

Received September 2, 2021, accepted September 7, 2021, date of publication September 10, 2021, date of current version September 17, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3111917

Pedestrian Trajectory Prediction Based on Transfer Learning for Human-Following Mobile Robots

RINA AKABANE¹ AND YUKA KATO², (Member, IEEE)

¹Graduate School of Science, Tokyo Woman's Christian University, Suginami, Tokyo 1678585, Japan

²Department of Information and Science, Tokyo Woman's Christian University, Suginami, Tokyo 1678585, Japan

Corresponding author: Yuka Kato (yuka@lab.twcu.ac.jp)

This work was supported in part by the Telecommunications Advancement Foundation, Grants-in-Aid for Scientific Research (KAKENHI), Japan Society for the Promotion of Science, under Grant 20K11776 and Grant 20K12011, and in part by the Cooperative Research Project, Research Institute of Electrical Communication, Tohoku University.

ABSTRACT Recent developments in the field of service robots have led to a renewed interest in human-robot coexistence environments such as at home and office. In this regard, this study focuses on one of such service robots, a human-following mobile robot. In particular, we consider predicting the future trajectory of pedestrians using a machine learning algorithm to improve the accuracy of tracking people. Massive trajectory data is required in existing methods to train the prediction model; however, collecting a sufficient amount of data in general public places before providing services is challenging. Therefore, in this study, we propose a trajectory prediction method based on extracting similar datasets from a large-size dataset and generating a pre-trained prediction model using the extracting datasets. We express the data features in the source and target environments as probability distributions and evaluate the divergence between them. Specifically, the dataset features are expressed as a multidimensional Gaussian distribution and discrete distribution of samples. Then, similarities using the Kullback-Leibler divergence are compared. To verify the effectiveness of the proposed method, we compare the prediction results of the LSTM-based algorithm with those obtained by extracting multiple source datasets from a large dataset and training prediction models using these datasets. The result shows that the proposed method makes it possible to construct an appropriate prediction model with high accuracy in trajectory prediction.

INDEX TERMS Human-robot coexistence environments, intelligent robots, Kullback-Leibler divergence, LSTM, machine learning, predictive models, service robots, trajectory prediction, transfer learning.


I. INTRODUCTION

Recent developments in service robots have led to a renewed interest in human-robot coexistence environments such as at home, office, and public facilities. Service robots are defined as robots providing various kinds of services to people, e.g., a robot delivering relief supplies in a disaster, assisting people with physical disabilities, and guiding people in public facilities. An example is a human-following mobile robot that constantly detects the target pedestrian and follows the person. Human-following mobile robots can be used in various situations, such as to carry heavy baggage or guide through indoor/outdoor public facilities. Such robots generally track a person using a motion model of the robot and a Bayesian filter based on an observation model, e.g., using sensing data from

light detection and ranging (LiDAR). However, frequently the mobile robot loses track of the person due to occlusion by obstacles. Therefore, the difficulty of tracking a target person, especially in a crowded environment, has become a challenging problem.

To solve this problem, we consider predicting the future trajectory of the target pedestrian using a machine learning algorithm for the tracking control of a mobile robot. The proposed method enables following a target person, even if an obstacle interrupts the target and the robot. Moreover, the approach can detect and follow the target by predicting the future position of the target. We incorporate this approach into a following robot. Fig. 1 shows the usage image for the trajectory prediction result in the following robot.

Until now, several methods have been proposed for pedestrian trajectory prediction. One approach is to formulate

The associate editor coordinating the review of this manuscript and approving it for publication was Tallha Akram .

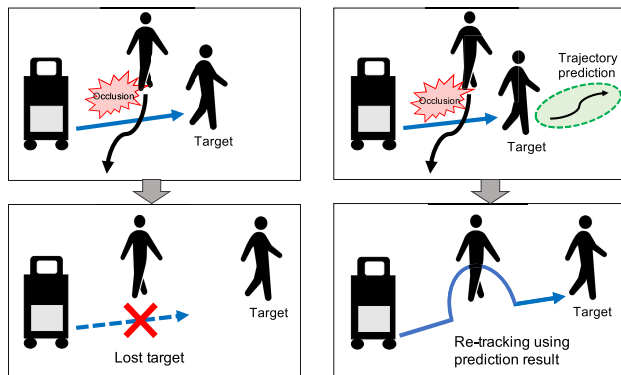


FIGURE 1. Usage image of the trajectory prediction result. The method predict the future trajectory of a target pedestrian based on past positions and uses the prediction result for the tracking control of robots.

pedestrian dynamics using a mathematical model and estimate the trajectories according to the pedestrian model. In this regard, Helbing's social force model (SFM) [1] is a well-known seminal study. We have also been conducting research on formulating a pedestrian model in a human-robot coexisting environment [2] based on SFM. The other approach is to estimate the trajectories in the target environment using a machine learning algorithm considering massive observation data. There are many successful application of Deep Learning methods to several verticals, including communications and networking [3], [4]. In addition, many studies have been done on trajectory data mining (trajectory similarity search, trajectory classification, trajectory quality improvement, etc.) [5].

The advantage of the modeling method is that no massive observation data is required, and the method is straightforward. However, accurate modeling is required, which is a complicated task that can cause numerous problems. For instance, the models rely on a large number of parameter values that must be estimated. Thus, adjusting the model for an accurate prediction becomes a challenging task. In contrast, the machine learning approach can generate accurate prediction models even in relatively complex environments. Nevertheless, machine learning-based methods require collecting massive observation data in advance. In this study, we adopt the latter methods because we consider tasks in complex environments where the tracking person is frequently lost. Thus, massive trajectory data is required to train the prediction model; however, collecting a sufficient amount of data is difficult because of the following requirements: (i) avoid deploying additional types of IoT devices to the environment and (ii) avoid acquiring sensing data in the target environment in advance. Consequently, there is a growing interest in developing a method to predict trajectories using a prediction model pre-trained with different sensors in different environments from the prediction target [6].

Based on the above background, in this study, we propose a method for predicting trajectories by constructing the training data from a large open dataset containing a large number of pedestrian trajectories in various situations. The

TABLE 1. List of the acronyms used in the paper.

Acronym	Definition
ADE	Average Displacement Error
CNN	Convolutional Neural Network
FDE	Final Displacement Error
GAN	Generative Adversarial Networks
KF	Kalman Filter
KL divergence	Kullback-Leibler divergence
LiDAR	Light Detection And Ranging
LSTM	Long Short-Term Memory
RNN	Recurrent Neural Network
SFM	Social Force Model

proposed method generates a pre-trained prediction model using the training data. This approach relies on the method for constructing the source data in transfer learning. More specifically, we first express the properties of the feature space of the source/target datasets as a function. Subsequently, an appropriate source dataset for training is selected by comparing the similarity between these functions. After that, we predict the trajectory by using the prediction model pre-trained with the selected dataset. For expressing the feature space, we assume a two-dimensional feature space with the velocity and angular velocity norms of walking as features, referring to the study on classifying pedestrian trajectories [7]. Whereas, for expressing the spatial properties, we use a probability density function (e.g., Gaussian distribution) or a probability mass function. The similarity between functions is compared using the Kullback-Leibler divergence (KL divergence) [8]. From the above, a pre-trained prediction model is generated by extracting the source dataset with the highest similarity to the target dataset and inputting the dataset to the model as the training data. The prediction model (i.e., pre-trained model) generated by the proposed method is shown in Fig. 2. We develop the prediction model using a large dataset in advance (in the offline process) and bring it to the actual environment where the robot performs the task. As a result, we enable adaptive tracking control to various environments (in the online process).

The contribution of this study can be summarized as follows:

- A method to generate pre-trained accurate prediction models for pedestrian trajectory prediction is developed.
- How the similarity differences in the training data affect the trajectory prediction accuracy when using the pre-trained prediction model is evaluated.

The properties of the proposed scheme are:

Adaptive: We can apply it to various environments where data collection is difficult, such as public spaces, because it only requires a small number of samples to be collected in the target environment.

Efficient: The model can be trained with fewer data by extracting datasets that are similar to the target environments. As a result, the training time is reduced.

Accurate: It improves prediction accuracy by constructing appropriate training data.

