Robotica (2022), **40**, pp. 544–569
doi:10.1017/S0263574721000679

CAMBRIDGE UNIVERSITY PRESS

RESEARCH ARTICLE

Learning-based simulation and modeling of unorganized chaining behavior using data generated from 3D human motion tracking

Abhinav Malviya* and Rahul Kala

Centre of Intelligent Robotics, Indian Institute of Information Technology, Allahabad, Prayagraj, India
*Corresponding author. Email: abhinavmcs0001@gmail.com

Received: 22 July 2020; **Revised:** 19 January 2021; **Accepted:** 4 May 2021; **First published online:** 16 June 2021

Keywords: Social robotics, Behavior analysis, Behavior modeling, Human detection, Tracking, Simulation

Abstract

The paper models the unorganized chaining behavior, where humans need to walk in a chain due to a constrained environment. Detection and tracking are done using a 3D LiDAR, which has the challenges of environmental noises, uncontrolled environment, and occlusions. The Kalman filter is used for tracking. The trajectories are analyzed and used to train a behavioral model. The modeling has applications in socialistic robot motion planning and simulations. Based on the results, we conclude that the trajectory prediction by our approach is more socialistic and has a lesser error when compared to the artificial potential field method.

List of symbols

Symbols	Description
Detection	
$P(t)$	Input point cloud at time t
$P'(t)$	Processed point cloud using a pass-through filter
$p(p_x, p_y, p_z)$	Single point in the input point cloud
$[X_{min}, X_{max}]$, $[Y_{min}, Y_{max}]$, $[Z_{min}, Z_{max}]$	Range in X, Y, and Z coordinates for a pass-through filter
$Moving\ (t)$	Points corresponding to moving people at time t
$d()$	Distance function
η	Distance threshold between consecutive frames, above which an entity is called moving
$Moving'(t)$	Moving points after outlier removal
$\delta\ (p)$	neighborhood function (k-nearest neighbors)
$D\ (p_i)$	Average distance of point p_i with neighbors
$\overline{D}$ and $\sigma\ (D)$	Average distance and standard deviation of all points in the point cloud with their neighbors
α	Maximum Z-score after which a point is called an outlier
C	Set of clusters (moving people)
fixed(p)	Point is a part of a cluster?
c	Candidate cluster
E	Distance threshold for inclusion of points in the Euclidean clustering
$z_j\ (t)$	Centroid of the jth cluster at time t

Symbols	Description
Tracking and Kalman filter	
$\kappa\ (i)$	Correspondence function
$z_{\kappa(i)}\ (t)$	Observation
$S_i\ (t)$	Updated estimate/state after observation
$p_i\ (t)$	Process covariance matrix after observation
$\hat{S}_i\ (t)$	Predicted state before observation
A and A'	State transition matrix and its transpose
$\hat{p}_i\ (t)$	Process covariance matrix before observation
$W\ (t)$	Gaussian motion noise
$K_i\ (t)$	Kalman gain
H and H'	Observation matrix and its transpose
$R\ (t)$	Process noise covariance
I	Identity matrix
T	Time-annotated trajectories of all persons
K	Set of all Kalman trackers for all detected persons
Ψ	Predicted positions of all persons using respective tracker
A	Distance threshold for correspondence matching
t_{last}	Last time when the person was observed
B	Maximum invisibility time after which a person is said to have left
Learning and modeling	
S_i, $S^i_{ahead,1}$ and $S^i_{ahead,2}$	Position of the ith person, first, and second person ahead of person i
S^i_{ahead}	All persons ahead of S_i
θ_i, $\theta_{i,1}$ and $\theta_{i,2}$	Current orientation of person i, angle of first and second person ahead of person i relative to θ_i
α_{ij}	Angle of person j relative to the walking direction of person i
v_i, $v_{i,1}$ and $v_{i,2}$	Linear speed of the ith person, first, and second person ahead of person i
ω_i, $\omega_{i,1}$ and $\omega_{i,2}$	Angular speed of the ith person, first, and second person ahead of person i
$\theta_{i,goal}$	Angle to goal of person i
Support Vector Regression	
f_{motion}	Motion feature
W	Weight
B	Bias
y, $O(f_{motion})$	Output produced by SVR
$L_\vartheta()$	Loss function
ϑ	Error tolerance
Performance	
$\tau_{i,predicted}$	Predicted trajectory
$\tau_{i,actual}$	Actual trajectory
$d\left(\tau_{i,predicted}, \tau_{i,actual}\right)$	Average distance error
$\chi\left(\tau_{i,predicted}, \tau_{i,actual}\right)$	Prediction accuracy
$E\ (a, b)$	Distance between a and b less than a threshold ε?

https://doi.org/10.1017/S0263574721000679 Published online by Cambridge University Press

1. Introduction

The classic problem of robot motion planning asks the robot to go from the current place to the goal, while avoiding obstacles and optimizing against the objectives of path length, smoothness, etc. Despite the success in the field, the robots are still not considered safe for operating amidst humans as they do not consider the social conventions expected by humans. A human expects an oncoming robot or a robot ahead to give way by drifting at a socially compliant side, while also not coming too close even if that is otherwise safe.

The paper has an application in social robot motion planning. Robots are now operating in workspaces amidst humans, and therefore should obey the same social etiquette as humans. Humans often expect each other to act in a certain way and if a human does something contradictory, it causes social discomfort. Therefore, robots must understand and display the same conventions. If the robots are to operate amidst humans, they must adhere to social conventions. Our long-term aim is to make a navigation model, where robots show social etiquette learned from humans. Our focus is on modeling rare behaviors that are often neglected, and therefore cannot be learned from a general database of human trajectories. The learned model will be loaded on a robot that will act socially when moving alongside humans.

In addition to this, it is important to predict the future trajectory of each person in real time. The predicted trajectories are useful for performing safe navigation of the robot. A robot must know the intention and the trajectories of all dynamic entities for planning. In the case of multiple robots, the trajectories can be communicated. However, for humans, the trajectories and intent must be guessed by the robot. The guess must incorporate the social conventions as displayed by humans.

