Multiple Human Tracking Using an Omnidirectional Camera with Local Rectification and World Coordinates Representation

Hitoshi NISHIMURA†,††,1, Naoya MAKIBUCHI†,†b, Kazuyuki TASAKA†,††,c, Yasutomo KAWANISHI†,††,d, Members, and Hiroshi MURASE†,††,e, Fellow

SUMMARY Multiple human tracking is widely used in various fields such as marketing and surveillance. The typical approach associates human detection results between consecutive frames using the features and bounding boxes (position+size) of detected humans. Some methods use an omnidirectional camera to cover a wider area, but ID switch often occurs in association with detections due to following two factors: i) The feature is adversely affected because the bounding box includes many background regions when a human is captured from an oblique angle. ii) The position and size change dramatically between consecutive frames because the distance metric is non-uniform in an omnidirectional image. In this paper, we propose a novel method that accurately tracks humans with an association metric for omnidirectional images. The proposed method has two key points: i) For feature extraction, we introduce local rectification, which reduces the effect of background regions in the bounding box. ii) For distance calculation, we describe the positions in a world coordinate system where the distance metric is uniform. In the experiments, we confirmed that the Multiple Object Tracking Accuracy (MOTA) improved 3.3 in the LargeRoom dataset and improved 2.3 in the SmallRoom dataset.

1. Introduction

Multiple human tracking is a fundamental technique and widely used in various fields such as marketing, surveillance, and virtual reality. The task of multiple human tracking is to achieve continuous detection of multiple humans while maintaining their identities (ID) given time-series images [1]. In a large-scale practical system, one approach that is efficient is to transmit captured images to a server and process them collectively. In such a system, it is expected that the frame rate is low in terms of keeping bandwidth or data storage.

Most state-of-the-art tracking methods [2]–[7] are based on a tracking-by-detection approach owing to detection accuracy improvement. The tracking-by-detection approach achieves multiple human tracking by iterative data association [8]. The data association matches detection results between consecutive frames with an association metric.

Human detection is making remarkable advances based on deep learning (e.g. Faster R-CNN [9], YOLO [10], and SSD [11]), and the detection accuracy has significantly improved. The use of Convolutional Neural Networks (CNNs) is one of the most important deep learning methods, and can extract powerful discriminative feature representations. Most tracking-by-detection approaches use both features and bounding boxes (position+size) for the association metric, and utilize CNNs for feature extraction and bounding box estimation. In this work, we employ powerful deep learning methods.

Conventional deep-learning-based tracking methods [2]–[7] have commonly used a normal camera. In addition to a normal camera, in recent years, an omnidirectional camera has been used for tracking. The omnidirectional camera has a 360-degree view, and can cover a wide area using a single camera. Therefore, the omnidirectional camera reduces initial costs and subsequent maintenance costs (e.g. costs associated with setup, labor, repairs, and software licensing) compared to those of a normal camera. In this study, we utilize only one omnidirectional camera.

However, it is difficult to simply apply the deep-learning-based tracking method [2]–[7] to an omnidirectional image. While omnidirectional images have serious distortions, most deep-learning-based detection methods [9]–[11] estimate the human region as a simple axis-aligned bounding box. When applying these methods to omnidirectional images, ID switch, which means the target human ID changes to another ID, often occurs. ID switch occurs due to following two factors. i) The feature is adversely affected because the bounding box includes many background regions when a human is captured from an oblique angle (Fig. 1 (a)). ii) The position and size change dramatically between consecutive frames because the distance metric is non-uniform (Fig. 1 (b)).

Some tracking methods use omnidirectional cameras exclusively. Most of them rectify the entire omnidirectional image (expand to a panoramic image, hereinafter, "global rectification") before initiating tracking [12]–[18]. In the rectified image, the angle of the human can be normalized. However, the bounding box includes many background regions because there is serious distortion when the human’s position is around the center in the original image.
Therefore, the above two factors illustrated in Fig. 1 are not solved.

In this paper, we propose a novel method that accurately tracks humans with an association metric for omnidirectional images. Note that the proposed method works for omnidirectional images captured by a camera fixed on the ceiling. The proposed method has two key points: i) For feature extraction, we introduce local rectification, which rectifies the human regions locally (not the overall image). It reduces the effect of background regions in the bounding box. ii) For distance calculation, we describe the positions in a world coordinate system where the distance metric is uniform. The proposed method (above two points) can be added to other arbitrary state-of-the-art trackers and improve the tracking accuracy of those trackers.

Our main contributions are as follows:

- We propose local rectification for reducing the effect of background regions when extracting features.
- We describe the human position in a world coordinate system where the distance metric is uniform.
- For above two contributions, we utilized a 3D-human model which is robust against unstable human detection.

The rest of the paper is organized as follows. First, we review related work in Sect. 2. Then, we describe the proposed method in Sect. 3, and conduct the experiments in Sect. 4. Finally, we conclude our work in Sect. 5.

2. Related Work

In this section, we review multiple human tracking methods in relation to the type of camera, rectification, and use of deep learning. Table 1 shows related work compared to ours.

A) Many state-of-the-art tracking methods use normal cameras [2]–[7] and are based on a tracking-by-detection framework. Bewley et al. proposed SORT, which utilizes only bounding boxes for data association [2]. Wojke et al. extended SORT [2], and data association is performed using not only the bounding boxes but also features [3]. Both features and bounding boxes are estimated by deep learning.

