y 座標が実際より大きく推定されることがあることがわかった。これは、セグメントされた人物領域が小さすぎたことが原因である。また、図 83 と図 84 のように、手前の人物の Y 座標が実際よりも小さく推定されることがあることがわかった。これは、システムで補正を行う人物を選定しておらず、全身が映っている人物に対しても補正を行っていることが原因である。

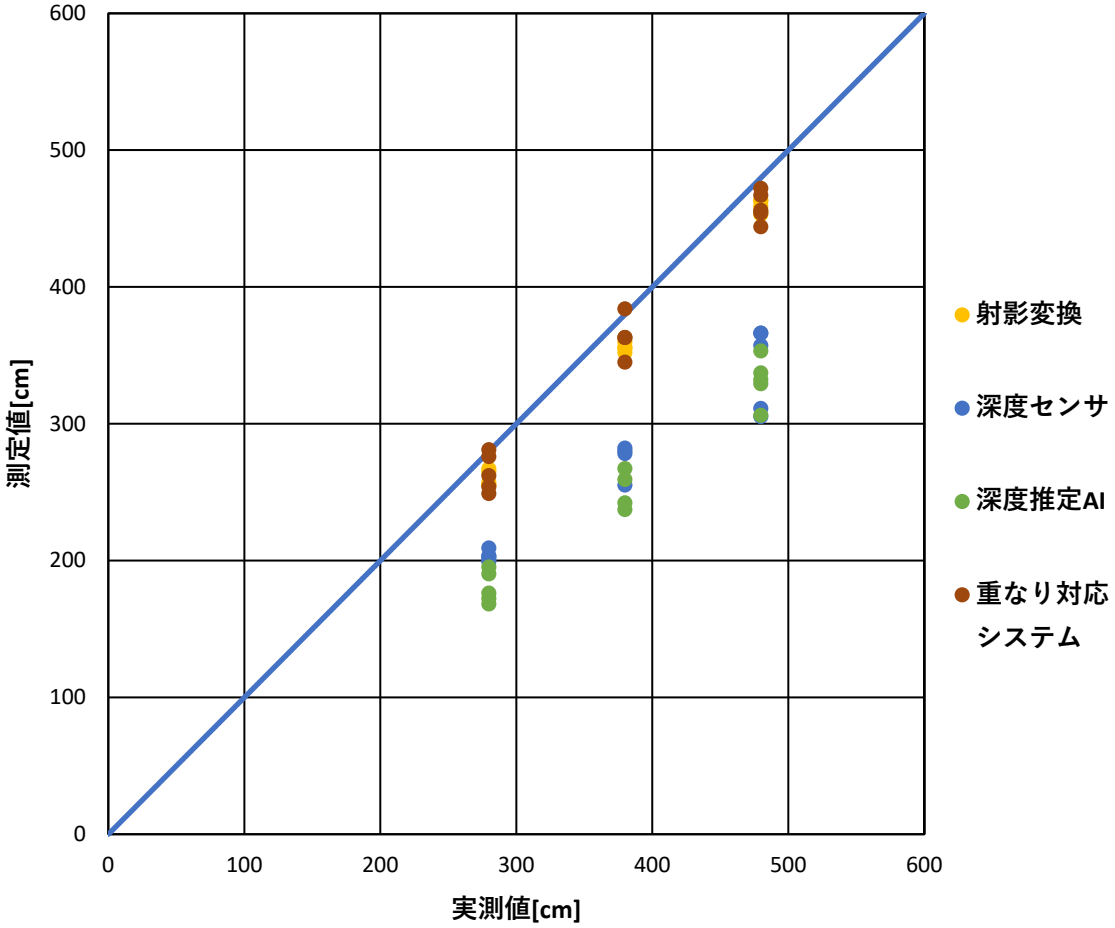

図 78　複数の人物が実験領域の左列で重なった場合の深度の比較

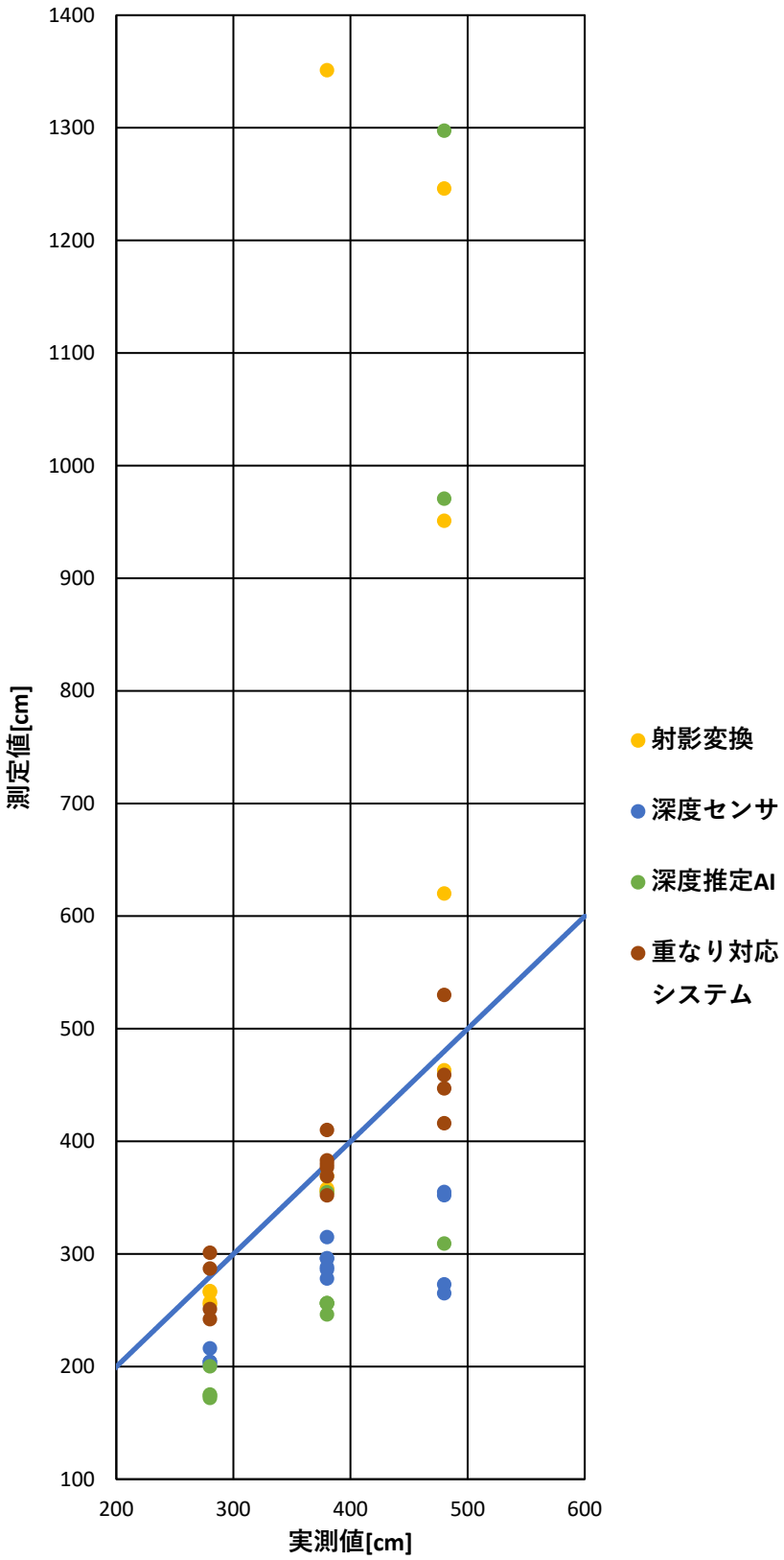

図 79　複数の人物が実験領域の中央列で重なった場合の深度の比較

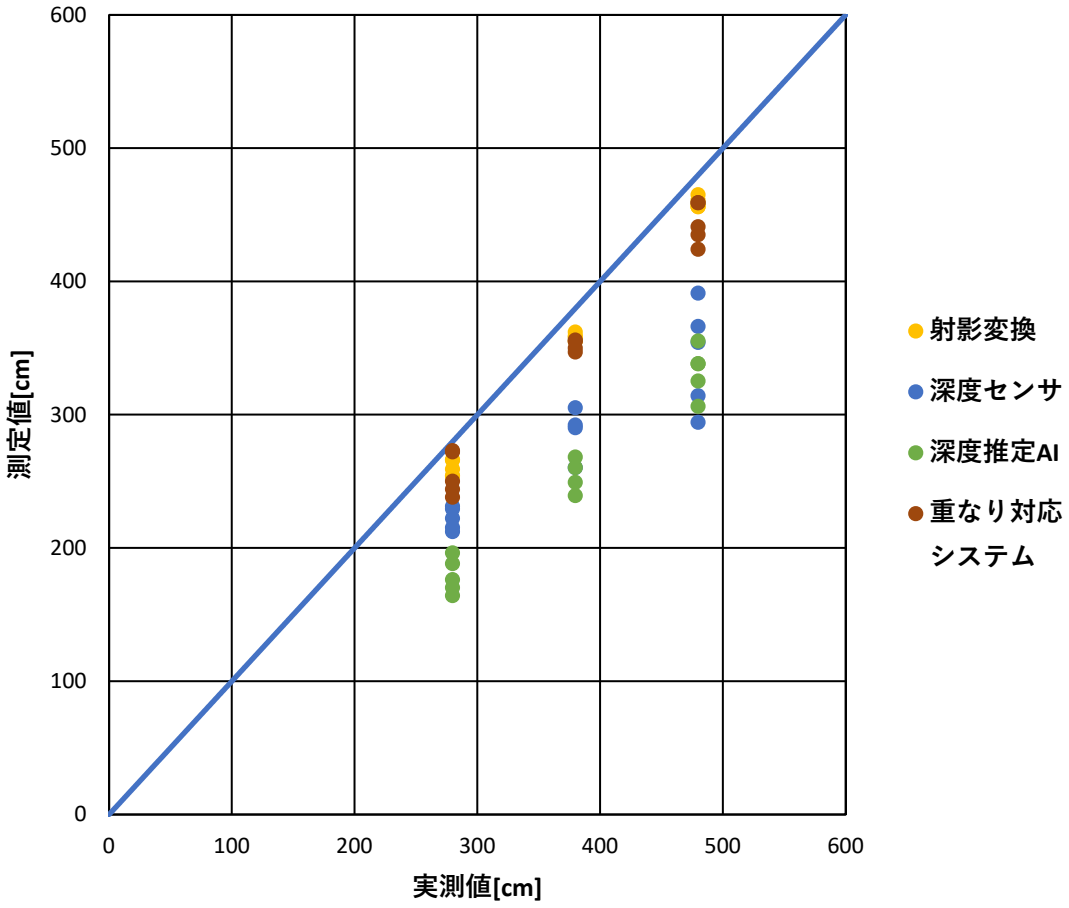