Another common application is crowd simulation, where a study of human behavior enables measuring the efficiency of crowd management systems, possible situations in case of an emergence, scheduling events to avoid congestion, etc. Simulations enable computing time within which the facility can be vacated, congestion levels at different times, average speeds possible, etc. This requires socially modeling the different agents and simulating them using learned models.

With the advancement of technology, detection and tracking are widely used in the area of robotics to extract the trajectory of humans and to model the walking patterns, human behaviors while moving, etc. Robots must learn the observed human navigation characteristics. In the context of robotics, it is essential to calculate the present and future location of each person with respect to the robot for collision-free and socially compliant navigation. In various applications, pedestrian detection and their future location prediction can be used for discovering abnormal activity or safety purposes. Autonomous vehicles are getting more attention recently because their driving is more secure [1] and for transportation efficiency [2].

In general, the problem of learning the behavior from such data are hard because each person has a different behavior that depends upon the situation, person, and also the behavior of the neighbors. Our approach here is to separately learn and model the different atomic behaviors while accepting that the choice of the displayed behavior may be a personal endeavor and hard to model. As an example, in a situation, some people may prefer overtaking a person, and some may prefer following the person. In a situation, some people may prefer waiting for an oncoming person, while others may aggressively walk in.

The choice of behavior here is an unorganized chain in dense environments. Consider the situation where many people are attempting to get downstairs, which has a limited capacity. Hence, a queuing will happen and the people will follow an organized queue. The difference between an unorganized queue and an organized queue is that in an unorganized queue some lateral movement is possible and the people may not necessarily be exactly behind one another. Similarly, such a behavior develops while navigating in dense environments within corridors, walkways, entry points, etc. Such behavior is rarely studied, though it forms a dominant behavior faced by humans in everyday lives.

3D object detection plays an essential role in the visual perception system of robots. In general, the robots are equipped with several sensors such as lidar, cameras. Unlike many popular pieces of

https://doi.org/10.1017/S0263574721000679 Published online by Cambridge University Press

literature, a 3D LiDAR is used for tracking human behavior for learning instead of a video camera to limit the noises coming from the projection of a 2D camera into 3D. The laser scanner provides accurate depth information, which increases the performance and accuracy. We aim at highly accurate 3D localization of humans in the real world. The people typically may have small distances between themselves in situations and noises can be highly dangerous in such scenarios. Some recent approaches draw 3D window in voxel grid and use CNN [6, 7]. Also, a stereo camera is not used because it typically has a highly limited field of view.

We had some past experience collecting such data using actors. However, the behavior of the actors was observed to be highly biased toward the instructions given, which does not display a natural behavior. Several other works have been done for replicating the movement of mobile agent such as refs. [3]–[5], but these models are applicable for simple movements and are not good enough for modeling the motion of the mobile user in natural scenarios, that's why these are not practical in real environments. Our approach deals with the natural environment of the crowd, and to model their behavior using their trajectory with their social interactions. Hence, for this paper, all data are collected from natural environments, which also pose the problems of occlusions, noises, and limited field of view that had to be dealt with by detection and tracking algorithms.

The recorded human motion is then subjected to learning. Conventional approaches attempt to regress the parameters of algorithms like the social potential field for the modeling of human behavior. However, human behavior may have characteristics that are not represented in the model and thus are never displayed. Therefore, a learning-based approach is used for the modeling, where the behavior is a function of the distances and angles of the neighbors and the intended direction of motion. Many learning-based approaches aim to learn a complete navigation function, which is highly dependent upon the person and context, and hence the model generalizes only to situations similar to the training data. Furthermore, the learned model may not learn the rare behaviors that occur infrequently and may mostly learn the behaviors of avoiding the neighboring people and obstacles. Here, we instead learn a specific behavior of motion within an unorganized chain.

In this paper, we deal with a densely crowded place on our university campus, where people are moving toward their class and some of them are leaving the class after completion of lecture. All people move naturally, and a natural behavior is observed here. Here, an unorganized chaining behavior is observed and analyzed when people go in a bunch toward the same goal. Each individual has slightly different dynamics, constraints, and behaviors. While moving, a homogenous interaction corresponds to how two persons in close proximity affect each other's motion.

It is observed that the trajectory predicted by the model is correct and closer to the ground truth. We also compare the performance of our approach. The performance is compared with the artificial potential field method. It is observed that the learned model is much better than the potential field method. The trajectory predicted by our approach is more natural, including modeling the turns, and gives an overall smaller average distance error when compared to the potential field method.

Main Contributions: The main contributions of the approach are as follows:

(i) Unorganized chaining behavior is introduced and analyzed. This behavior is observed when a group of people moves together toward the same goal, for which they need to pass through a gate or corridor with a limited capacity. A chain is formed automatically so that people can pass through without any collision, here referred to as the unorganized chain. The behavior is different from an organized chain where every person follows the person in front, for which a lot of work has been done in the literature.

(ii) The unorganized chaining behavior is modeled in a data-driven approach and following learning-based modeling. Data are collected by a Quanergy 3D LiDAR and using the same data, all moving people are detected and tracked. While tracking, we deal with the challenges of occlusions and correspondence matching due to a large crowd density. Kalman filter is used for preparing the trajectory database, which is used for training the model.

https://doi.org/10.1017/S0263574721000679 Published online by Cambridge University Press

(iii) The study is done by observing the natural motion of the humans in natural spaces, without incorporating actors who are specifically briefed for the task. Many socialistic studies use human actors who behave synthetically and behave as per the briefing given that result in non-socialistic results. In this study, the humans displaying the explicit behavior are tracked while they were moving naturally as per their daily routine.
(iv) The results demonstrate better performance when compared to the widely used potential field approach. The trained model can predict a person's future trajectory better when compared to the artificial potential field method.