B) Many methods that globally rectify omnidirectional images before the tracking have been proposed [12]–[18]. In a globally rectified image, the angle of the human can be normalized. Gächter utilized the temporal and background change detection [12]. Cielnias et al. proposed a method that performs human extraction and applies a Kalman filter [14]. Liu et al. proposed a method that detects a human based on a background model, and a greedy data association is performed [13]. Kobilarov et al. introduced a method that utilizes a Kanade Lucas Tomasi (KLT) tracker and performs data association with a Probabilistic Data Association Filter (PDAF) [15]. Song et al. proposed a method that rectifies only part of the outside image, and a human is tracked using a particle filter [16]. Kawasaki et al. combined static and dynamic background subtraction [17] for the human tracking. Delforouzi et al. introduced a method that can detect unknown objects based on a Training-Learning-Detection (TLD) scheme [18]. Yao et al. proposed a method that applies vertical vanishing point mapping to a normal image [23]. These methods reduce the dramatic changes in the bounding box position between consecutive frames if a human moves in the horizontal axis direction in the rectified image.

C) Several methods track humans in omnidirectional images without rectification [19]–[22]. Chen et al. proposed a method that tracks a human by Markov Random Fields (MRF) [19]. Zhang et al. proposed a method that extracts a human region by matching the foreground region and 3D-human model [22]. The foreground region is estimated by background subtraction. Rameau et al. introduced a method that tracks humans using a particle filter, the state vector of which is based on a sphere [20]. Cinaroglu et al. proposed a method that detects humans using a sliding window based on a Riemannian metric [21].

In A), ID switch often occurs due to following two factors. i) While images captured by the omnidirectional camera have serious distortions, most deep-learning-based detection methods estimate only a simple axis-aligned bounding box. The feature is adversely affected because the bounding box includes many background regions and the human’s angles vary when a human is captured from an oblique angle. Although semantic segmentation methods [24], [25] can be used for background reduction, they incur a heavy computational cost. ii) The position and size change dramatically because the distance metric is non-uniform. In B), the bounding box includes many background regions because the area around the center of the image is excessively expanded. In C), since most methods are not based on the tracking-by-detection approach, it is difficult to incorporate existing deep-learning-based detec-

| Camera       | Rectification | Tracking-by-detection |
|--------------|---------------|-----------------------|
| A) [2]–[7]   | Normal        | -                     |
| B) [12]–[18]| Omni          | Global rectification  | No                    |
| C) [19]–[22]| Omni          | -                     | No                    |
| Ours         | Omni          | Local rectification   | Yes                   |
tors that rely on normal images.

3. Proposed Method

The proposed method has two keypoints. i) We introduce local rectification, which rectifies only the human region in order to reduce the effect of background regions. ii) We describe the positions in the world coordinates where the distance metric is uniform in order to avoid dramatical changes in the position and shape.

The proposed method is based on the tracking-by-detection approach. The overall process of the proposed method is shown in Fig. 2. After the targeted humans are detected in the image coordinates (Sect. 3.1), the association metric is calculated (Sect. 3.2). Using the metric, these humans are tracked by data association (Section 3.3). The original feature of the proposed method is the association metric explained in Sect. 3.2. The regions are locally rectified and the features are extracted from the regions (Sect. 3.2.2). Also, the positions of the targeted humans are estimated in the world coordinates (Sect. 3.2.3).

Before describing the proposed method in detail, allow me to formalize the multiple human tracking. Let $o^f$ be an omnidirectional image at frame $f$. Let $T^f = (t_1^f, t_2^f, \ldots, t_{N_t}^f)$ be tracklets at frame $f$, where $t_i^f$ is the $i$-th tracklet. Let $B^f = (b_1^f, b_2^f, \ldots, b_{N_b}^f)$ be bounding boxes at frame $f$, where $b_j^f$ is the $j$-th bounding box. The multiple human tracking is formalized as the problem of sequentially estimating $T^f$ where $T^{f-1}$ and $o^f$ are given.

3.1 Human Detection in Image Coordinates

For each frame $f$, humans are detected by the deep-learning-based detection method. Each bounding box $b_j^f$ is defined as a normal rectangle, and is represented by $n = (x, y, w, h)$. $x$ and $y$ are the x-axis and y-axis in the upper left of the rectangle in an omnidirectional image. $w$ and $h$ are the width and height of the rectangle in an omnidirectional image. $n$ is calculated using a human detector. Although we chose SSD [11] for the detector in this work, any other detector can be used. The detector is trained using omnidirectional images in advance.

3.2 Proposed Association Metric

The local rectification (Sect. 3.2.2) and human position estimation in the world coordinates (Sect. 3.2.3) are performed using the bounding boxes obtained in Sect. 3.1. Before these estimations, the estimator is trained in advance (Sect. 3.2.1).

In the proposed method, the bounding box $b_j^f$ is defined as a rotated rectangle, and is represented by $r = (x_r, y_r, w_r, h_r, \phi)$. $x_r$ and $y_r$ are the x-axis and y-axis of the center position of the rectangle. $w_r$ and $h_r$ are the width and height of the rectangle. $\phi$ is an angle that is oriented clock-wise. The positive direction of the horizontal axis is defined as $\phi = 0$. The domain of $\phi$ is $0 \leq \phi < 2\pi$. $x_r$ and $y_r$ are set as the center of rotation.