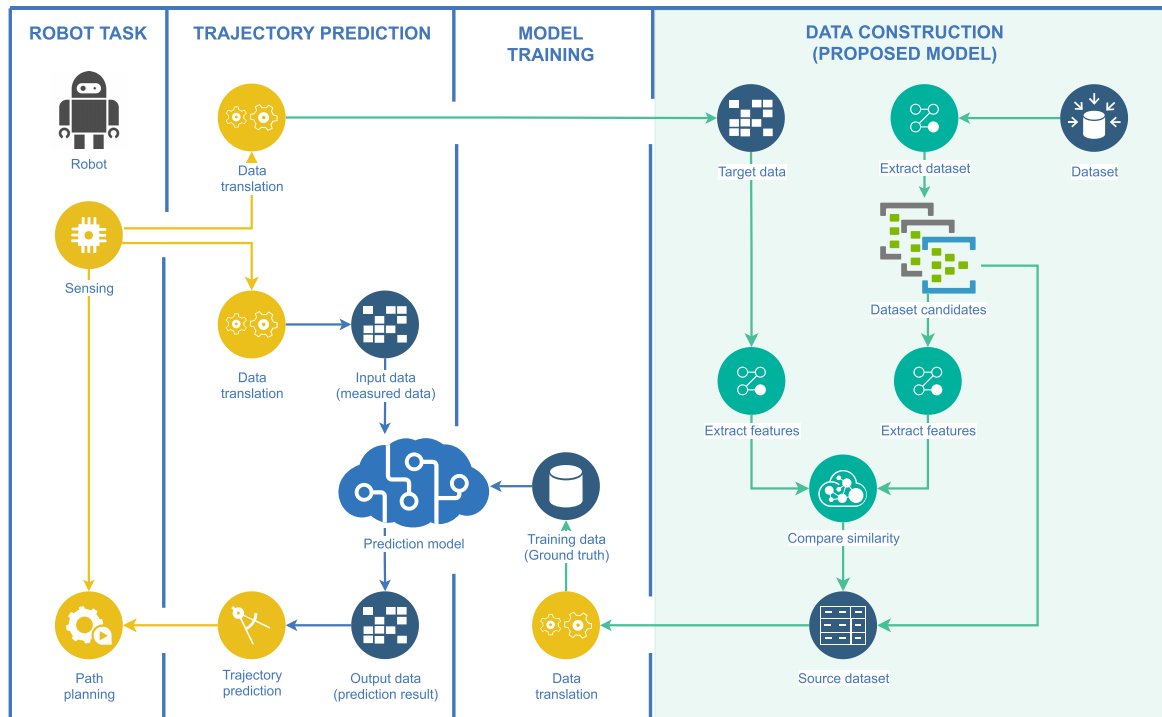


FIGURE 2. Usage image of the pre-trained pedestrian model generated by the proposed method. The prediction model is generated using a large dataset in advance (in the offline process) and deployed to the actual environment where the robot performs the task. Adaptive tracking control to various environments is performed (in the online process).

The remaining of the paper is organized as follows: We briefly present and discuss related work in Section II. After that, basic preliminary concepts and methods (e.g., formulation of KL divergence and explanation of the machine learning algorithm used in the proposed method) are presented in Section III. The proposed modeling approach and the corresponding algorithm are described in Section IV and V, respectively. Finally, we discuss the experimental results in Section VI and conclude the study in Section VII. For the sake of readability, the acronyms used in this paper are summarized in Table 1.

II. RELATED WORK

A. METHOD PREDICTING PEDESTRIAN TRAJECTORY

Pedestrian trajectory prediction (a method predicting future trajectories of pedestrians based on past trajectories) is a research topic that has been actively studied recently, especially in the vision field. Various studies based on a Bayesian filter have been presented. These methods perform short-term predictions of trajectories for tracking targets, for example, using gait velocity and generating an extended Kalman filter (KF) by integrating the proximity effect of mutual interaction between persons with the motion model [9]. A method combining multiple prediction models generated independently by LSTM (long short-term memory) was described in [10]. Predicting the target positions in the next cycle based on three features (appearance, motion and interaction) was proposed in [11]. Moreover, in [12], a method estimating the

state of the target (i.e., active, inactive, tracked, and lost) using a Markov decision process and determining the model for each state was presented. Finally, a method predicting a tracking candidate in real-time by combining it with a pedestrian detection based on a convolutional neural network (CNN) was described in [13]. Although these methods design various schemes to redetect targets, a long-term prediction is still a challenging problem.

Recently, several studies have also considered methods generating prediction models based on recurrent neural network (RNN) and predicting trajectories as sequence data [14], e.g., Social LSTM [15], Social GAN [16]. Social LSTM generates a model of each trajectory (i.e., sequence data) by LSTM and expresses the effect of the mutual interaction between persons by sharing those states of the hidden layers. Social GAN is the extension of Social LSTM that uses generative adversarial networks (GAN), which can generate multiple trajectories simultaneously. Trajectory Forecasting Challenge¹ has also been held.

RNN-based methods have been actively studied in natural language processing and have achieved rapid improvements in prediction accuracy. These methods can be applied to various domains related to the prediction/generation of sequence data. In this study, we also use these methods to predict future trajectories.

¹<http://trajnet.stanford.edu>

B. STUDY ON TRANSFER LEARNING

Transfer learning is a method adapting a pre-trained machine learning model that has been trained for a task in one domain to other tasks in another related domain. It is a branch of machine learning, also known as inductive learning, domain adaptation, learning to learn, and has received increasing interest in recent years to solve the difficulties in collecting sufficient training data for supervised learning. Similar ideas include semi-supervised learning, which prepares a small number of labeled data and trains the models using a large number of unlabeled. Moreover, active learning trains models using only some data with a high learning effect. In particular, theoretical methods of transfer learning can be found in covariate shift [17] and sample selection bias [18], while for practical ones, there are applications to image classification tasks [19] and document classification tasks [20], [21].

In this study, to solve the difficulties in collecting a sufficient amount of data in general public space before providing services, we consider using a pre-trained model that has been generated in advance with data acquired in other environments to predict pedestrian trajectories.

C. METHOD OF HUMAN-TRACKING

Sensing data acquired from a LiDAR are often used in a robot for human-tracking. LiDARs are used in various environments because the measurement accuracy is high. The sensing results do not tend to be affected by environmental conditions such as sunlight. Moreover, estimated target sizes, locations, and moving directions can be detected with single LiDAR. As this method does not use a camera, it is not possible to obtain visual information (e.g., generic object recognition). However, the methods using a LiDAR detect and track the target person through various mechanisms. For example, a method for simultaneous mobile robot localization and human-tracking using a LiDAR is proposed [22]. In the method, a human body is modeled by a cylinder, while the error distance between the circle as the horizontal cross-section of the body and the actual sensing value is used for likelihood. The probability distribution of the human location is calculated using the likelihood.

In this study, we use these types of methods for human-tracking by a robot as well. Moreover, we try to improve the accuracy of the tracking by combining them with the prediction methods described above.

The relative advantages of the proposed method are summarized in Table 2 along with their main distinctive characteristics as follows: (A) applicable to a variety of environments; (B) trainable on the small size of data; (C) ease of model construction; (D) prediction accuracy; (E) available in crowded environments. The summary shows that only our method meets all requirements.

III. THEORETICAL BACKGROUND

To generate an appropriate pre-trained model by transfer learning, we consider selecting an appropriate source dataset

TABLE 2. Summary of previous works along with their main distinctive characteristics.

	(A)	(B)	(C)	(D)	(E)
SFM [1]	✓	✓			
KF [9]	✓	✓			
with RNN/LSTM [10] [11]	✓			✓	
[12]	✓			✓	
with CNN [13]			✓	✓	
RNN [14]			✓	✓	
S-LSTM/GAN [15] [16]			✓	✓	✓
Transfer Learning [17] [18]		✓	✓		
Our method	✓	✓	✓	✓	✓

(A) Applicable to a variety of environments.

(B) Trainable on the small size of dataset.

(C) Ease of model construction.

(D) Prediction accuracy.

(E) Available in crowded environments.

by expressing the source and destination feature spaces with a function and comparing the similarities between the functions. This section explains the Gaussian distribution used as an example of the function and the Kullback-Leibler divergence used as an example for measuring a similarity between functions. Moreover, we also explain the outline of Social LSTM, which is a machine learning algorithm used to generate the prediction model.

A. GAUSSIAN DISTRIBUTION

In a wide range of fields such as statistics, machine learning, and pattern recognition, probabilistic modeling is often used to estimate the trend of a massive amount of data using probability distributions. There are various types of distributions used in such processes, for example, Bernoulli, binomial, and categorical distributions as discrete probability distributions, while beta, Dirichlet, gamma, and Gaussian distributions as continuous probability distributions.

Among them, the Gaussian distribution is one of the most representative probability distributions. When the mean and variance are determined using the Gaussian distribution, the information content (or the entropy) is the minimum (or the maximum). Thus, a good approximation is stably obtained even if the sample size is small.

The probability density function of multidimensional Gaussian distribution for $\mathbf{x} \in \mathbb{R}^D$ is expressed as follows:

$$\mathcal{N}(\mathbf{x} | \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^D |\boldsymbol{\Sigma}|}} \exp \left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right\} \quad (1)$$

where $\boldsymbol{\mu} \in \mathbb{R}^D$ is the mean vector and $\boldsymbol{\Sigma}$ is the covariance matrix. Specifically, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are expressed as

$$\boldsymbol{\mu} = (\mu_1, \mu_2, \dots, \mu_D)^\top, \quad (2)$$

$$\boldsymbol{\Sigma} = \begin{pmatrix} \sigma_1^2 & \dots & \sigma_{1i} & \dots & \sigma_{1D} \\ \vdots & \ddots & & & \vdots \\ \sigma_{i1} & \dots & \sigma_i^2 & \dots & \sigma_{iD} \\ \vdots & & & \ddots & \vdots \\ \sigma_{D1} & \dots & \sigma_{Di} & \dots & \sigma_D^2 \end{pmatrix}. \quad (3)$$

B. KL DIVERGENCE

We explain the KL divergence as a metric to evaluate the similarity between probability distributions.