Limitation. It is also good to consider the major limitations of the work, which are as follows:

(i) The paper analyzes a single behavior only. Humans display a wide range of behaviors that must all be modeled for a realistic overall simulation and planning. While a lot of literature is available on obstacle avoidance, goal-seeking, and other behaviors, it cannot be ascertained that these are the only behaviors in human navigation. Further, the boundary cases between the unorganized chaining behaviors and the other behaviors are yet to be identified. Specifically, the interaction between unorganized chaining, obstacle avoidance, and interaction between different unorganized chains are interesting aspects to study in the future.
(ii) The proposed work does not exhaustively cover all types of unorganized chaining possible. The human spaces are easy to classify as narrow resulting in chaining and sufficiently wide spaces where no chaining takes place, and the models are generalizable to most narrow spaces that can result in unorganized chaining. However, studying human behavior across different geographical conditions and maps can uncover new patterns of unorganized chaining.
(iii) In the assessment of socialistic behaviors like overtaking, goal-seeking, etc., the behavior of the person is largely affected by the social interaction with the neighbors. Friends generally keep a small socialistic distance from each other while moving, while the distance increases in the case of a lack of acquaintance between people. Similarly, the speed is larger in case of urgency. The unorganized chaining behavior is specifically lesser prone to such factors as the chain moves much slower and the distances maintained between people are reasonably smaller. This is attributed to the fact that everybody following a chain intends to move out as quickly as possible. The effects of human personal attributes and acquaintance level still need to be formally studied.
(iv) The overall aim of the work is to make a robot behave socially accounting for the same etiquettes as displayed by humans. Robots may have different constraints and social acceptance and such factors affect modeling, which needs to be studied in the future.
(v) The learned model is tested on simulations, which needs to be extended to real mobile robots operating amidst humans. The work assumes a fairly accurate knowledge of the people ahead of the robot, which will require vision algorithms that have not been developed yet.

2. Literature survey

2.1. Pedestrian detection and tracking using a 3D LiDAR

3D LiDAR provides the data in the form of a point cloud and most of the existing approaches encode the point cloud into voxel grid representation. In Vote3D [6] and Sliding Shape, [8] geometric features were extracted from the point cloud and then an SVM classifier was applied on 3D grids, which were encoded with the geometric feature. It used depth information for overcoming the major difficulties such as occlusion, variation in texture, shape, viewpoint, clutter, sensor noise, and illumination. Recently, Convolutional Neural Network (CNN) is widely used in the area of object classification and detection due to its accuracy in the result. Here, we have mentioned some state-of-the-art CNN-based approaches for object detection in the 3D point cloud. In a recent approach, CNN [7, 9, 10] was used to improve the feature representation, but it required a high computation. This computational cost can be improved

https://doi.org/10.1017/S0263574721000679 Published online by Cambridge University Press

by projecting the point cloud in an image plane [11] or ground plane [12, 13]. Here, the data were represented in a 2D point map and a single 2D fully convolutional network was used for predicting the full 3D bounding box [11]. 3D LiDAR provides a dense point cloud, which contains more information about the environment when compared to other camera and also provides rich information about the person's location. In literature, [14, 15, 16], pedestrians were detected using 3D LiDAR and the result analysis showed its effectiveness and accuracy of detection. Lidar generates a point cloud of the object, but when the object goes away, then its point cloud density reduces. This is a challenging problem because it directly affects the result of detection and that's why density enhancement can solve this issue and easily detect the human in a long range [14]. These dense point clouds can help to detect and localize the pedestrians more reliably. In another approach [17] a new feature for human classification in a sparse and long-range point cloud was used from tracking and trajectory analysis. It can be utilized for classification so that in case of a sparse point cloud, all humans are detected correctly, even those who are far away from the laser scanner. Recent approaches such as PointPillars, [18], Frustum PointNets, [19], AVOD, [12] and MV3D [13] are based on deep neural network architecture that takes Lidar point clouds and RGB images as input for generating complex features of the object from 3D point cloud and predict oriented 3D bounding boxes. These methods achieve a high detection performance. In MV3D, [13], the sparse 3D point cloud was encoded with multiple view representation and the network was formed by two subnetworks. The first was for object proposal generation, which was used for generating a 3D candidate box and the second one was for multi-view feature fusion that combines features from multiple views.

Tracking has also been applied to detect humans for estimating their position. Recently, various approaches have been proposed for tracking such as Kalman filter, [20], Particle filter, joint probabilistic data association [21, 22]. In this paper, we have used the Kalman filter for tracking. It was also used to solve the correspondence problem and to separate the trajectory of each person. Similarly, several studies are there in the area of tracking in suburban districts where the background is simple and sometimes occlusions occur [23, 24, 25].

In a recent work, Dequaire et al. [26] proposed an end-to-end trainable framework that predicted a fully un-occluded occupancy grid from the raw input. This framework learned the model in an unsupervised manner. This approach was based on deep tracking [27]. Here, the framework focused on learning to track in a real situation where the sensor can be static or dynamic. Methods based on deep tracking perform better than the traditional approaches and predict the location of the human, car, and cyclist accurately even in case of complete occlusion. Tracking pedestrian in traffic is a challenging problem because of the cluttered background and high occlusions. Gao et al. [28] proposed a layered graph model for reliable multi-pedestrian tracking using RGB-D image. To solve the high 3D occlusion, a new strategy was proposed. Real-time tracking is a challenging problem and most of the algorithms suffer from low accuracy and uncertain background. Held et al. [29] proposed an approach for real-time tracking, namely annealed dynamic histograms. In this approach, 3D shape, colors, and motion cue information were combined so that the tracker becomes capable of tracking all moving objects more accurately than using one or two cues. Recently, Lee et al. [30] proposed a method of tracking based on the road pedestrians using multiple moving camera recordings. It also determined whether a person belonged to one or more cameras using an associated likelihood of a tracked person, which was computed based on the motion cue and appearance feature of the tracked person.