Fig. 2 Overall process of the proposed method. First, the targeted humans are detected in the image coordinates. Second, the regions are locally rectified, and the features are extracted from the regions. At the same time, the positions of the targets are estimated in the world coordinates. Finally, these humans are tracked by data association using the features and positions of the targeted humans.
Sect. 3.1. The output of the estimator is rotated rectangle in the image coordinates. Then a normal rectangle and a rotated rectangle are calculated in the image coordinates. For each footpoint \( p \) the human between the image coordinates and the world coordinates. For each footprint \( p = (p_1, p_2) \) in the image coordinates, the following set of procedures is repeated (1 \( \leq p_1 \leq 1280, 1 \leq p_2 \leq 960 \)).

- A 3D-human model is virtually located in the world coordinates according to \( q \).
- A human contour that consists of points is calculated in the image coordinates using the located 3D-human model. Each vertex in the world coordinates is projected into the image coordinates by Eq. (1).
- A normal rectangle \( n = (x, y, w, h) \) and a rotated rectangle \( r = (x_r, y_r, w_r, h_r, \phi) \) are calculated in the image coordinates using the human contour. Both rectangles are calculated as a circumscribed rectangle of the human contour in terms of rectangle area minimization.
- The correspondence between a query vector \( n \) and a rotated rectangle \( r \) is registered. Also, the correspondence between the query vector \( n \) and a footpoint \( q \) is registered.

Since the query vectors obtained in these procedures do not cover all possible \( n \), we employ a nearest neighbor search. Kd-tree [29] is utilized to accelerate the nearest neighbor search. Therefore, if a query vector is input, we can obtain the corresponding rotated rectangle and footpoint efficiently.

3.2.2 Local Rectification and Feature Extraction in Image Coordinates

Local rectification consists of estimating the rotated rectangle and rotating it. The rotated rectangle \( r = (x_r, y_r, w_r, h_r, \phi) \) is calculated by the estimator using a query vector \( n \). The rotated rectangle \( r \) is rotated to be \( \phi = 0 \) in order that the footpoint is always in the lower part. The rotation is performed by the following rotation matrix:

\[
R = \begin{bmatrix}
\alpha & -\beta & (1 - \alpha)x_r - \beta y_r \\
\beta & \alpha & \beta x_r - (1 - \alpha)y_r \\
0 & 0 & 1
\end{bmatrix},
\]

where \( \alpha = \cos(-\alpha) \), \( \beta = \sin(-\alpha) \),

A virtual 3D-human model in the world coordinates and a human contour in the image coordinates are shown in Fig. 3. Human regions are associated via a footpoint of the human between the image coordinates and the world coordinates. For each footpoint \( p = (p_1, p_2) \) in the image coordinates, the following set of procedures is repeated (1 \( \leq p_1 \leq 1280, 1 \leq p_2 \leq 960 \)).

- A normal rectangle \( n = (x, y, w, h) \) and a rotated rectangle \( r \) are calculated in the image coordinates using the human contour. Both rectangles are calculated as a circumscribed rectangle of the human contour in terms of rectangle area minimization.

3.2.3 Human Position Estimation in World Coordinates

The footprint \( q = (q_1, q_2, 0) \) in the world coordinates is obtained from the query vector \( n \) through the estimator.

A 3D-human model is virtually located in the world coordinates according to \( q \).

A human contour that consists of points is calculated in the image coordinates using the located 3D-human model. Each vertex in the world coordinates is projected into the image coordinates by Eq. (1).

- A normal rectangle \( n = (x, y, w, h) \) and a rotated rectangle \( r = (x_r, y_r, w_r, h_r, \phi) \) are calculated in the image coordinates using the human contour. Both rectangles are calculated as a circumscribed rectangle of the human contour in terms of rectangle area minimization.

- The correspondence between a query vector \( n \) and a rotated rectangle \( r \) is registered. Also, the correspondence between the query vector \( n \) and a footpoint \( q \) is registered.

Since the query vectors obtained in these procedures do not cover all possible \( n \), we employ a nearest neighbor search. Kd-tree [29] is utilized to accelerate the nearest neighbor search. Therefore, if a query vector is input, we can obtain the corresponding rotated rectangle and footpoint efficiently.

Local rectification consists of estimating the rotated rectangle and rotating it. The rotated rectangle \( r = (x_r, y_r, w_r, h_r, \phi) \) is calculated by the estimator using a query vector \( n \). The rotated rectangle \( r \) is rotated to be \( \phi = 0 \) in order that the footpoint is always in the lower part. The rotation is performed by the following rotation matrix:

\[
R = \begin{bmatrix}
\alpha & -\beta & (1 - \alpha)x_r - \beta y_r \\
\beta & \alpha & \beta x_r - (1 - \alpha)y_r \\
0 & 0 & 1
\end{bmatrix},
\]

where \( \alpha = \cos(-\alpha) \), \( \beta = \sin(-\alpha) \),

A 3D-human model is virtually located in the world coordinates according to \( q \).

- A normal rectangle \( n = (x, y, w, h) \) and a rotated rectangle \( r \) are calculated in the image coordinates using the located 3D-human model. Each vertex in the world coordinates is projected into the image coordinates by Eq. (1).

- A normal rectangle \( n = (x, y, w, h) \) and a rotated rectangle \( r = (x_r, y_r, w_r, h_r, \phi) \) are calculated in the image coordinates using the human contour. Both rectangles are calculated as a circumscribed rectangle of the human contour in terms of rectangle area minimization.

- The correspondence between a query vector \( n \) and a rotated rectangle \( r \) is registered. Also, the correspondence between the query vector \( n \) and a footpoint \( q \) is registered.

Since the query vectors obtained in these procedures do not cover all possible \( n \), we employ a nearest neighbor search. Kd-tree [29] is utilized to accelerate the nearest neighbor search. Therefore, if a query vector is input, we can obtain the corresponding rotated rectangle and footpoint efficiently.