1) KL DIVERGENCE FOR CONTINUOUS PROBABILITY DISTRIBUTION

Let $p(x)$ and $q(x)$ denote probability density functions. The KL divergence is defined as

$$\mathcal{D}_{KL}(p \parallel q) = \int p(x) \log \frac{p(x)}{q(x)} dx. \tag{4}$$

For any given pair of probability distributions, $\mathcal{D}_{KL} \geq 0$, and

$$\mathcal{D}_{KL}(p \parallel q) = 0 \Leftrightarrow \forall x, p(x) = q(x). \tag{5}$$

In general, $\mathcal{D}_{KL}(p \parallel q) \neq \mathcal{D}_{KL}(q \parallel p)$, which does not satisfy the mathematical metric axioms.

2) KL DIVERGENCE FOR DISCRETE PROBABILITY DISTRIBUTION

Let $P(s)$ and $Q(s)$ denote probability mass functions on a discrete space. The KL divergence is defined as

$$\mathcal{D}_{KL}(P \parallel Q) = \sum_{s \in M} P(s) \log \frac{P(s)}{Q(s)}. \tag{6}$$

Similar to the case of continuous probability distribution, $\mathcal{D}_{KL}(P \parallel Q) \geq 0$, and $\mathcal{D}_{KL}(P \parallel Q) \neq \mathcal{D}_{KL}(Q \parallel P)$.

C. SOCIAL LSTM

In this study, we use Social LSTM as a machine learning model for trajectory prediction. Social LSTM predicts the trajectories of the target pedestrians with high accuracy even in a crowded environment with several pedestrians. This matches our target of a public facility with multiple pedestrians. Note that the method can be replaced by other models (e.g., Social GAN) as long as they can handle multiple pedestrians and use LiDAR-measured data (pedestrian location coordinate data) as input data.

The method is based on LSTM and incorporates the interaction effect among persons (e.g., moving behavior to avoid a mutual collision) into the prediction model by sharing the states of the hidden layers of all pedestrians in the target area. We explain the outline of the method according to the literature [15] in the following sections.

1) PROBLEM FORMULATION

The target problem is to obtain a model representation that outputs the future trajectory sequence ($t = T_{obs+1}$ to T_{pred}) by inputting the past observed trajectory sequence ($t = 1$ to T_{obs}). These sequences are represented as a series of position coordinates (x_t^i, y_t^i) of pedestrian i at time t .

2) SOCIAL POOLING AND HIDDEN STATES

The hidden layer sharing is realized by a concept called social pooling. Each pedestrian has its own LSTM model,

and the information sharing among the LSTMs is achieved by defining a social hidden-state tensor for each target area in the social pooling. Specifically, we denote the hidden-state vector of LSTM by h_t^i , then a hidden-state tensor for the target area H_t^i is defined as

$$H_t^i(m, n, :) = \sum_{j \in \mathcal{N}_i} l_{mn}[x_t^j - x_t^i, y_t^j - y_t^i] h_{t-1}^j. \tag{7}$$

If the grid size of the target area is $N \times N$, and the dimension of the hidden layer is $D \times N \times N$, the dimension of the tensor is $D \times N \times N$. Here, $l_{mn}[x, y]$ represents the xy coordinates in cell (m, n) , and \mathcal{N}_i represents the spatial information of pedestrians in the target area. Then, the position coordinate vector and the hidden-state tensor are embedded into the following vectors:

$$e_t^i = \phi(x_t^i, y_t^i; W_e) \tag{8}$$

$$a_t^i = \phi(H_t^i; W_a). \tag{9}$$

Using them, the state at time t is obtained from

$$h_t^i = \text{LSTM}(h_{t-1}^i, e_t^i, a_t^i; W_l) \tag{10}$$

where $\phi(\cdot)$ is the ReLU function, and W_e, W_a, W_l are the respective weights.

3) POSITION ESTIMATION

Using the state of the hidden layer at time t calculated above, we estimate the position coordinates $(\hat{x}, \hat{y})_{t+1}^i$ for the following cycle. We assume a two-dimensional Gaussian distribution with parameters $\mu_{t+1}^i = (\mu_x, \mu_y)_{t+1}^i$ as the mean value, $\sigma_{t+1}^i = (\sigma_x, \sigma_y)_{t+1}^i$ as the standard deviation, and ρ_{t+1}^i as the correlation coefficient. After that, we obtain $(\hat{x}_t^i, \hat{y}_t^i)$ as follows:

$$(\hat{x}, \hat{y})_t^i \sim \mathcal{N}(\mu_t^i, \sigma_t^i, \rho_t^i). \tag{11}$$

The parameters are estimated by the linear layer with a weight matrix W_p . The LSTM model parameters are trained by minimizing the negative log-likelihood (loss function) shown below for all pedestrians using the training dataset:

$$(\mu_t^i, \sigma_t^i, \rho_t^i) = W_p h_{t-1}^i \tag{12}$$

$$L^i(W_e, W_l, W_p) = - \sum_{t=T_{obs}+1}^{T_{pred}} \log(p(x_t^i, y_t^i) | (\mu_t^i, \sigma_t^i, \rho_t^i)) \tag{13}$$

where L^i is the loss function for pedestrian i .

4) INFERENCE FOR TRAJECTORY PREDICTION

Using the results obtained above, we predict the position coordinates $(\hat{x}, \hat{y})_t^i$ of the pedestrian from $t = T_{obs}+1$ to T_{pred} . By applying (10), the position coordinates of the next time are estimated sequentially. Finally, the prediction result as a series of data is obtained.

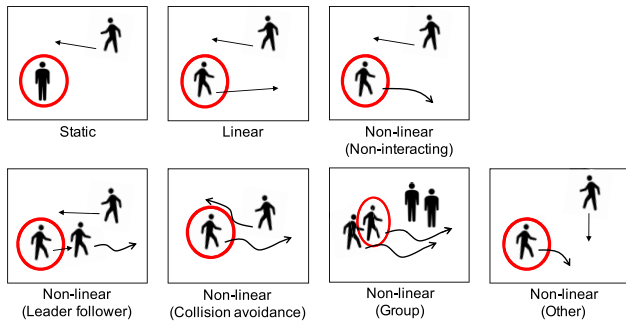


FIGURE 3. Classification of human walking trajectories [7]. The trajectories are classified into three major patterns: Static, Linear, and Non-linear. The Non-linear pattern is divided into further classifications to include more details.

IV. PROPOSED MODEL

The training data used to generate the pre-trained model is constructed by extracting a set of data similar to the target data from a large dataset:

- Extract multiple datasets that are candidates for the training data.
- Select the appropriate source dataset by representing the feature spaces of the source and target data by a function and comparing the similarities between the function values.
- Construct the training data from the selected data set and obtain the pre-trained model.

In the following, we describe the details of each procedure. For the sake of explanation, we will explain the method in the following order: how to represent the feature space, how to compare similarities, how to extract data sets, and how to construct training data.

A. FEATURE SPACE REPRESENTATION

This study aims to the pedestrian trajectory prediction. Various factors can characterize the walking trajectory; however, we consider a feature space with the velocity and angular velocity norms of walking as the feature values, referring to the recent research on classifying walking trajectories [7]. For simplicity, in the remaining parts, we will refer to the velocity and angular velocity norms as velocity and angular velocity, respectively. The image of the walking trajectory classification is shown in Fig. 3. Here, the human walking trajectory is classified into three major patterns: static, linear, and non-linear. Non-linear is further classified in detail. The specific major classification patterns are as follows:

- Static: standing still.
- Linear: moving according to the linear model (i.e., Kalman filter).
- Non-linear: moving in a way that does not follow the linear model.

The specific classification of non-linear patterns is subdivided as follows:

- Non-interacting: not interacting with other persons.

- Leader follower: following another person.
- Collision avoidance: moving to avoid a collision.
- Group: moving in groups.
- Other: none of the above.

The literature [7] shows that the proportion differences of each trajectory pattern in the dataset affect the accuracy of the trajectory prediction model trained by the data. In other words, the mixing degree of these patterns can characterize the trend of the person moving in the target area. In this study, we hypothesize that the above moving tendency can be expressed by two parameters, velocity and angular velocity, because walking trajectories are expressed as a series of temporal changes in position coordinates. In reality, the frequency distributions of velocity and angular velocity will differ depending on the trajectory classification regarding the seven patterns defined above. For example, in an area where most trajectories are Static, both velocity and angular velocity will be very small. In the Linear classification, the variances, as a statistical value, will be small. Moreover, when most trajectories are in Non-linear, the variance of both velocity and angular velocity is expected to be higher. Under the hypothesis that velocity and angular velocity can be used as the hidden variables for the classification (that is, the proportion of each trajectory pattern can be inferred to some extent by velocity and angular velocity), we adopt the feature space considering these statistical variables.