The approach of object detection in 3D LiDAR is only applicable for detecting their location in point cloud, but this is not enough for various applications such as trajectory prediction, intent recognition, and social interaction. Pedestrian orientation from a single frame image was estimated by considering this problem as a multi-classification problem, where orientation was discretized into a fixed number of bins [31, 32]. Approach in ref. [31] consisted of three steps. Initially, it had a bounding box, which was defined in terms of histogram features, its aspect ratio, and size. Then, the viewing angle was predicted under the assumption that the object instance is contained in the bounding box. Finally, a single classifier was used to tune the viewpoint so that it can confirm that the object exists in the estimated bounding

https://doi.org/10.1017/S0263574721000679 Published online by Cambridge University Press

box. Another approach considered pedestrian orientation as a regression problem [33, 34] and estimated the pedestrian pose.

The different approaches are also summarized in Table I. The tracking presented in this paper has the unique challenges of a very small amount of data, high occlusion, a low-resolution lidar sensor, and a further small resolution of data due to distant people; because of which it is neither possible to learn data-rich models, nor to use conventional detection techniques. Hence, the aim is to complement detection with tracking for accurately extracting the trajectories from the point cloud data.

2.2. Prediction algorithm

Human motion has been widely studied, and based on research in motion analysis, several models are proposed that are used to mimic the movement of the mobile agents and to simulate their mobility patterns using algorithms such as Monte Carlo Simulation, [35], Levy walk mobility model, [5], Gauss–Markov mobility [36]. These approaches are not applicable for the real-world scenario, but there are several efforts that have been tried for making these models applicable for real worlds. Another widely used approach of trajectory prediction based on Markov models is the Hidden Markov Model. In ref. [37], a specific rule was proposed for moving from one place to the other. Later, its performance was improved by adding the user behavior of the particular situation [38].

Machine learning and deep learning are also involved in improvements. Recent approaches based on deep learning perform a multiuser multistep trajectory prediction [39]. LSTM network has been used to understand the mobility pattern of the mobile agent from a historical trajectory, and then it is used to predict the future trajectory. Another approach based on deep learning is that it extracts the visual feature of the target object and sequential position information about the object. Information combined was used to predict the future trajectory [40]. Understanding the activity of the people in the real world and identifying their coherent behavior makes the prediction of the trajectory more reliable. In earlier work, a framework was proposed where multiple individuals were tracked and their activity such as walking, interaction was observed by isolating them. These coherent behaviors make the prediction of the trajectory more effective and natural. We have applied a similar idea in our paper. First, we track all people walking in natural scenarios and their activity was analyzed. Then, we obtain a natural unorganized chain formed by a group of people moving from one place to another, followed by trajectory prediction.

Most of the models learn the movement pattern using a trajectory database for specific behavior. Trajectory for specific behaviors is clustered, which is used by the models [41, 42] to predict the future location of the person. In a recent work, the recurrent neural network was successful for sequence prediction and social LSTM [43] based on handcrafted functions such as social force was proposed that learned the human movement pattern and then predicted the future route. Trajectory prediction algorithms are also involved in robotics [44, 45, 46, 47]. These approaches make a mobile robot capable of finding safe navigation in a crowded environment without any collision.

Unlike the approaches, the aim here is to learn a specific behavior that is neither actively studied, nor is observed in generic navigation to be naïvely observed. Unlike the discussed approaches, synthetic means to generate a large volume of data were rejected to make the data realistic, while the learning was done on a limited amount of data collected on the specific behavior. The approach hence does not surrender to mere parameter optimization on known motion models that may not capture hidden behaviors, uses simple learning techniques so as not to require a very large amount of data, and is modeled on a specific behavior to make it easier to learn as a machine learning problem.

2.3. Social robot motion planning

Even though the focus of the paper is for modeling human behavior, the social robot motion planning techniques are very briefly discussed since they form a dominant application area. Mobile robots have

https://doi.org/10.1017/S0263574721000679 Published online by Cambridge University Press

Table I. *A summary of the literature on pedestrian detection and tracking using a 3D LiDAR.*

S.No.	Paper	Description
1	[6]	3D object detection using a sliding window approach. Sparse searching with a voting scheme was used for algorithmic speedup, equivalent to that obtained for a sparse CNN
2	[7]	3D object detection using CNN. Sparse convolution using a voting scheme (similar to ref. [6]) was used for a speedup creating a new benchmark
3	[8]	3D object recognition using depth maps and SVM. The training was done using synthetic depth maps created from CAD models. The model showed higher average precision when compared to DPM and RCNN
4	[9]	CNN-based approach including Region Proposal Network for geometric features and 2D image features for a 2D+3D joint object recognition network. Two-hundred times faster than sliding shapes
5	[10]	3D fully convolutional network for 3D detection. The performance was verified on the vehicle detection problem using the KITTI datasets
6	[11]	An improvement over [10]. The approach presents the data in a 2D point map and uses a single 2D fully convolutional network for predicting the full 3D bounding box and object confidence score
7	[12]	Recognition using point cloud and RGB image for generating sharable features. Region Proposal Network and second stage detector were used for category classification and 3D bounding box prediction for road scenes
8	[13]	Multi-view 3D network for 3D object detection in road scene using both lidar point cloud and image. An object proposal network generated a 3D candidate box and a multi-view feature fusion network encapsulated features from multiple views
9	[14]	Density enhancement in the point cloud (as far-away objects in a lidar have a poor resolution) with an evaluation metric of the local coordinate system to select good shape parameters and RBF-based interpolation. The approach also checked for object geometric shape fitting
10	[15]	Fast object extraction and classification in lidar point cloud by utilizing a deep learning-based object appearance model and contextual scene analysis to separate between façade and short objects
11	[16]	KAIST multi-spectral dataset was provided for autonomous vehicle scenarios in driving in all time slots such as morning, evening, and night
12	[17]	Online learning approach based on 3D LiDAR cluster detector, multi-target tracker, human classifier, and sample generator for detecting humans by a mobile robot. The detections were used to further train the network
13	[18]	3D object detector by representing point cloud features organized in a vertical column or Pillar to be usable by a standard 2D convolutional detector for predicting a 3D bounding box. This approach could run at a speed of 62 Hz
14	[19]	Object detection technique by using a mature 2D detector to reduce the search space in the point cloud. Advanced 3D deep learning was used for localization in the presence of occlusion or a sparse point cloud