Local rectification consists of estimating the rotated rectangle and rotating it. The rotated rectangle \( r = (x_r, y_r, w_r, h_r, \phi) \) is calculated by the estimator using a query vector \( n \). The rotated rectangle \( r \) is rotated to be \( \phi = 0 \) in order that the footpoint is always in the lower part. The rotation is performed by the following rotation matrix:

\[
R = \begin{bmatrix}
\alpha & -\beta & (1 - \alpha)x_r - \beta y_r \\
\beta & \alpha & \beta x_r - (1 - \alpha)y_r \\
0 & 0 & 1
\end{bmatrix},
\]

where \( \alpha = \cos(-\alpha) \), \( \beta = \sin(-\alpha) \),

- A 3D-human model is virtually located in the world coordinates according to \( q \).
- A human contour that consists of points is calculated in the image coordinates using the located 3D-human model. Each vertex in the world coordinates is projected into the image coordinates by Eq. (1).
- A normal rectangle \( n = (x, y, w, h) \) and a rotated rectangle \( r = (x_r, y_r, w_r, h_r, \phi) \) are calculated in the image coordinates using the human contour. Both rectangles are calculated as a circumscribed rectangle of the human contour in terms of rectangle area minimization.
- The correspondence between a query vector \( n \) and a rotated rectangle \( r \) is registered. Also, the correspondence between the query vector \( n \) and a footpoint \( q \) is registered.

Since the query vectors obtained in these procedures do not cover all possible \( n \), we employ a nearest neighbor search. Kd-tree [29] is utilized to accelerate the nearest neighbor search. Therefore, if a query vector is input, we can obtain the corresponding rotated rectangle and footpoint efficiently.

Local rectification consists of estimating the rotated rectangle and rotating it. The rotated rectangle \( r = (x_r, y_r, w_r, h_r, \phi) \) is calculated by the estimator using a query vector \( n \). The rotated rectangle \( r \) is rotated to be \( \phi = 0 \) in order that the footpoint is always in the lower part. The rotation is performed by the following rotation matrix:

\[
R = \begin{bmatrix}
\alpha & -\beta & (1 - \alpha)x_r - \beta y_r \\
\beta & \alpha & \beta x_r - (1 - \alpha)y_r \\
0 & 0 & 1
\end{bmatrix},
\]

where \( \alpha = \cos(-\alpha) \), \( \beta = \sin(-\alpha) \),

- A 3D-human model is virtually located in the world coordinates according to \( q \).
- A human contour that consists of points is calculated in the image coordinates using the located 3D-human model. Each vertex in the world coordinates is projected into the image coordinates by Eq. (1).
- A normal rectangle \( n = (x, y, w, h) \) and a rotated rectangle \( r = (x_r, y_r, w_r, h_r, \phi) \) are calculated in the image coordinates using the human contour. Both rectangles are calculated as a circumscribed rectangle of the human contour in terms of rectangle area minimization.
- The correspondence between a query vector \( n \) and a rotated rectangle \( r \) is registered. Also, the correspondence between the query vector \( n \) and a footpoint \( q \) is registered.

Since the query vectors obtained in these procedures do not cover all possible \( n \), we employ a nearest neighbor search. Kd-tree [29] is utilized to accelerate the nearest neighbor search. Therefore, if a query vector is input, we can obtain the corresponding rotated rectangle and footpoint efficiently.

Local rectification consists of estimating the rotated rectangle and rotating it. The rotated rectangle \( r = (x_r, y_r, w_r, h_r, \phi) \) is calculated by the estimator using a query vector \( n \). The rotated rectangle \( r \) is rotated to be \( \phi = 0 \) in order that the footpoint is always in the lower part. The rotation is performed by the following rotation matrix:

\[
R = \begin{bmatrix}
\alpha & -\beta & (1 - \alpha)x_r - \beta y_r \\
\beta & \alpha & \beta x_r - (1 - \alpha)y_r \\
0 & 0 & 1
\end{bmatrix},
\]

where \( \alpha = \cos(-\alpha) \), \( \beta = \sin(-\alpha) \),

- A 3D-human model is virtually located in the world coordinates according to \( q \).
- A human contour that consists of points is calculated in the image coordinates using the located 3D-human model. Each vertex in the world coordinates is projected into the image coordinates by Eq. (1).
- A normal rectangle \( n = (x, y, w, h) \) and a rotated rectangle \( r = (x_r, y_r, w_r, h_r, \phi) \) are calculated in the image coordinates using the human contour. Both rectangles are calculated as a circumscribed rectangle of the human contour in terms of rectangle area minimization.
- The correspondence between a query vector \( n \) and a rotated rectangle \( r \) is registered. Also, the correspondence between the query vector \( n \) and a footpoint \( q \) is registered.

Since the query vectors obtained in these procedures do not cover all possible \( n \), we employ a nearest neighbor search. Kd-tree [29] is utilized to accelerate the nearest neighbor search. Therefore, if a query vector is input, we can obtain the corresponding rotated rectangle and footpoint efficiently.
in the first stage, the assignment is performed using $C_{pos}$ based on feature and is calculated. $C_{pos}$ and the bounding boxes at the current frame.

between the human tracking results at the previous frame $t_{f-1}$, where $t = (r, id)$. In the algorithm, a cost matrix $C(T^{f-1}, B^f)$ is calculated. $C(T^{f-1}, B^f)$ consists of $c(t_{f-1}^i, b_j^f)$ which is the cost between the tracklet $t_{f-1}^i$ and the bounding box $b_j^f$. Associated pairs are estimated by solving a linear assignment problem. It is solved efficiently using the Hungarian algorithm.