The feature space is represented with the probability density function or the probability mass function. We propose two methods: the parametric method (**M1**) that represents the space as a two-dimensional Gaussian distribution, and the non-parametric method (**M2**) that discretizes the sample values and obtains a distribution directly. The dataset is assumed to comprise the following data as one record, and the dataset is arranged in ascending order with respect to timestamp and person ID. In general, multiple person IDs exist with the same timestamp value.

- Record ID
- Timestamp
- Person ID
- Velocity norm
- Angular velocity norm

Although various datasets do not directly contain the velocity and angular velocity norms, these values can be calculated using the position coordinates and the measured frequency.

1) PARAMETRIC METHOD (M1)

The method using a two-dimensional Gaussian distribution to represent the feature space is called **M1** (the parametric method). This is one of the most straightforward representations, expressed only considering the mean and variance of the sample values. Here, we find the means μ_v , μ_ω and the variances σ_v^2 , σ_ω^2 for the velocity v_i and angular velocity ω_i where i is the record ID as the sample values without

distinction of timestamps and person IDs as follows:

$$\mu_v = \frac{1}{N} \sum_{i=1}^N v_i, \quad \mu_\omega = \frac{1}{N} \sum_{i=1}^N \omega_i \quad (14)$$

$$\sigma_v^2 = \frac{1}{N} \sum_{i=1}^N (v_i - \mu_v)^2, \quad \sigma_\omega^2 = \frac{1}{N} \sum_{i=1}^N (\omega_i - \mu_\omega)^2 \quad (15)$$

where the number of samples is N . From those, we obtain the mean vector μ and the covariance matrix Σ of the two-dimensional Gaussian distribution as

$$\mu = \begin{pmatrix} \mu_v \\ \mu_\omega \end{pmatrix}, \quad \Sigma = \begin{pmatrix} \sigma_v^2 & 0 \\ 0 & \sigma_\omega^2 \end{pmatrix}. \quad (16)$$

We assume $\sigma_{v\omega} = \sigma_{\omega v} = 0$ for simplicity. By substituting (16) into (1), the probability density function for the target dataset is determined.

2) NON-PARAMETRIC METHOD (M2)

The method that directly obtains the probability distribution by discretizing the sample values is called **M2** (the non-parametric method). Datasets extracted from a large dataset often have unknown distribution types. Therefore, we also adopt a non-parametric type method as one of the candidates to represent the dataset. Here, we first discretize the feature space, which is divided into grids of size $s \times t$ for the velocity axis (s) and angular velocity axis (t), respectively. Then, the probability for each grid is determined by dividing the number of samples in each grid by the total number of samples. Let n_k be the number of samples in grid k and N be the total number of samples; thus, the probability mass function $p(k)$ is determined as $p(k) = n_k/N$.

B. SIMILARITY COMPARISON

The goal here is to construct the training data by extracting a dataset having similar walking tendencies to those in the target environment from a large dataset. Numerous datasets can be extracted from a large dataset. Therefore, we select the dataset having the highest similarity between the probability distributions of the extracted dataset and the target environment. Moreover, the space features are represented using the probability distributions presented in the previous section to calculate the similarity.

Furthermore, we evaluate the similarities using KL divergence. This is a popular metric for the similarity of probability distributions; the smaller the value, the more similar their probability distributions between distributions. We use the metrics from (4) for **M1** and (6) for **M2**.

C. DATASET EXTRACTION

The candidate datasets for the training data are automatically extracted from a large dataset. The extraction method of the source dataset is shown in Fig. 4, which can be summarized as follows: (i) generate an exhaustive set of candidate datasets with a sliding window; (ii) extract the walking trajectories contained in each window defined with the same shape as

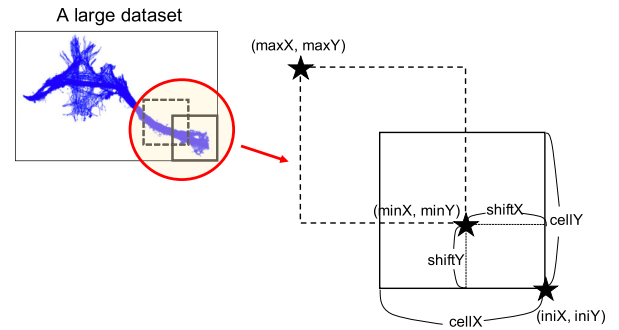


FIGURE 4. Method for extracting candidates of source datasets with a sliding window. The window is shifted by a unit length in the x and y directions and also shifted in the t -axis every unit time to extract multiple candidate datasets.

the target environment; (iii) obtain the velocity and angular velocity values from the trajectories; and (iv) determine the two-dimensional Gaussian distribution or the frequency distribution. The window is shifted by a unit length in the x and y directions and also shifted in the t -axis every unit time to extract multiple candidate datasets.

V. PREDICTION ALGORITHM

This section describes the prediction model of future pedestrian trajectories using the data constructed in the previous section. The model is divided into the following two phases: a training phase (i.e., offline process) and a prediction phase (i.e., online process). The model trained by a source dataset is used to predict trajectories in the target environment. The process can be summarized as follows:

- Training phase
 - 1) Generate training data from a large dataset.
 - 2) Generate a prediction model using the training data.
- Prediction phase
 - 1) Measure the trajectory data of pedestrians.
 - 2) Generate input data from the measurement data.
 - 3) Predict the trajectory using the input data.

The flowchart of the trajectory prediction process is outlined in Fig. 5.

A. TRAINING PHASE

1) GENERATING TRAINING DATA

Subsequently, the training data for machine learning algorithms is generated by extracting a portion of data from a large open dataset. After that, an appropriate data transformation is applied. We assume that the following items are contained in each record in the target dataset:

- Timestamp
- Person ID
- Position coordinate of the pedestrian at the measurement time (x, y)

First, datasets are extracted from a large dataset according to the method described in the previous section. Then, the

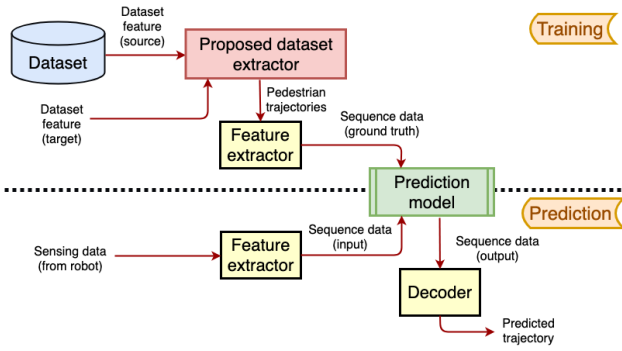


FIGURE 5. Outline of the trajectory prediction process. We train a prediction model using training data generated in the previous section in advance. The future trajectory is predicted using the model and sensing results acquired by a human-following mobile robot.

target dataset \mathcal{S}^* is determined as

$$\mathcal{S}^* = \underset{\mathcal{S}_i}{\operatorname{argmin}} \mathcal{D}_{\text{KL}}(p_i^s \parallel p^t) \quad (17)$$

where the probability distribution representing the feature of the dataset \mathcal{S}_i is p_i^s , and the distribution of the target dataset is p^t .

Next, we extract the pedestrian trajectory data for each person ID from \mathcal{S}^* . The trajectory data are generated as sequences by combining the input data sequence $\{\mathbf{x}_j^i, \mathbf{x}_{j+1}^i, \dots, \mathbf{x}_{j+k-1}^i\}$ consisting of k frames with the output data sequence $\{\mathbf{x}_{j+k}^i, \mathbf{x}_{j+k+1}^i, \dots, \mathbf{x}_{j+k+l-1}^i\}$ consisting of l frames. Here, $\mathbf{x}_j^i = (x_j^i, y_j^i)$ is the initial position coordinate where the initial frame of person i is j . Finally, we vertically concatenate the joint data sequences in the direction of the person IDs and output them as the training data. One record of the training data is as follows:

- Frame ID
- Person ID
- Position coordinates of the pedestrian at the measurement time

Fig. 6 shows an image regarding how the training data is generated. For example, among the dataset, we extract the position coordinate data from the records whose person ID is 1 and generate a joint sequence as temporary data. Similarly, we generate temporary data for person ID 2, 3, ... Finally, the training data is constructed by concatenating these sequences vertically.

2) GENERATING THE PREDICTION MODEL

We input the training data constructed in the previous section into the machine learning algorithm and generate a pre-trained prediction model.