図 80　複数の人物が実験領域の右列で重なった場合の深度の比較

図 81　重なり対応システムの検出例 1

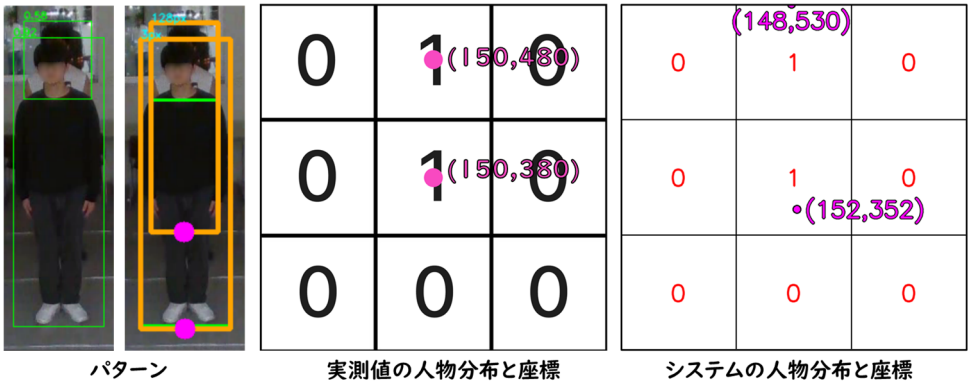

図 82　重なり対応システムの検出例 2

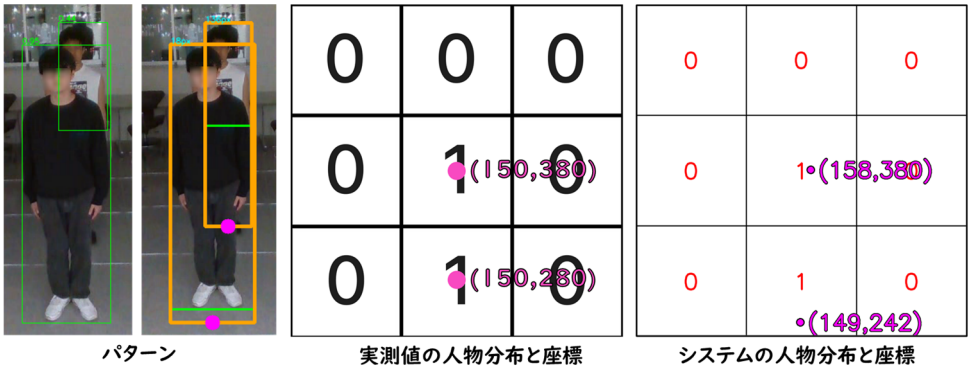

図 83　重なり対応システムの検出例 3

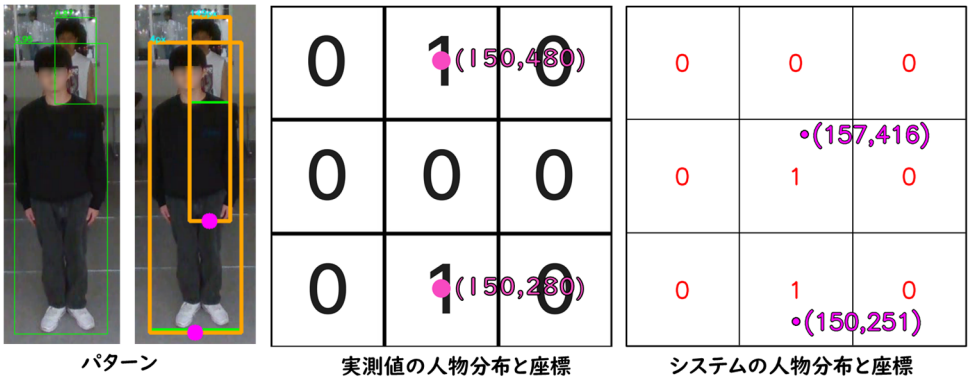

図 84　重なり対応システムの検出例 4

5.3.4 重なり対応システムと各深度取得方法の平均処理時間

本項では、重なり対応システムの平均処理時間を各深度取得方法と比較し、重なり対応システムのリアルタイム性について検討する。