We calculate $c(t_{f-1}^i, b_j^f)$ based on feature/position, respectively. Therefore, we can obtain two cost matrices, $C_{feat}$ based on feature and $C_{pos}$ based on position. Although there are several ways of solving the linear assignment problem using two cost matrices, we introduce a two-step algorithm in this paper. First, we solve the linear assignment problem of $C_{feat}$. Second, for only unmatched tracking results in the first stage, the assignment is performed using $C_{pos}$. If $c(t_{f-1}^i, b_j^f) \leq \varepsilon$, $c(t_{f-1}^i, b_j^f) = \infty$ is set. $\varepsilon$ is a predefined parameter, and it is separately prepared for feature $\varepsilon_{feat}$ and position $\varepsilon_{pos}$.

### 3.4 Tracking Algorithm

The algorithm of the proposed method is shown in detail in Algorithm 1. The tracking algorithm was a simple online algorithm. The association algorithm and other handling (add/delete humans) were based on the DeepSORT algorithm [3].

### 4. Experiments

We conducted experiments on multiple human tracking in order to verify the effectiveness and efficiency of the proposed method.

#### 4.1 Experimental Conditions

We made two datasets that were created under a variety of conditions in the rooms we used for our experiments. We used a Panasonic WV-SF438 fisheye camera as the omnidirectional camera. The image resolution of this camera is $1280 \times 960$ with a video frame rate of 15 [fps]. The camera parameters were calculated by calibration using OCamCalib†. For the LargeRoom dataset, the area of the room was about $128 \text{ [m}^2\text{]}$ (8 [m] wide \times 16 [m] long). For the SmallRoom dataset, the area of the room was about $36 \text{ [m}^2\text{]}$ (4 [m] wide \times 9 [m] long). The details of the datasets are shown in Tables 2 and 3.

For the human detector (SSD), we used the default hyper-parameters. The detector was trained using 220,874 images captured in various rooms including SmallRoom. For the data association parameter, we changed the two parameters, $\varepsilon_{feat}$ ∈ [200, 300, 400] and $\varepsilon_{pos}$ ∈ [0.3, 0.5, 0.7, 0.9]. Then $\varepsilon_{feat} = 300$, $\varepsilon_{pos} = 0.7$ were determined using validation data.

The human detector was implemented in MXNet, and the feature extractor was implemented in PyTorch. We used a 4.20GHz Intel® Core™ i7-7700K CPU, a 32GB RAM, and a NVIDIA GeForce Titan X Pascal GPU.

For the evaluation metric, we used Multiple Object Tracking Accuracy (MOTA) metric. MOTA is a widely used and comprehensive metric that combines three error sources

---

### Table 2 LargeRoom dataset.

| Sequence ID | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|-------------|---|---|---|---|---|---|---|---|---|----|
| Camera ID   | 1 | 2 | 1 | 2 | 1 | 2 | 1 | 1 | 2 | 1  |
| The number of humans | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6  |
| Sequence length [sec] | 180 | 180 | 180 | 180 | 180 | 180 | 180 | 180 | 180 | 180 |

### Table 3 SmallRoom dataset.

| Sequence ID | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-------------|---|---|---|---|---|---|---|---|
| Camera ID   | 1 | 2 | 1 | 2 | 1 | 2 | 1 | 2 |
| The number of humans | 4 | 4 | 10 | 10 | 3 | 3 | 3 | 3 |
| Sequence length [sec] | 182 | 182 | 155 | 155 | 268 | 268 | 105 | 105 |

---

1††https://security.panasonic.com/products/wv-sf438/
2†https://sites.google.com/site/scarabotix/ocamcalib-toolbox
as follows:

\[
MOTA = 1 - (FN + IDs + FP) / DET,
\]

where FN, IDs, FP, and DET denote the total number false negatives, ID switches, false positives, and detections, respectively. The MOTA score ranges from \(-\infty\) to 100. More details about these metrics are described in another paper [34]. For bounding boxes, since the ground truth used a normal rectangle, the proposed method estimated tracking results in the normal rectangle \(n = (x, y, w, h)\). We made the ground truths at 1 [fps] because the sequences include a large number of frames. Therefore, the tracking results were only evaluated for the annotated frames.

### 4.2 Evaluation of Multiple Human Tracking

We evaluated each proposed function and their combinations. Also, we verified that the proposed method solves the existing problems. A summary of the tracking results is shown in Table 4. The MOTA in the table is an average of all the sequences. For the feature, “without local rectification” or “with local rectification” was used. For the position, “rectangle in image coordinates” or “position in world coordinates” was used. More details are shown in Appendix A.