B. PREDICTION PHASE

1) MEASURING DATA

For tracking, we use the same method as described in [23] (the ankle of the pedestrian is tracked in [23]; nevertheless, the bodies are tracked in this study). The position of each

Algorithm 1 Algorithm of the Particle Filter [24]

Input: $\mathcal{X}_{t-1}, \mathbf{a}_t, \mathbf{z}_t$

Output: \mathcal{X}_t

- 1: $\bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$
- 2: **for** $m = 1, \dots, M$ **do**
- 3: sample $\mathbf{x}_t^{[m]} \sim p(\mathbf{x}|\mathbf{a}_t, \mathbf{x}_{t-1}^{[m]})$
- 4: $w_t^{[m]} = p(\mathbf{z}_t|\mathbf{x}_t^{[m]})$
- 5: $\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle \mathbf{x}_t^{[m]}, w_t^{[m]} \rangle$
- 6: **end for**
- 7: **for** $m = 1, \dots, M$ **do**
- 8: draw i with probability $\propto w_t^{[i]}$
- 9: add $\mathbf{x}_t^{[i]}$
- 10: **end for**
- 11: **return** \mathcal{X}_t

tracking target is expressed as a probability distribution, and the state of the tracking object is recursively estimated using a probability density function from a transition model (the movement of the pedestrian) corresponding to the tracking target and an observation model based on the sensing data (the distance measured by the LiDAR). We use a particle filter for the probability distribution of a pedestrian position. In the particle filter, the probability distribution is defined as a set of samples (particles), and a posterior probability is generated by sampling the particles according to the weights. An algorithm of the particle filter is shown in Algorithm 1, where $\mathbf{x}_t = (x_t, y_t, \theta_t, v_t)^T$ denotes position, posture, and velocity of a person in two-dimensional plane, respectively. Moreover, the state \mathbf{a}_t is the motion, and \mathbf{z}_t is the observation. \mathcal{X}_t is a set of particles at time t and expressed as follows:

$$\mathcal{X}_t := \mathbf{x}_t^{[1]}, \mathbf{x}_t^{[2]}, \dots, \mathbf{x}_t^{[M]}. \quad (18)$$

Each particle $\mathbf{x}_t^{[m]}$ is a concrete sample of the state at time t , and the probability distribution is expressed by a set of the samples \mathcal{X}_t . In this algorithm, a temporal set of particles $\bar{\mathcal{X}}_t$ is generated by operating \mathbf{a}_t and \mathbf{z}_t to a set of particles in the previous step \mathcal{X}_{t-1} , and then, a set of particles in the next step \mathcal{X}_t is generated by conducting weighted sampling (re-sampling) of the temporal set.

The positions of the pedestrians are calculated from the distances and angles measured by the LiDAR by converting the polar coordinate to the orthogonal coordinate. From the result, $(\hat{x}_m^i, \hat{y}_m^i)$, which is the position data of each pedestrian in the measuring range per frame, is obtained. At that time, these values are represented in the local coordinate system of the robot.

Here, by using the particle filter (Bayesian filter) to track the position of pedestrians as a probability distribution, we obtain input data (a series of data) in which random noise, outliers, and missing values during observation are pre-processed. Therefore, there is no need to consider the potential influences of random noise and other factors on the trajectory prediction performance.

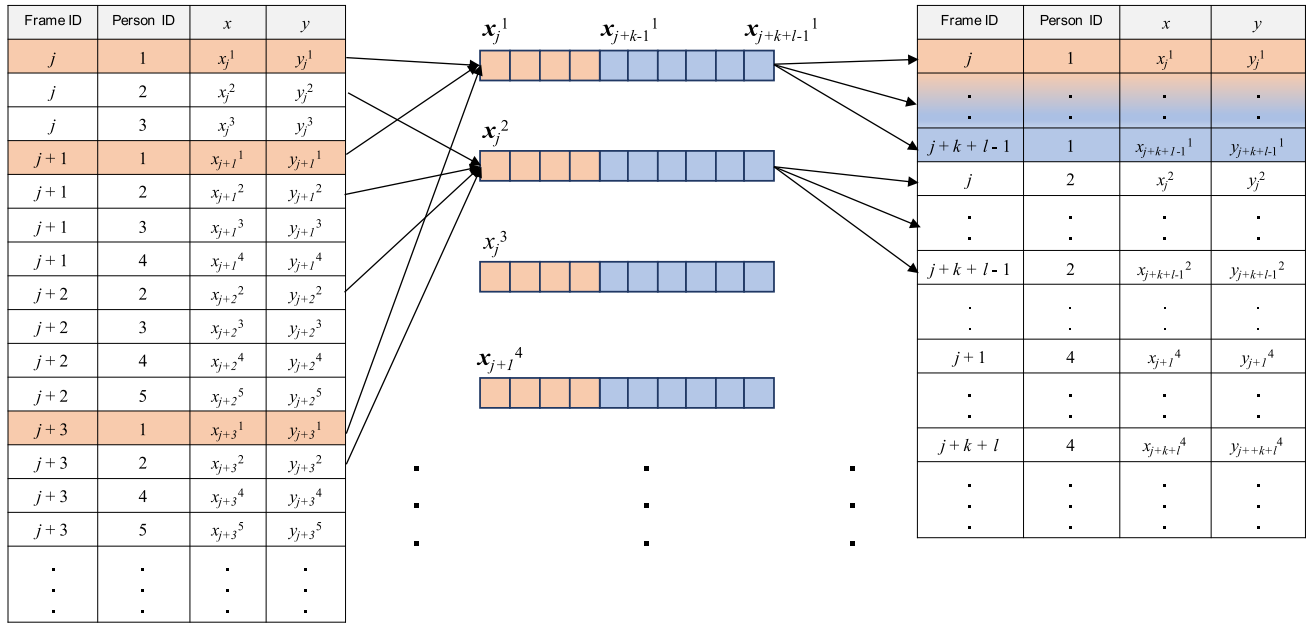


FIGURE 6. Development of the training data. Among the dataset, the position coordinates data from the records whose person ID is 1 are extracted as temporary data. Similarly, temporary data for person ID 2, 3, ... are also generated. Finally, the training data is constructed by concatenating these sequences vertically.

2) GENERATING INPUT DATA

From the obtained measurement data, we generate input data for the prediction model. As the pedestrian position data sequence in the world coordinate system is input to the prediction model, we prepare data matching the model. Specifically, we collect $(\hat{x}_m^i, \hat{y}_m^i)$ for every k frame for the number of the pedestrians observed (i.e., the maximum value of i) and transform the local coordinate system into the world coordinate system.

Here, the coordinate of pedestrian i at frame m in the world coordinate system is calculated using (X_m^R, Y_m^R) and φ_m , which are the coordinates (the center coordinates of the axle) and posture (the direction that the robot is facing) of the following robot, respectively, in the world coordinate system as follows:

$$\begin{pmatrix} x_m^i \\ y_m^i \end{pmatrix} = \begin{pmatrix} \cos \varphi_m & -\sin \varphi_m \\ \sin \varphi_m & \cos \varphi_m \end{pmatrix} \begin{pmatrix} \hat{x}_m^i \\ \hat{y}_m^i \end{pmatrix} + \begin{pmatrix} X_m^R \\ Y_m^R \end{pmatrix}. \quad (19)$$

Moreover, we assume that the coordinate and posture of the robot can be calculated using other schemes (such as a localization method using an environmental map).

3) PREDICTING THE TRAJECTORY

Using the method described in the previous section, we construct $\{x_j^i, x_{j+1}^i, \dots, x_{j+k-1}^i\}$ for the observed pedestrians and input the data into the pre-trained prediction model. As a result, we can obtain $\{x_{j+k}^i, x_{j+k+1}^i, \dots, x_{j+k+l-1}^i\}$ as the future position coordinate series.

VI. EXPERIMENTS

To verify the effectiveness of the proposed training data construction method, we generated the pre-trained prediction

model using training data extracted from a large dataset and conducted experiments to evaluate the prediction accuracy using test data. Specifically, we extract the source datasets from a large dataset and compare the similarities of the feature space between them using **M1** and **M2** described in Section IV. Furthermore, by generating pre-trained models using several datasets, we compare how the similarity differences between datasets affect the trajectory prediction accuracy. The Social LSTM is used to generate the prediction model. Moreover, for the parameter values for generating the training data, we set the number of input frames k to eight and that of output frames l to 12.

A. EXPERIMENTAL SETTINGS

1) PREPARATION OF TEST DATA

In the experiments, we evaluate the trajectory prediction accuracy in the target environment using pre-trained prediction models. Therefore, the datasets collected in the target environment are the test data for the evaluation. In this regard, we prepared two types of datasets with different features as test data. The first set corresponded to the dataset generated from data measured in a public space, the Fukagawa Edo Museum,² on August 9 and 10, 2019 (called **T1**). The second set was the ETH Dataset [9], which is often used in trajectory prediction research (called **T2**). We converted it to fit the experimental environment.