表 5 より、各深度取得方法と比較して、重なり対応システムの平均処理時間は 10〜300 倍と、リアルタイム処理には課題があることがわかった。これは、他の深度取得方法と比較して、処理の工程が多いことと、各工程にかかる処理に時間がかかることが原因である。

なお、表 5 は、複数人物検出モデルの推論時間 25.2ms/Frame を除いたものである。

表 5　平均処理時間（重なり対応システム有）

射影変換	3.6ms/Frame
深度センサ	3.3ms/Frame
深度推定 AI	119.8ms/Frame
重なり対応システム	974.8ms/Frame

第 6 章 結論（米原）

本研究では単眼カメラを用いて重なりに強い人物分布推定を行うことを目標に人物分布推定システムの構築を行った。

単眼カメラで撮影した動画に対して物体検出モデル RF-DETR で人物検出を行い、取得した足元位置の座標(px, py)を、各深度取得方法で実世界の位置(X', Y')に変換した。Y'を測定した深度 Y と比較することで、深度取得精度と処理時間を評価した。その結果、重なりがない、または足元が見えている状況では射影変換が一番優れていることが分かった。しかし、完全に人が重なっている、もしくは足元が見えない状況の場合、深度取得精度が低下する問題があることが分かった。

そこで、人物の重なりに対応したシステムを構築し、深度を補正した。その結果、すべてのパターンに改善が見られ、単眼カメラを用いた重なりに強い人物分布推定を実現することができた。ただし、重なりがないもしくは足元が見えているパターンでは射影変換と同等程度の精度であった。