#### 4.2.1 Local Rectification

We evaluated the effects of local rectification. Let us compare (no rectification & -) to (local rectification & -). First, we present the results for LargeRoom. At 1 [fps], MOTA improves +1.8 (8.4 vs 10.2). At 15 [fps], MOTA is almost the same (−3.5 vs −3.4). Next, we present the results for SmallRoom. At 1 [fps], MOTA improves +0.8 (51.0 vs 51.8). At 15 [fps], MOTA is almost the same (48.7 vs 48.5). Local rectification is shown to be effective particularly at a low frame rate. Local rectification is just as effective as no rectification at a normal frame rate. At 1 [fps], local rectification is more effective in the case of LargeRoom than for SmallRoom. (LargeRoom:+1.8 vs SmallRoom:+0.8)

#### 4.2.2 World Coordinates Representation

We then evaluated the world coordinates representation. Let us compare (- & rectangle in image) to (- & position in world). First, we present the results for LargeRoom. At 1 [fps], MOTA improves +18.6 (−10.5 vs 8.1). At 15 [fps], MOTA is almost the same (−1.4 vs −1.1). Next, we present the results for SmallRoom. At 1 [fps], MOTA improves +28.6 (24.5 vs 53.1). At 15 [fps], MOTA is almost the same (49.9 vs 49.7). The position in world coordinates is particularly effective at a low frame rate. The position in world coordinates is just as effective as the rectangle in image coordinates at a normal frame rate. At 1 [fps], the position in world coordinates is more accurate in the case of SmallRoom than for LargeRoom. (LargeRoom:+18.6 vs SmallRoom:+28.6)

#### 4.2.3 Tendency Analysis

We analyzed those cases where the proposed method was particularly effective. The evaluation metric is MOTA which is regarded as the normalized ID switch. We used all sequences of all frame rates in the LargeRoom dataset. For analysis, the image coordinates \((X, Y)\) are converted to the polar coordinates \((\theta, r)\). The center point (640, 480) in the

![Fig. 4 MOTA with respect to \(\theta\) and \(r\).](image-url)
image coordinates is set to point to the origin in the polar coordinates. $\theta$ [rad] is the angle made by the point of origin, and $r$ [pixel] is the distance from the point of origin. Figure 4 shows the analysis results. The horizontal axis denotes $\theta$ and $r$, and the vertical axis denotes MOTA.

\(\theta\): We analyzed those cases where local rectification was particularly effective. Figure 4 (a) shows a comparison of MOTA between the cases where the human was vertical/horizontal ($-9/8\pi < \theta \leq -7/8\pi, -5/8\pi < \theta \leq -3/8\pi, -1/8\pi < \theta \leq 1/8\pi, 3/8\pi < \theta \leq 5/8\pi$) and where the human was captured at an oblique angle ($-7/8\pi < \theta \leq -5/8\pi, -3/8\pi < \theta \leq -1/8\pi, 1/8\pi < \theta \leq 3/8\pi, 5/8\pi < \theta \leq 7/8\pi$). When the human was vertical/horizontal, MOTA improved $+0.011 (-0.566 \text{ vs } -0.555)$. When the human was captured at an oblique angle, MOTA improved $+0.031 (0.205 \text{ vs } 0.236)$. Local rectification was more effective in the case where the human was captured at an oblique angle compared to the case where the human was vertical/horizontal. This is owing to background reduction.

\(r\): We analyzed those cases where the world coordinates representation was particularly effective. Figure 4 (b) shows a comparison of MOTA between the case where the human was near the origin ($0 < r \leq 200$) and where the human was close to the outside ($300 < r \leq 500$). When the human was near the origin, MOTA improved $+17.703 (-17.81 \text{ vs } -0.107)$. When the human was close to the outside, MOTA improved $+1.757 (-2.521 \text{ vs } -0.764)$. The world coordinates representation was more effective in the case where the human was near the origin than where the human was close to the outside.

4.2.4 Local Rectification and World Coordinates Representation

We evaluated the combination of local rectification and world coordinates representation at 1 [fps]. Let us compare (local rectification & -) to (- & position in world).

In the case of LargeRoom, local rectification was more effective than the world coordinates representation ($10.2 \text{ vs } 8.1$). This is because local rectification is effective where the human is located around the outside of the image as described in Sect. 4.2.3. When combining local rectification with world coordinates representation, MOTA improved 1.5. This is because the world coordinates representation...
is effective where the human is located around the center of the image as described in Sect. 4.2.3.

In the case of SmallRoom, conversely, the world coordinates representation is more effective than local rectification (51.8 vs 53.3). This is because the world coordinates representation is effective in cases where humans are located around the center of the image as described in Sect. 4.2.3. However, combining local rectification and the world coordinates representation improved MOTA by only 0.2. The effectiveness of local rectification is low because the background area is small when humans are located around the center of the image.

Therefore, the proposed method (combining local rectification and the world coordinates representation) is effective, particularly in the case of a low frame rate (1 fps) and a large room (LargeRoom).

### 4.3 Evaluation of Computational Time

We evaluated the computational time needed for estimating rotated rectangles and human positions in the world coordinates. Table 5 shows the computational time which is the average of all frames in sequence 1 in the SmallRoom dataset. The computational times for rotated rectangles and human position estimation are 1.0 and 0.7 [msec], respectively. These times are very fast and have little impact on the overall tracking time. There are 4 humans in Sequence 1; therefore, it takes 0.25 [msec/human] to estimate the rotated rectangle and 0.18 [msec/human] to estimate the human position. The computational time needed for human detection accounts for a large percentage in the overall system. We can reduce it by employing other fast detectors or downsizing input images.

### 4.4 Tracking Examples

Some tracking examples are shown in Fig. 5. “Previous” denotes (no rectification & rectangle in image) and “Proposed” denotes (local rectification & position in world). #(number) denotes the frame number. Red circles denote ID switch and yellow circles denote the prevention of ID switch. ID switches are prevented in “Proposed” in some frames.