A floor map of the museum and the collected data samples are shown in Fig. 7. The dataset (**T1**) contains several trajectories where people walk slowly around while looking

²Fukagawa Edo Museum: <https://www.kcf.or.jp/fukagawa/>

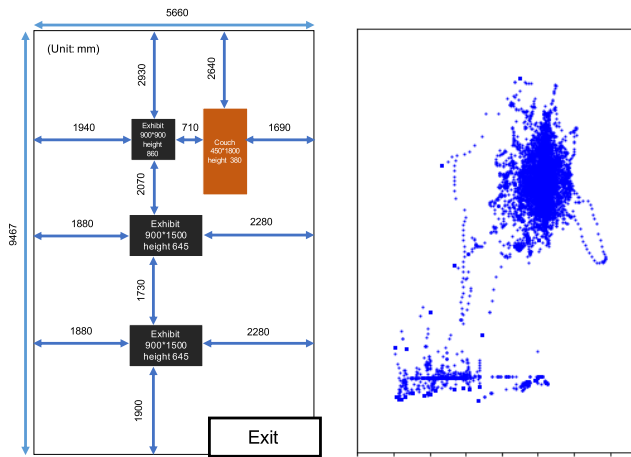


FIGURE 7. Floor map and photo of the Fukagawa Edo Museum and collected data samples of the trajectories. The dataset contains several trajectories where people walk slowly around while looking at the exhibitions.

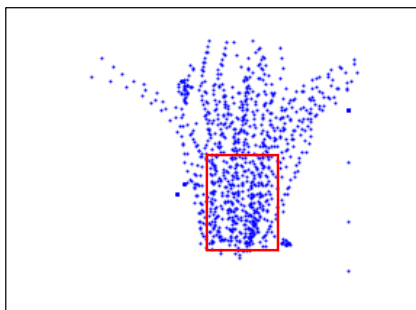


FIGURE 8. Data samples of the trajectories contained in the ETH dataset. The dataset comprises multiple pedestrian scenes in the city obtained at a bird's-eye view. It also contains data on the trajectory of human movements in a relatively simple environment. For the experiment, the trajectory data in the red frame is considered to fit the experimental environment.

at the exhibitions. The total number of trajectories is seven. One trajectory is obtained for each track. Moreover, multiple series of data (sequences) divided into 20 frames are obtained for each trajectory.

For the ETH dataset (**T2**), the data samples of the trajectories in the set are shown in Fig. 8. This dataset consists of multiple pedestrian scenes in the city obtained at a bird's-eye view and contains data regarding the trajectory of human movements in a relatively simple environment. In this experiment, to compare the prediction results with those of **T1**, we use the data from a particular area. Specifically, we use the data in the red frame shown in Fig. 8. In practice, the trajectory of each pedestrian is extracted from the video data. Moreover, the data transformed into a series of position coordinates are used. The total number of trajectories is 268.

2) PREPARATION OF TRAINING DATA

The candidate datasets for generating the pre-trained model are extracted from the large dataset, the ATC pedestrian

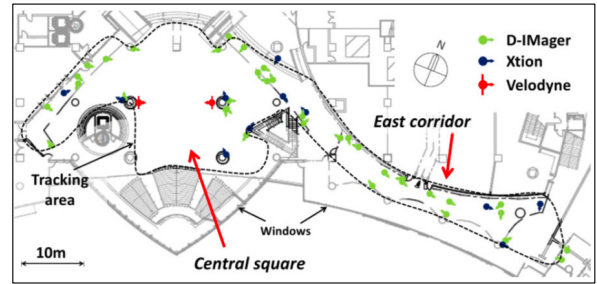


FIGURE 9. Floor map of the shopping mall (from [25] ©2013 IEEE). The candidate data sets for generating the pre-trained model are extracted from the dataset collected in the mall. Here, human positions were continuously measured at 10 ~ 40 Hz with multiple 3D range image sensors.

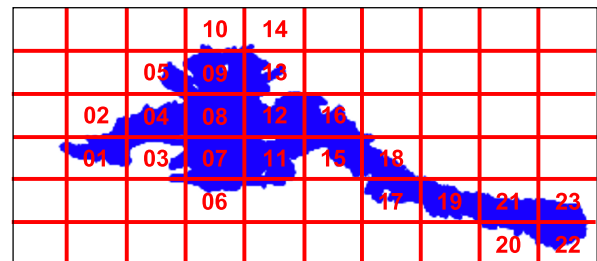


FIGURE 10. Mapping between training data candidates and area numbers. Areas that do not contain any trajectories are excluded from the numbering.

dataset [25]. This dataset comprises moving trajectories of 3,758,346 pedestrians (the average is 40,851 per day, including duplication) in a commercial facility, the ATC Shopping Mall in Osaka, Japan. The data considers 92 days corresponding from one year (from 9:40 to 20:20 every Wednesday and Sunday from October 2012 to November 2013). The set includes some metadata such as person IDs. Human positions were continuously measured at 10 ~ 40 Hz (depending on human density) with multiple 3D range image sensors. The floor map of the shopping mall is shown in Fig. 9. From the ATC dataset, the experiment considered 10,000 trajectories randomly selected from the data collected between 9:20 to 10:20 a.m. on October 24, 2012. The newly extracted dataset is hereafter referred to as the large dataset.

We generated the candidate datasets for training automatically and exhaustively according to the procedure shown in Section IV. In this experiment, we set the size of one grid to 5,700 × 9,500 mm and wrote the trajectories contained in each area in a separate file for each area. These areas were identified by numbers. The mapping between training data candidates and area numbers in this experiment is shown in Fig. 10. Note that areas that do not contain any trajectories are excluded from the numbering.

3) CALCULATION OF KL DIVERGENCE

To select the training data used in the experiment, we first determined the form of the function representing the space

TABLE 3. Trajectory data features in the datasets and KL divergence for each method and test data.

ID	Feature	M1 (T1)	M2-s (T1)	M2-m (T1)	M1 (T2)	M2-s (T2)	M2-m (T2)
01	S-, L+	9.77	6.73	5.66	10.69	7.29	6.44
02	S-, L+	8.18	5.94	3.98	9.80	6.67	4.79
03	S-, L+	8.65	6.16	5.68	10.49	7.38	6.13
04	S-, L+	9.03	6.20	5.86	4.17	3.00	2.32
05	S-, L+	7.36	5.82	4.13	5.32	3.22	2.91
06	S+, L-, N+	7.36	5.80	4.70	8.49	6.33	4.62
07	S+, L-, N+	7.31	5.40	3.49	5.24	3.13	2.99
08	S+, L-, N+	5.63	5.23	3.22	3.33	2.72	2.02
09	S-, L+, N-	5.43	5.57	3.99	2.52	1.01	1.13
10	S-, L+, N-	6.96	6.59	5.14	3.43	2.76	2.10
11	S+, L-, N+	9.46	4.86	3.12	5.76	3.34	3.45
12	S+, L-, N+	4.89	3.30	3.79	10.99	7.34	5.46
13	S-, L+, N-	4.37	5.50	4.11	8.76	6.38	4.35
14	S-, L+, N-	6.04	5.70	5.17	8.22	6.11	4.77
15	L+	6.96	5.01	3.99	5.14	3.45	2.33
16	L+	5.98	6.47	6.12	6.21	4.74	3.99
17	L+	9.07	7.37	6.73	9.98	6.91	5.19
18	L+	9.77	7.12	6.11	13.01	8.22	7.18
19	L+	10.13	7.65	10.50	14.42	9.34	10.63
20	L+	9.81	6.85	5.86	9.33	6.43	4.86
21	L+	11.10	7.64	5.99	10.40	7.28	5.93
22	L+	9.16	6.50	4.69	10.12	7.20	5.42
23	L+	9.31	6.50	4.98	13.04	8.54	9.30

S = Static, L = Linear, and N = Non-linear; + implies that the category contains a large number of trajectories, and - implies that the category contains a small number of trajectories; the figures correspond to the KL divergence.

features using **M1** and **M2** for each area. After that, we calculated the KL divergences for each test data (**T1** and **T2**). For **M2**, two types of methods were prepared to evaluate the effect of the grid size: **M2-s** with $s = 15$, $t = 6$ and **M2-m** with $s = 200$, $t = 1000$. The results are listed in Table 3. The features of the trajectory data in the datasets are also included in the table. Moreover, we classified the trajectory data contained in each dataset into three types: Static, Linear, and Non-linear, as the same in the literature [7]. The percentage of each type by sight was also evaluated.

The KL divergence values of **T1** and **T2** for each area is shown in Fig. 11. As for the KL divergence, even though the order is reversed in some parts, the trend of the fluctuation is almost the same for all methods (i.e., **M1**, **M2-s**, and **M2-m**) regardless of **T1** and **T2**. In contrast, the areas with the minimum KL divergence are different depending on the test data. This result shows that the proposed method can represent the differences in the target environments. Regarding the result, we selected areas 9, 11, 12, and 19 for the experiment, generated the prediction models using these data as training datasets, and compared the prediction accuracy with **T1** and **T2**.