重なり対応システムは多くの処理工程を含むため、平均処理時間が他の深度取得方法と比較して大きく、リアルタイム処理への適用には課題が残っている。解決策として、ネットワーク上で利用可能な外部の計算環境を用いて、複数の GPU を用いた並列処理を行うことが考えられる。

参考文献

[1] 国土交通省,"訪日外国人旅行者数・出国日本人数｜観光統計・白書",
https://www.mlit.go.jp/kankocho/tokei_hakusyo/shutsunyukokushasu.html
[2] 豊島区の観光客データ　東京都
https://machi-graph.com/city/toshima-13116/tourist
[3] 豊島区における「群衆行動解析技術」を活用した総合防災システム
https://jpn.nec.com/techrep/journal/g18/n01/180107.html
[4] 群衆行動解析技術を用いた混雑推定システム-NEC
群衆行動解析技術を用いた混雑推定システム（Vol.67 No.1 2014 年 11 月　社会の安全・安心を支えるパブリックソリューション特集）: 2014 年 ｜NEC,
https://jpn.nec.com/techrep/journal/recommend_year/2014/06.html
[5] 田淵 義宗, 高橋 友和, 出口 大輔, 井手 一郎, 村瀬 洋, 黒住 隆行, 鹿野 清宏. (2013). 複数カメラを用いた人数分布推定に関する検討. 電子情報通信学会技術研究報告. PRMU, パターン認識・メディア理解, 113(230), 1-6.
https://nagoya.repo.nii.ac.jp/records/21715
[6] 永田 毅, 鎌田 清一郎, 溝口 理一郎. (2004). 監視カメラ映像中の局所的な動き検出とイベント累積による時間的かつ空間的な混雑度調査. 電気学会論文誌D（産業応用部門誌）, 124(10), 1060-1066.
[7] Intel Corporation, “インテル® RealSense™ デプスカメラ D435：製品仕様.” [Online]. Available: https://www.intel.co.jp/content/www/jp/ja/products/sku/128255/intel-realsense-depth-camera-d435/specifications.html.
[8] Intel Corporation, “Intel® RealSense™ D400 Series Product Family Datasheet,” Document Number: 337029-005, Revision 005, Jan. 2019. [Online]. Available: https://cdrdv2-public.intel.com/841984/Intel-RealSense-D400-Series-Datasheet.pdf.
[9] Daniil Osokin. Real-time 2D Multi-Person Pose Estimation on CPU: Lightweight OpenPose. arXiv:1811.12004, 2018. https://arxiv.org/abs/1811.12004.
[10] Aleksei Bochkovskii, Amaël Delaunoy, Hugo Germain, Marcel Santos, Yichao Zhou, Stephan R. Richter, Vladlen Koltun. Depth Pro: Sharp Monocular Metric Depth in Less Than a Second. arXiv:2410.02073, 2024. https://arxiv.org/abs/2410.02073.
[11] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, Christoph Feichtenhofer. SAM 2: Segment Anything in Images and Videos. arXiv:2408.00714, 2024. https://arxiv.org/abs/2408.00714.
[12] Isaac Robinson, Peter Robicheaux, Matvei Popov, Deva Ramanan, Neehar Peri. RF-DETR: Neural Architecture Search for Real-Time Detection Transformers.

arXiv:2511.09554, 2025. https://arxiv.org/abs/2511.09554.
[13] Intel RealSense Community, “Depth distance value.” [Online]. Available: https://support.realsenseai.com/hc/en-us/community/posts/37255812238227-Depth-distance-value
[14] ROBO-HI 株式会社, “ステレオカメラについて.” [Online]. Available: https://www.robo-hi.jp/knowledge/adas_dev/adas_sensor/adas_camera/adas_stereo

謝辞

最後に、本研究を進めるにあたり、2 年間にわたってご指導、ご鞭撻のほどをいただいた金丸隆志教授には、心より感謝申し上げます。研究を進める中で、自分らの未熟さから意見が対立する場面もありましたが、その都度真摯に向き合ってくださり、ご指導いただきましたことに深く感謝いたします。