Table I. *continue.*

S.No.	Paper	Description
15	[20]	A mathematical model was proposed for studying the motion of any object. Our work uses a similar technique for tracking humans based on point cloud data
16	[21]	Single target tracking was performed by using a filter-based approach and nearest-neighbor association technique for data association
17	[22]	Joint Probabilistic Data Association was used for multi-target tracking, with accurate detection in noisy and occluded environments. The approach showed a better performance in molecular application and pedestrian tracking
18	[23]	Pedestrian detection in a busy zone using depth and appearance-based information from the video. Ground plane estimation was also done. The approach was unable to provide accurate 3D information
19	[24]	Stereo image-based pedestrian detection, localization, and tracking from a moving vehicle. The region of interest was segmented, and features extracted were used for classification. Tracking was used to estimate the velocity of the person, which is useful for path planning
20	[25]	Multi-cue vision system for pedestrian detection and tracking. Cascade module with a tight integration of stereo-based ROI generation, shape-based detection, textured classification, and dense stereo-based verification was used
21	[26]	Tracking buses, cars, cyclists, and pedestrians in both static and dynamic platforms using RNN for estimating the state of the object. The RNN could predict an un-occluded occupancy grid directly from the raw point cloud
22	[27]	End-to-end object tracking that is deep tracking. It could directly map the raw sensor input to the object tracking in the sensor space. It could work with occluded objects, which is done as a deep learning task
23	[28]	A layered graph model in RGB-D image for real-time multi-pedestrian tracking. Depth and vision data were integrated with layer level constraints and formed a layer level graph. The approach could handle occlusions and give correct data associations
24	[29]	Tracking objects in real time by combining color, 3D shape, and motion cue features using a probabilistic framework to robustly track objects against several factors of variation and occlusions
25	[30]	Pedestrian tracking from multiple moving vehicles. The approach determined whether the tracked person belongs to one or multiple moving camera fields of view using associated likelihood
26	[31]	3D multi-view object detection and localization in multiple viewpoints that change the feature values and handling a variation in the aspect ratio and size of the object
27	[32]	The paper compared 2D and 3D object recognition. Features were extracted from detected bounding boxes using modern encoding and in most of the cases performed better than the state-of-the-art methods that contain 3D information
28	[33]	Multi-view object model of different object categories for extracting any virtual image features. 2D appearance was represented as a distribution of low-level, fine-grained image features in the number of discrete viewpoints
29	[34]	Node-splitting approach for regression tree training and its integration with a regression forest framework for accurately estimating the pose and direction of the object

become popular among people. So, for making it acceptable to society, it is necessary that the robot should be aware of the environment and should guarantee human safety and comfort. The real environment is uncertain, dynamic, and crowded. To ensure human safety, several approaches have been proposed [48, 49] so that robot the becomes aware of the human. Furthermore, path planning method has been developed for motion planning, which is also goal-oriented [53]. In the real world, people interact with each other while walking. This information makes the motion planning algorithm more reliable and makes the robot socially acceptable. Gap detection is also useful for the robot's safe navigation and 3D information of the unconstrained environment helps the robot for gap detection [55].

In a recent approach, six harmonious rules were proposed [50] that a mobile robot should follow for safety purposes in the presence of humans or other robots, but this approach does not include social interaction information between the persons. Early work used this social information for making a mobile robot able to interact with the human in a social complaint way. Kretzschmar et al. [51] proposed an approach that models cooperative human navigation behavior and using the pedestrian trajectory, the robot learned how to replicate the specific behavior in a certain situation. Learning for pedestrian trajectory makes the robot socially acceptable because while moving in a crowded place, it follows the pattern that the human used which is completely natural and safe. To make a safe robot navigation, the robot slowing down behavior [54] is also useful for motion planning. Similarly, Truong et al. [52] proposed a proactive social motion model that enabled the mobile robot to consider the social interactive information that is characteristics of the human and human group, while navigating safely and socially in a crowded and uncertain environment.

3. Overall approach

In this paper, we observe the natural behavior of humans in a crowded environment and model this behavior. The model is used by our framework for trajectory prediction so that the mobile robot can plan a collaborative and socially compliant collision-free trajectory. Primarily, the scenario of a group of students leaving their class when the lecture gets over and moving toward the building exit is considered. Due to a large crowd and a limited walking space, a chain is formed automatically, which is referred to as an unorganized chain. The behavior is generic to any such queuing by the humans, which happens in many situations, primarily because the walking space at several places is restricted to narrow corridors for indoor scenarios.

The overall approach is summarized in Fig. 1. The main focus of this paper is the use of a 3D LiDAR for capturing data. Here, we detect all the moving people in a 3D point cloud. The detection of the pedestrian is performed by spatial changes in successive point clouds frame. These point clouds are represented in an octree data structure and recursive comparison is performed for spatial change detection, which is represented as a difference in voxel configuration. Later, a clustering algorithm is applied for isolating each person and its centroid value represents each person. The centroid value of all persons is passed to a Kalman filter for accomplishing the tracking. Since the data are recorded in a real environment, the density of people is often very large, and the people move in close proximity. Due to this, we are unable to isolate each person for all point clouds. Also, some portion of the trajectory of a person may be erroneously assigned to another person's trajectory leading to wrong results, commonly referred to as the *correspondence problem.*

The Kalman filter is used for removing the limitations and tracking, which is also used to solve the correspondence problem. For each person tracker, we predict its location for tracking. For dealing with the correspondence problem, all trackers predict the future position (without using observation). After predicting, the location of all people is compared with the observation. An observation is mapped to the tracker that has the closest distance and subject to an additional threshold, which is calculated experimentally. Every instance of tracker provides a trajectory for each person correctly. The trackers are used to build a trajectory database, which is later used to model human behavior in an unorganized chain.

https://doi.org/10.1017/S0263574721000679 Published online by Cambridge University Press