B. EVALUATION METRICS

As evaluation metrics, we used the average displacement error (ADE), which is the average value of the frame-by-frame error (Euclidean distance) between the predicted trajectory and the ground truth value. Furthermore, we considered the final displacement error (FDE), which is the difference in the final destination between the predicted

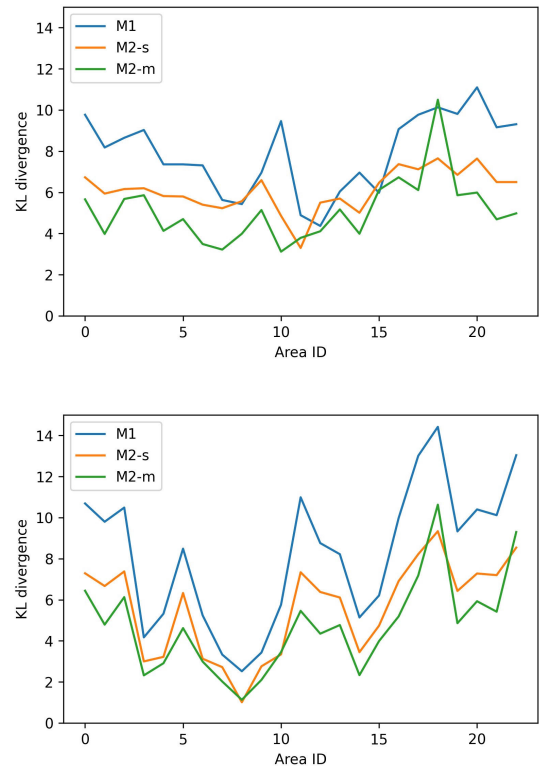


FIGURE 11. Values of KL divergence of **T1** (the figure on the top) and **T2** (the figure on the bottom) for each area. Although the order is reversed in some parts, the fluctuation trend is almost the same for all methods.

trajectory and the ground truth value. These values are calculated as

$$ADE = \frac{1}{M} \sum_{m=1}^M \|\hat{x}_m - x_m\|_2 \tag{20}$$

$$FDE = \|\hat{x}_M - x_M\|_2 \tag{21}$$

where x_m is the predicted coordinate of the m frame, \hat{x}_m is the ground truth value of the m frame, and M is the predicted sequence length.

C. EXPERIMENTAL RESULTS

The experimental results are shown in Table 4. The relationship between KL divergence and the performance evaluation values (ADE, FDE) shows that the evaluation value is the smallest (i.e., the highest performance) in the area with the smallest value of KL divergence, and the evaluation value is the largest (i.e., the lowest performance) in the area with the largest value of KL divergence. For example, for **T1**, the KL divergence in area 12 is the smallest, and the area values of ADE and FDE are the minimum in all areas.

These results show that the proposed method can represent feature spaces and compare the similarities among them. Therefore, the approach can construct the training data in the source environment and generate an appropriate pre-trained prediction model by using the constructed data.

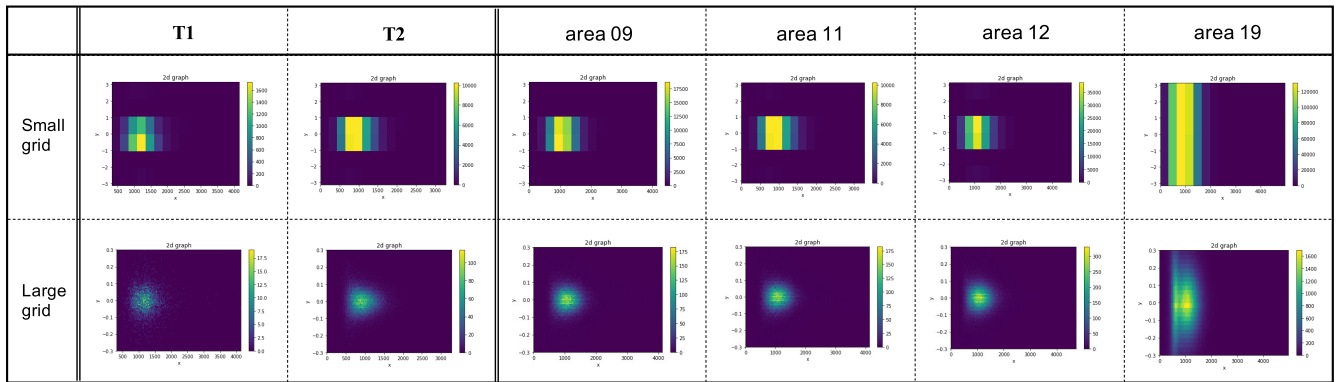


FIGURE 12. Visualization results of discretizing the sample values contained in each dataset. The results are plotted in the feature spaces for T1 and T2. We selected areas 09, 11, 12, and 19. The vertical and horizontal axes correspond to the velocity and angular velocity, respectively.

TABLE 4. Experimental results. Relationship between KL divergence and performance evaluation values (ADE, FDE).

ID	KL (T1)	ADE (T1)	FDE (T1)	KL (T2)	ADE (T2)	FDE (T2)
09	5.57	2.75	3.41	1.01	0.92	1.31
11	4.86	2.02	3.29	3.34	1.96	2.32
12	3.30	1.83	3.17	7.34	4.65	5.11
19	7.65	5.34	6.09	9.34	6.12	6.75

The values of M2-s are used as the KL divergences in this table.

D. DISCUSSION

1) VISUALIZATION OF FEATURE SPACE

We selected several datasets with different features from the candidates and visualized them by plotting the sample values in the feature spaces. This allows us to recognize differences between datasets visually and subjectively. We chose areas 11, 12, and 19 for T1 and areas 09 and 19 for T2, discretized the sample values contained in each dataset and plotted them in the feature space (with velocity and angular velocity). We created two types of graphs in both cases, one with a small discretization grid width and the other with a large discretization grid width. Fig. 12 shows the result.

The results show that the T1 and areas 11 and 12 present similar shapes. In contrast, the shape from area 19 differs substantially. The same results are obtained for T2; namely, area 09 is similar in shape, while area 19 is substantially different. As the KL divergence properly evaluates the similarity of this shape, the value is considered an effective measure to evaluate the similarity between datasets when the feature space is properly represented by the proposed method. Moreover, comparing the shapes of T1 and T2, T1 has higher variance of angular velocity than T2. This may be the reason why the areas where the KL divergence is minimized are different in the two test datasets.

2) VISUALIZATION OF PREDICTED TRAJECTORY

In addition, we visualized representative prediction trajectories with different types of pre-trained models and different types of test datasets. This allows us to recognize visually and subjectively the effect between the similarity differences

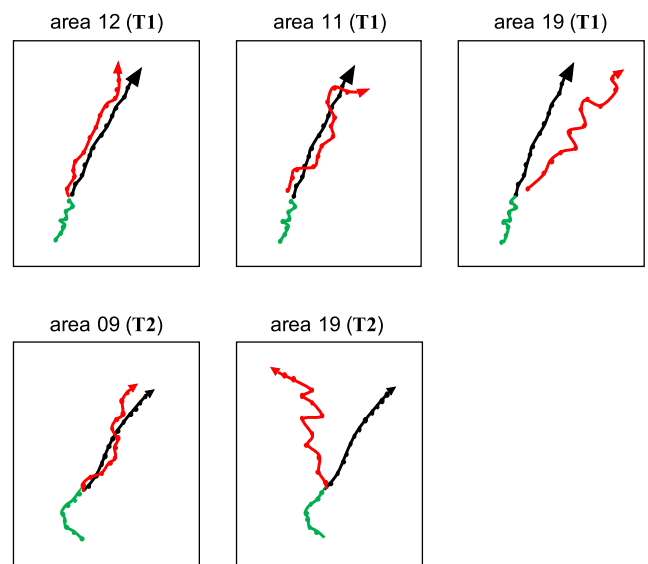


FIGURE 13. Results of visualizing the predicted trajectory for sequence data. The observed data input to the predictor (8 frames of data) is depicted in green lines, the predicted data output from the predictor (12 frames of data) are depicted in red lines, and the ground truth values contained in the test data (12 frames of ground truth) is depicted in black lines. The higher the degree of proximity between the red and black lines, the higher the prediction accuracy.

of the datasets regarding the trajectory prediction results. Similar to the previous section, we selected areas 11, 12, and 19 for T1, and areas 09 and 19 for T2, and visualized the prediction series outputted from the pre-trained models. The results are shown in Fig. 13. The observed data input to the predictor (8 frames of data) is depicted in green lines, the predicted data output from the predictor (12 frames of data) are depicted in red lines, and the ground truth values contained in the test data (12 frames of ground truth) are depicted in black lines. The higher the degree of proximity between the red and black lines, the higher the prediction accuracy.

Fig. 13 shows that we can obtain highly accurate prediction results by generating a prediction model with a dataset that

has a small KL divergence, i.e., the high similarity in the feature space. In both testing datasets, the predicted trajectory and the ground truth are significant differences in area 19, which has a large KL divergence. Thus, the dataset is inappropriate to generate the prediction model. Moreover, the experimental result of **T1** shows that the prediction accuracy can be improved by selecting an area with a smaller KL divergence.

Here, the predicted trajectories are not smoother than the ground truth. This is due to the fact that the method can predict linear and nonlinear trajectories. Although the trajectories are more jagged, the trend of the obtained sequences is appropriate. On the other hand, it is difficult for existing Bayesian filters to predict such trajectories when the sequence length is long. In practice, when implementing the proposed method in a real robot, we generate probability distributions from the obtained sequence and use them as a cost map for navigation. Therefore, the jagged trajectory itself is not a major issue.