### 4.5 Discussion

We conducted an additional experiment using a more complex sequence in which more humans are moving freely. In the additional SmallRoom2 dataset, more humans (11) are moving in longer sequence lengths (169 [sec]), compared with sequences 3 and 4 in the SmallRoom dataset. The area of SmallRoom2 is the same as that of SmallRoom. The tracking results for SmallRoom2 are shown in Table 4. In addition to other datasets, the proposed method (combining local rectification and the world coordinates representation) is effective, particularly in case of a low frame rate (1 fps).

### 5. Conclusion

In this paper, we proposed a new method that accurately tracks humans using an association metric for omnidirectional images. The key ideas of the proposed method are as follows: i) Reducing the background regions by local rectification. ii) Describing the human position in the world coordinate system. In the experiments, we confirmed that the proposed method is effective, particularly at a low frame rate. MOTA improved 3.3 in the LargeRoom dataset and MOTA improved 2.3 in the SmallRoom dataset. It takes only 0.43 [msec] per human in a frame to calculate the proposed association metrics. In the future, it will be important not only reducing background regions and normalize angles but also to capture human appearance itself. Additionally, the same idea as the proposed method will be generalized for standard cameras when perspective effects and lens distortions are quite noticeable.

### References

[1] R.T. Collins, A.J. Lipton, T. Kanade, H. Fujiyoshi, D. Duggins, Y. Tsin, D. Tolliver, N. Enomoto, O. Hasegawa, P. Burt, et al., “A system for video surveillance and monitoring,” VSAM final report, pp.1–68, March 2000.
[2] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, “Simple online and realtime tracking,” Proceedings of the 23rd IEEE International Conference on Image Processing (ICIP), pp.3464–3468, Sept. 2016.
[3] N. Wojke, A. Bewley, and D. Paulus, “Simple online and real-time tracking with a deep association metric,” Proceedings of the 24th IEEE International Conference on Image Processing (ICIP), pp.3645–3649, Sept. 2017.
[4] C. Kim, F. Li, A. Ciptadi, and J.M. Rehg, “Multiple hypothesis tracking revisited,” Proceedings of the 15th IEEE International Conference on Computer Vision (ICCV), pp.4696–4704, Dec. 2015.
[5] Q. Chu, W. Ouyang, H. Li, X. Wang, B. Liu, and N. Yu, “Online multi-object tracking using cnn-based single object tracker with spatial-temporal attention mechanism,” Proceedings of the 16th IEEE International Conference on Computer Vision (ICCV), pp.4846–4855, Oct. 2017.
[6] A. Sadeghian, A. Alahi, and S. Savarese, “Tracking the untrackable: Learning to track multiple cues with long-term dependencies,” Proceedings of the 16th IEEE International Conference on Computer Vision (ICCV), pp.300–311, Oct. 2017.
[7] J. Son, M. Baek, M. Cho, and B. Han, “Multi-object tracking with quadruplet convolutional neural networks,” Proceedings of the 30th IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.5620–5629, July 2017.
[8] L. Zhang, Y. Li, and R. Nevatia, “Global data association for multi-object tracking using network flows,” Proceedings of the 21st IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.1–8, June 2008.
[9] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.39, no.6, pp.1137–1149, 2017.
Appendix A: Details of Tracking Results

A summary of the tracking results is shown in Tables A-1, A-2, A-3 and A-4. We evaluated ID switches (IDs), Fragmentation (FM), Recall (Rec), Precision (Prc), Multiple Object Tracking Accuracy (MOTA) and Multiple Object Tracking Precision (MOTP). The numbers of IDs and FM are the sum of all the sequences, respectively. Rec, Prc, MOTA, and MOTP are the average of all the sequences, respectively.

A.1 LargeRoom Dataset

No rectification vs Local rectification: Let us compare (no rectification & -) to (local rectification & -). In 1 [fps], MOTA improves (8.4 vs 10.2). This is because IDs decrease (1279 vs 1101) while retaining Recall and Precision. Local rectification is effective, particularly at a low frame rate. At 15 [fps], MOTA is almost the same (−3.5 vs −3.4).

Rectangle in image vs Position in world: Let us compare (- & rectangle in image) to (- & position in world). At 1 [fps], MOTA improves (−10.5 vs 8.1). This is because Recall and Precision are improved significantly (Recall: 13.9 vs 41.8, Precision: 42.2 vs 63.5). The position in world is effective, particularly at a low frame rate. At 15 [fps], MOTA is almost the same (−1.4 vs −1.1).

Previous method vs Proposed method: Let us compare (three previous method) to (local rectification & position in
Table A.1 Details of tracking results on LargeRoom dataset with 1 [fps].

| Feature    | Position         | Rcll ↑ | Prcn ↑ | IDs ↓ | FM ↓ | MOTA ↑ | MOTP ↑ |
|------------|------------------|--------|--------|-------|------|--------|--------|
| -          | no rectification | 48.9   | 63.4   | 1279  | 1587 | 8.4    | 30.7   |
| SORT [2]   | rectangle in image | 13.9   | 42.2   | 592   | 638  | -10.5  | 30.9   |
| DeepSORT [3] | rectangle in image | 23.1   | 52.2   | 669   | 911  | -4.3   | 30.6   |
| Proposed   | local rectification | 48.9   | 63.5   | 1101  | 1580 | 10.2   | 30.7   |
| -          | position in world | 41.8   | 63.5   | 1018  | 1384 | 8.1    | 30.6   |
| Proposed   | local rectification | 44.3   | 64.2   | 814   | 1409 | 11.7   | 30.6   |

Table A.2 Details of tracking results on SmallRoom dataset with 1 [fps].