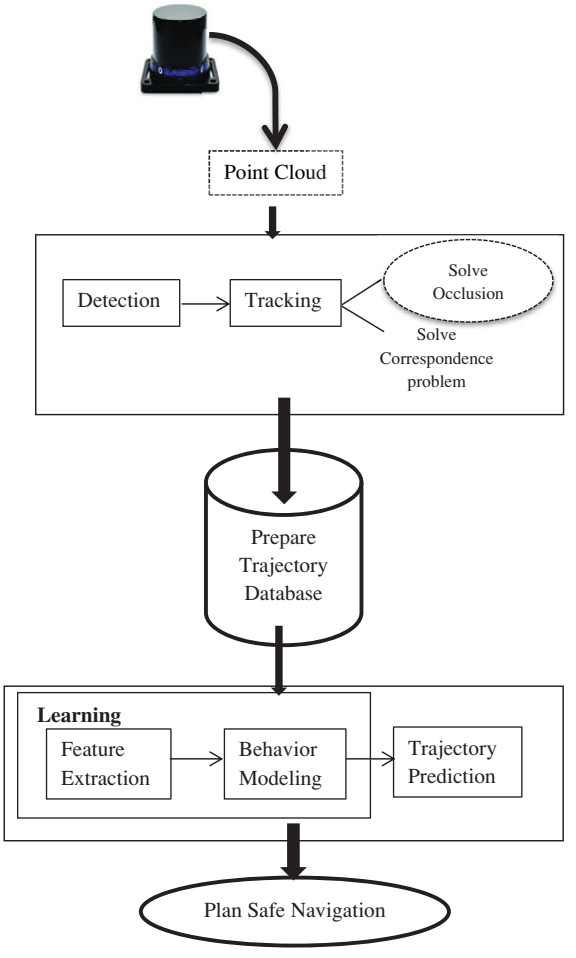

Figure 1. *Workflow model.*

After the trajectory database is prepared, each trajectory is analyzed. In doing so, homogenous interaction is considered. For this, some important features are extracted from the trajectories such as velocity, spatial distance from neighbors, angle of goal, which are used to train a behavior model. These features are fed into a support vector regression model to learn the behavior. Since the robot would constantly track and correct the trajectory of the humans, only a short-term prediction is required. This predicted trajectory is useful for the mobile robot in a crowded place so that the robot can plan safe navigation and behave socially while moving with humans.

4. Detection and tracking

4.1. Dataset pruning

Since the data are collected in a natural scenario of moving people, hence there are many noises present in the dataset. Many times, people come very close to each other. While walking, the people swing their hands and legs and it touches the other person. This leads to an error in detection. Our primary objective is to find the spatial relation that humans keep while walking in natural scenarios. In detection, more than one pedestrian can get assigned to the same cluster. This problem is solved by dataset pruning. The human head is stationary when compared to other body parts and that's why the point cloud is passed

https://doi.org/10.1017/S0263574721000679 Published online by Cambridge University Press

through a *pass-through filter*. The point cloud $P(t)$ is processed by this filter based on a filter field. In our case, height is used as the field and its value is calculated experimentally, which improves the accuracy of the clustering algorithm. Point cloud after applying a pass-through filter is given by Eq. (1) for the input point cloud $P(t)$.

$$P'(t) = \{p \in P\ (t) : X_{\min} \leq p_x \leq X_{\max} \wedge Y_{\min}\ \leq p_y \leq Y_{\max} \wedge z_{\min} \leq p_z \leq z_{\max}\} \tag{1}$$

Here, $X_{\min}$ and $X_{\max}$ is the range of X coordinate, $Y_{\min}$ and $Y_{\max}$ is the range of Y coordinate, $Z_{\min}$ and $Z_{\max}$ is the range of Z coordinate that has to be accepted by the pass-through filter. The resultant cluster is treated as space occupied by the person.

4.2. Detecting dynamic entities

We have a dataset of moving pedestrians in the point cloud format and detection of all moving persons is the next step of trajectory extraction. An octree is a tree-based data structure that organizes the 3D data. The 3D space is produced by recursively subdividing it into eight octants, where each internal node has exactly eight children. Indexing of 3D point cloud data using octree is performed by dividing the 3D boundary into eight octants, which are further recursively subdivided until the octant consists of a single point or it reaches a certain threshold or a maximum depth.

To detect the motion, background subtraction is applied by taking a difference of the consecutive frames, and for this, the first octree representation is used for the point cloud data. Once octree of the current point cloud frame $P'\ (t)$ and previous frame $P'\ (t-1)$ is built, spatial change is detected by recursively comparing the tree data structure of the octree and this is represented by a difference in the voxel configuration. Points corresponding to the moving people are calculated as given by Eq. (2)

$$Moving\ (t) = \left\{p \in P'\ (t) : \min_{q \in P'(t-1)} d\ (p, q)\ > \eta\right\} \tag{2}$$

Here, d is the distance function and η is the threshold. If $P'\ (t)$ is $\{(x_i, y_i, z_i)\}$, now the equation will be given by Eq. (3).

$$Moving\ (t) = \left\{(x_i, y_i, z_i) \in P'\ (t) : \min_{j \in P'(t-1)} d\left((x_i, y_i, z_i)\ , \left(x_j, y_j, z_j\right)\right)\ > \eta\right\} \tag{3}$$

4.3. Statistical outlier removal

Raw 3D point clouds obtained from a 3D scanner are often contaminated with noise and outliers. These are the consequence of a mistake in data collection, sensor imperfection, occlusion, or some other mishandling. Usually, the outlier is a point that is far away from the other observations in the dataset. There are several points in our dataset whose homogeneity is not matched with surrounding neighbors. These types of points are referred to as the outlier. The outliers are responsible for reconstructing a wrong geometry, which leads to corruption in the results. This is why these irregularities should be removed so that all points are distributed in a regular manner. Here, we use a simple and efficient method based on the geometric characteristics for removing outlier. Local density information is one geometrical characteristic that is used here because the outlier point is usually inconsistent with the local point density.