3) THEORETICAL ANALYSIS

Here, we provide a theoretical analysis of the computational complexity of the proposed method. Let nc be the number of candidate datasets extracted from a large dataset, and ng be the number of discretized feature spaces in **M2**.

*Lemma 1: The computation time of **M1** to generate the training data is $O(nc)$.*

Proof: To generate training data, **M1** extracts the candidate datasets from a large dataset and calculates KL divergence values of them. It requires $O(nc)$ time to compare the similarity between the feature space of each candidate and that of the target environment. For KL divergence calculation, it requires $O(1)$ time, because the feature spaces are represented as two-dimensional Gaussian distribution as follows:

$$\mathcal{D}_{\text{KL}} = \frac{1}{2} \left[\sum_{i=1}^2 \left(\frac{(\mu_i - \hat{\mu}_i)^2}{\hat{\sigma}_i^2} + \frac{\sigma_i^2}{\hat{\sigma}_i^2} - \ln \frac{\sigma_i^2}{\hat{\sigma}_i^2} \right) - 2 \right] \quad (22)$$

where (μ_1, μ_2) and $(\hat{\mu}_1, \hat{\mu}_2)$ are the mean vectors of the source and target data, respectively. σ_1, σ_2 and $\hat{\sigma}_1, \hat{\sigma}_2$ are the diagonal elements of the covariance matrix of the source and target data, respectively. Thus, the complexity is $O(nc)$. \square

*Lemma 2: The computation time of **M2** to generate the training data is $O(nc)$ if $nc > ng$ and $O(ng)$ otherwise.*

Proof: Similar to **M1**, it requires $O(nc)$ time to compare the similarity between the feature space of each candidate and that of the target environment. For KL divergence calculation, it requires $O(ng)$ time because the probability of each grid is determined by dividing the number of samples in each grid by the total number of samples. Thus, the complexity is $O(nc)$ if $nc > ng$ and $O(ng)$ otherwise. \square

VII. CONCLUSION

This paper described a novel method to generate an appropriate pre-trained model for transfer learning, which targets

pedestrian trajectory prediction methods using machine learning. The results of the study can be summarized as follows:

- We proposed generating an appropriate pre-trained prediction model by representing the feature spaces of the source and target datasets as a multidimensional Gaussian distribution and a frequency distribution to compare the similarity between the distributions with KL divergence.
- The experimental results showed that both methods adequately represent the data features. Moreover, an appropriate pre-trained model with high accuracy in trajectory prediction can be generated by comparing the similarity between datasets using these representations.

With the proposed method, the trajectory prediction method can be applied to robot control only using a few input data (simple measurement) and only deploying a robot to a target environment. In the future, we will extend the method so that the pre-trained model can be used even though prior input data do not exist, such as combining online learning with some functions on cloud environments [26], and the explainability techniques can be used to understand the internal behavior of the proposed approach [27]. We will also consider incorporating aspects such as the shape of the buildings, time of day, personal attributes into the feature values. Furthermore, it is necessary to consider features in space and time (i.e., features in the temporal direction).

REFERENCES

- [1] D. Helbing and P. Molnár, "Social force model for pedestrian dynamics," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 51, no. 5, p. 4282, 1995.
- [2] Y. Kato, Y. Nagano, and H. Yokoyama, "A pedestrian model in human-robot coexisting environment for mobile robot navigation," in *Proc. IEEE/SICE Int. Symp. Syst. Integr. (SII)*, Dec. 2017, pp. 992–997.
- [3] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Trans. Cogn. Commun. Netw.*, vol. 3, no. 4, pp. 563–575, Dec. 2017.
- [4] G. Aceto, D. Ciunzo, A. Montieri, and A. Pescapé, "Mobile encrypted traffic classification using deep learning: Experimental evaluation, lessons learned, and challenges," *IEEE Trans. Netw. Service Manag.*, vol. 16, no. 2, pp. 445–458, Feb. 2019.
- [5] Y. Huang, Y. Li, Z. Zhang, and R. W. Liu, "GPU-accelerated compression and visualization of large-scale vessel trajectories in maritime IoT industries," *IEEE Internet Things J.*, vol. 7, no. 11, pp. 10794–10812, Nov. 2020.
- [6] R. Akabane and Y. Kato, "Pedestrian trajectory prediction using pre-trained machine learning model for human-following mobile robot," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2020, pp. 3453–3458.
- [7] P. Kothari, S. Kreiss, and A. Alahi, "Human trajectory forecasting in crowds: A deep learning perspective," 2020, *arXiv:2007.03639*. [Online]. Available: <https://arxiv.org/abs/2007.03639>
- [8] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Statist.*, vol. 22, no. 1, pp. 79–86, 1951.
- [9] S. Pellegrini, A. Ess, K. Schindler, and L. van Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Sep. 2009, pp. 261–268.
- [10] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [11] A. Sadeghian, A. Alahi, and S. Savarese, "Tracking the untrackable: Learning to track multiple cues with long-term dependencies," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 300–311.

- [12] Y. Xiang, A. Alahi, and S. Savarese, "Learning to track: Online multi-object tracking by decision making," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4705–4713.
- [13] L. Chen, H. Ai, Z. Zhuang, and C. Shang, "Real-time multiple people tracking with deeply learned candidate selection and person re-identification," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jul. 2018, pp. 1–6.
- [14] N. Sakata, Y. Kinoshita, and Y. Kato, "Predicting a pedestrian trajectory using Seq2Seq for mobile robot navigation," in *Proc. 44th Annu. Conf. IEEE Ind. Electron. Soc. (IECON)*, Oct. 2018, pp. 4300–4305.
- [15] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human trajectory prediction in crowded spaces," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 961–971.
- [16] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social GAN: Socially acceptable trajectories with generative adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 2255–2264.
- [17] H. Shimodaira, "Improving predictive inference under covariate shift by weighting the log-likelihood function," *J. Statist. Planning Inference*, vol. 90, no. 2, pp. 227–244, 2000.
- [18] B. Zadrozny, "Learning and evaluating classifiers under sample selection bias," in *Proc. 21st Int. Conf. Mach. Learn. (ICML)*, 2004, pp. 903–910.
- [19] S. Thrun, "Is learning the n-th thing any easier than learning the first?" in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 8, 1995, pp. 640–646.
- [20] R. Raina, A. Y. Ng, and D. Koller, "Constructing informative priors using transfer learning," in *Proc. 23rd Int. Conf. Mach. Learn. (ICML)*, 2006, pp. 713–720.
- [21] C. B. Do and A. Y. Ng, "Transfer learning for text classification," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 18, 2006, pp. 299–306.
- [22] M. Montemerlo, S. Thrun, and W. Whittaker, "Conditional particle filters for simultaneous mobile robot localization and people-tracking," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2002, pp. 695–701.
- [23] S. Sakai, S. Kimura, D. Nomiya, T. Ikeda, N. Matsuhira, and Y. Kato, "Classification of age groups using walking data obtained from a laser range scanner," in *Proc. 42nd Annu. Conf. IEEE Ind. Electron. Soc. (IECON)*, Oct. 2016, pp. 5862–5867.
- [24] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, MA, USA: MIT Press, 2005.
- [25] D. Brscic, T. Kanda, T. Ikeda, and T. Miyashita, "Person tracking in large public spaces using 3-D range sensors," *IEEE Trans. Human-Mach. Syst.*, vol. 43, no. 6, pp. 522–534, Nov. 2013.
- [26] Y. Kato, T. Izui, Y. Tsuchiya, M. Narita, M. Ueki, Y. Murakawa, and K. Okabayashi, "RSI-cloud for integrating robot services with internet services," in *Proc. 37th Annu. Conf. IEEE Ind. Electron. Soc. (IECON)*, Nov. 2011, pp. 2164–2169.
- [27] G. Aceto, G. Bovenzi, D. Ciunzo, A. Montieri, V. Persico, and A. Pescape, "Characterization and prediction of mobile-app traffic using Markov modeling," *IEEE Trans. Netw. Service Manage.*, vol. 18, no. 1, pp. 907–925, Mar. 2021.



RINA AKABANE received the B.S. and M.S. degrees in mathematical science from Tokyo Woman's Christian University, Tokyo, Japan, in 2019 and 2021, respectively.

Her research interests include machine learning techniques and mathematical models for intelligent robotics.



YUKA KATO (Member, IEEE) received the B.S. degree in geophysics from The University of Tokyo, Tokyo, Japan, in 1989, and the M.E. and Ph.D. degrees in information systems from The University of Electro-Communications, Tokyo, in 1999 and 2002, respectively.

From 1989 to 1998, she was with Nippon Telegraph and Telephone Corporation and engaged in research on traffic control in ATM networks and interactive multimedia systems. She was a Research Associate with The University of Electro-Communications, from 2002 to 2006, an Associate Professor with the Advanced Institute of Industrial Technology, from 2006 to 2009, and a Professor with the Advanced Institute of Industrial Technology, from 2009 to 2014. Since 2014, she has been a Professor at Tokyo Woman's Christian University. Her research interests include information networks, intelligent robotics, and mathematical models for robotics.

...