| Feature    | Position         | Rcll ↑ | Prcn ↑ | IDs ↓ | FM ↓ | MOTA ↑ | MOTP ↑ |
|------------|------------------|--------|--------|-------|------|--------|--------|
| -          | no rectification | 73.2   | 80.0   | 247   | 432  | 51.0   | 22.2   |
| SORT [2]   | rectangle in image | 46.3   | 76.2   | 397   | 547  | 24.5   | 22.2   |
| DeepSORT [3] | rectangle in image | 52.5   | 77.4   | 352   | 520  | 30.9   | 22.1   |
| Proposed   | local rectification | 73.3   | 80.1   | 211   | 434  | 51.8   | 22.2   |
| -          | position in world | 72.3   | 81.2   | 161   | 424  | 53.1   | 22.2   |
| Proposed   | local rectification | 72.5   | 80.8   | 117   | 426  | 53.3   | 22.2   |

Table A.3 Details of tracking results on LargeRoom dataset with 15 [fps].

| Feature    | Position         | Rcll ↑ | Prcn ↑ | IDs ↓ | FM ↓ | MOTA ↑ | MOTP ↑ |
|------------|------------------|--------|--------|-------|------|--------|--------|
| -          | no rectification | 49.5   | 63.4   | 2576  | 1606 | -3.5   | 30.7   |
| SORT [2]   | rectangle in image | 47.0   | 64.7   | 2322  | 1607 | -1.4   | 30.6   |
| DeepSORT [3] | rectangle in image | 48.4   | 64.3   | 1634  | 1593 | 6.0    | 30.6   |
| Proposed   | local rectification | 49.5   | 63.4   | 2576  | 1604 | -3.4   | 30.7   |
| -          | position in world | 47.6   | 64.5   | 2386  | 1601 | -1.1   | 30.6   |
| Proposed   | local rectification | 48.6   | 64.1   | 1561  | 1584 | 6.6    | 30.6   |

Table A.4 Details of tracking results on SmallRoom dataset with 15 [fps].

| Feature    | Position         | Rcll ↑ | Prcn ↑ | IDs ↓ | FM ↓ | MOTA ↑ | MOTP ↑ |
|------------|------------------|--------|--------|-------|------|--------|--------|
| -          | no rectification | 74.2   | 79.7   | 427   | 438  | 48.7   | 22.2   |
| SORT [2]   | rectangle in image | 73.4   | 80.7   | 347   | 450  | 49.9   | 22.1   |
| DeepSORT [3] | rectangle in image | 74.0   | 80.2   | 193   | 441  | 52.3   | 22.2   |
| Proposed   | local rectification | 74.2   | 79.7   | 435   | 438  | 48.5   | 22.2   |
| -          | position in world | 73.4   | 80.6   | 371   | 455  | 49.7   | 22.1   |
| Proposed   | local rectification | 73.9   | 80.1   | 206   | 443  | 32.0   | 22.2   |

A.2 SmallRoom Dataset

No rectification vs Local rectification: Let us compare (no rectification & -) to (local rectification & -). At 1 [fps], MOTA improves (8.4 vs 11.7). Combining local rectification and position in world is effective, particularly at a low frame rate. At 15 [fps], MOTA slightly improves (6.0 vs 6.6).

Hitoshi Nishimura  received the B.E. and M.E. degrees in engineering from Kobe University, Japan, in 2013 and 2015, respectively. He joined KDDI Research, Inc. in 2016 and has been engaged in the research of human tracking and action recognition. He entered Nagoya University in 2018. He is a member of ITE.
Naoya Makibuchi received the B.E. and M.E. degrees in engineering from Tokyo Institute of Technology, Japan, in 2009 and 2011, respectively. In 2011, he joined KDDI Co. Ltd. and has been engaged in the research and development of person tracking and augmented reality. He is a research engineer of the Software Integration Laboratory at KDDI Research, Inc.

Kazuyuki Tasaka received his B.E. degree from Niihama National College of Technology in 2002, and his M.E. and Ph.D. degree from Nara Institute of Science and Technology in 2004 and 2010, respectively. Since joining KDDI Research, Inc. in 2004, he has worked in the field of network architecture, communication protocols and context recognition. Now, he is a R&D manager in the Media Recognition Laboratory and is a member of IEICE and IPSJ.

Yasutomo Kawanishi received his BEng and MEng degrees in Engineering and a Ph.D. degree in Informatics from Kyoto University, Japan, in 2006, 2008, and 2012, respectively. He became a Post Doctoral Fellow at Kyoto University, Japan in 2012. He moved to Nagoya University, Japan as a Designated Assistant Professor in 2014. Since 2015, he has been an Assistant Professor at Nagoya University, Japan. His research interests are Computer Vision techniques, especially Pedestrian Detection, Tracking, and Retrieval, for surveillance and in-vehicle videos. He received the best paper award from SPC2009, and Young Researcher Award from IEEE ITS Society Nagoya Chapter. He is a member of IEICE, IIEEJ, and IEEE.

Hiroshi Murase received the B.Eng., M.Eng., and Ph.D. in engineering from Nagoya University, Japan. From 1980 to 2003 he was a research scientist at the Nippon Telegraph and Telephone Corporation (NTT). He has been a professor of Nagoya University since 2003. He was awarded the IEEE CVPR Best Paper Award in 1994, the IEICE Distinguished Achievement and Contributions Award in 2018. He received Shijyu-hosho (the Medal with Purple Ribbon) in 2012. His research interests include computer vision, pattern recognition, and multimedia information processing. He is a fellow of IEEE, and a fellow of IPSJ.