In this method, we calculate the mean distance of a point with all of its neighbors. We assume that the resulting distribution is Gaussian with a mean and standard deviation. All those points are considered as an outlier whose mean distance is outside a certain interval defined by the mean and standard deviation. These points are removed from the point cloud.

In Algorithm 1, *Moving*(t)) is the result of spatial change detection. It contains a point cloud of all moving people along with noise and forms an input to the algorithm. Algorithm 1 removes noise from the input, which can corrupt the result. Line 1 takes an empty set, *Moving'*(t), as the prospective output that is incrementally populated. Line 2 loops through all points in the input point cloud. Line 3 computes $\delta\ (p)$, the set of k neighboring points of point p. Line 4 computes $D\ (p_i)$, the mean distance of point p_i with k neighbors. Line 5 uses $\overline{D}$ as the average distance of all $D\ (p_i)$ with a standard deviation of $\sigma\ (D)$.

https://doi.org/10.1017/S0263574721000679 Published online by Cambridge University Press

Algorithm 1. Statistical Outlier Removal Algorithm

Input: Point clouds with outliers (*Moving*(*t*))
Output: Point clouds without outliers

1: Let *Moving'*(*t*) = Empty set representing inliers
2: **for** all point p_i in *Moving*(*t*)
3: find set δ (p), set of k-nearest neighbors of point p_i
4: calculate average distance of p_i with all the neighbors $D(p_i) = \frac{\sum_{q \in \delta(p_i)} d(q,p_i)}{k}$
5: if $D(p_i) \leq \overline{D} + \alpha\ \sigma\ (D)$
6: then add p_i to *Moving'*(*t*)
7: **end for**
8: **return** *Moving'*(*t*)

Algorithm 2. Euclidean Clustering

Input: Point cloud *Moving'*(*t*)
Output: Clusters C

1: $C \leftarrow \emptyset$
2: fixed(p) $\leftarrow$ no $\forall\ p \in P$
3: **for** $p \in P$: fixed(p)=no
4: $c \leftarrow \{p\}$//Seed point is added
5: **while** true
6: $\mathfrak{R} \leftarrow \{\Upsilon : \min_{q \in c} d(\Upsilon, q) \leq \varepsilon \wedge$ fixed(Υ)=no $\}$// Compute all points belonging to the same cluster
7: if $\mathfrak{R} = \emptyset$, break
8: $c \leftarrow c \bigcup \mathfrak{R}$//Computed points added to the cluster
9: fixed(Υ)=yes $\forall\ \Upsilon \in \mathfrak{R}$//Points belonging to the same cluster are marked as processed
10: **end while**
11: add cluster c to C
12: **end for**

If the mean distance of a point p_i lies under $\overline{D} + \alpha\ \sigma\ (D)$, then the point is considered as an inlier and added to *Moving'*(*t*) in line 6, otherwise, the point is an outlier and is rejected. α is an algorithmic parameter.

4.4. Cluster segmentation

Now we have the point clouds of all detected humans without outliers. It is essential to classify and extract each individual from the point clouds for trajectory extraction. Clustering is one of the most famous techniques to find homogenous subgroups in the dataset, such that the data point belong to subgroups are of the same type according to a similarity measure. Therefore, Euclidean distance-based clustering algorithm and region growing is applied for performing the task of separating the human from the point cloud.

This algorithm starts from a set of random seed points representing the clusters and then builds its kd-tree for the nearest neighbor calculation. Cluster size increases by adding neighboring points to the cluster and this insertion of point stops when there are no points left in the Euclidean space of seed points. The complete procedure is described in Algorithm 2.

Algorithm 2 explains the steps of Euclidean clustering, where *Moving'*(*t*) is the point cloud of all moving people after removing noise, taken as an input. Line 1 initializes C, the set of all clusters as an empty set that will be iteratively populated. Line 2 initializes *fixed*(p), representing whether the point p is processed or not, as initially unprocessed for all points. Line 3 loops through only the non-processed points in the point cloud. Every such point p is taken as a set of a new cluster c. The loop in Lines 5–10

https://doi.org/10.1017/S0263574721000679 Published online by Cambridge University Press

Algorithm 3. Kalman Filter

Input: $\mathbf{S_i}(t-1)$, $\mathbf{p_i}(\mathbf{t}-1)$,$\mathbf{z}_{\kappa(\mathbf{i})}(\mathbf{t})$
Output: $\mathbf{S_i}(\mathbf{t})$,$\mathbf{p_i}(\mathbf{t})$

1: $\hat{S}_i(t) = AS_i(t-1)$
2: $\hat{p}_i(t) = Ap_i(t-1)A' + W(t)$
3: $K_i(t) = \hat{p}_i(t)H'/(H\hat{p}_i(t)H' + R(t))$
4: $S_i(t) = \hat{S}_i(t) + K_i(t)(z_{\kappa(i)}(t) - H\hat{S}_i(t))$
5: $p_i(t) = (I - K_i(t)H)\hat{p}_i(t)$

iteratively grows this cluster. At any time, the set of all points unprocessed that lie at a distance of less than ε to any point in the cluster c is computed, called R in Line 6. Here, ε is the distance threshold for clustering. Since these points are reasonably close to an existing point in the cluster, they are added to the cluster c in Line 8. Line 7 is the condition where the cluster can no longer be expanded, and therefore the loop breaks. Line 9 marks all the added points as processed. Line 11 adds the grown cluster c to the set of all clusters C. C is the output of the clustering algorithm that contains clusters of all moving persons separately.

Subsequently, the centroid of the cluster is calculated, which is used to represent the position of the person, given by Eq. (4).

$$z_j(t) = \frac{\sum_{p \in c_j(t)} p}{size(c_j(t))} \tag{4}$$

Here, $c_j(t) \in C(t)$ is the jth cluster (person) and the area of the cluster is the space acquired by the human, which may change according to the situation like space availability. The person is represented by the centroid $z_j(t)$. The detection is shown in Fig. 2.

4.5. Kalman filter

Once all the moving people are detected, then tracking is performed. Kalman filter is applied for tracking. This is a recursive predictive filter based on the state-space technique. This filter doesn't store previous measurement detail. Due to its less computational time and optimal results, the filter is widely used in the majority of the fields. The filter assumes that the uncertainties are Gaussian in nature. The algorithm is structured in two distinct phases, namely *motion update* and *measurement update*.

The motion update phase is also known as the predictor state, where the next state $\hat{S}_i(t)$ is predicted using a linear model and the current state $S_i(t-1)$ and process covariance matrix $\hat{p}_i(t)$ is also calculated using the model and uncertainty of the process model. Here, we represent the state by the XY coordinates and the velocity in these coordinates, $S_i(t) = [x_i(t)\ y_i(t)\ \dot{x}_i(t)\ \dot{y}_i(t)]'$.

In the measurement update phase, the predicted value $\hat{S}_t$ is corrected based on the difference between the actual measured value and the expected measured value from the measurement model. Its mathematical form is shown in Algorithm 3.

Here, $W(t)$ is a Gaussian motion noise. $K_i(t)$ is the Kalman gain. $z_{\kappa(i)}(t)$ is the observation where $\kappa(i)$ is the correspondence function. $R(t)$ is the process noise covariance. A is the state transition matrix given by Eq. (5). H is the observation matrix given by Eq. (6).

$$A = \begin{bmatrix} 1 & 0 & \triangle t & 0 \\ 0 & 1 & 0 & \triangle t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5}$$

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{6}$$

https://doi.org/10.1017/S0263574721000679 Published online by Cambridge University Press

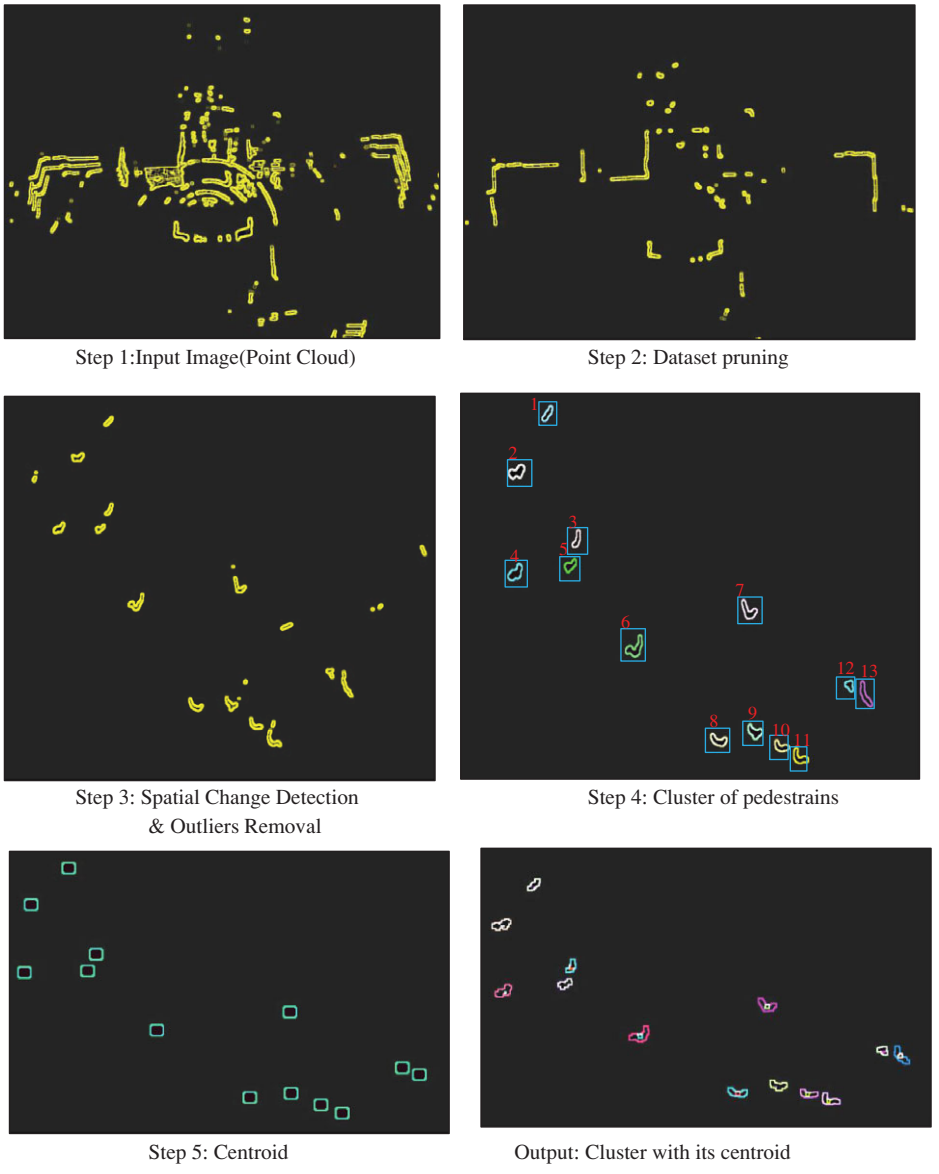

Figure 2. *Procedure of moving pedestrian detection using a 3D LiDAR.*

In Algorithm 3, Line 1 uses the motion model to update the mean state of the ith person $S_i(t\text{-}1)$ using the motion matrix A. Line 2 similarly updates the covariance matrix $p_i(t\text{-}1)$ using the motion matrix A and the process noise $W(t)$. Line 3 computes the Kalman gain ($K_i(t)$) using the updated covariance along with the observation matrix H and the observation noise $R(t)$. Line 4 further updates the state using the computed Kalman gain along with the new observation $z_{\kappa(i)}(t)$ with the correspondence matching function given by $\kappa(i)$. Line 5 further updates the covariance matrix using the Kalman gain and observation matrix.

Figure 3 shows how the tracker reads the points and starts tracking. Here, a 3D LiDAR sensor is used for generating the point cloud and on each frame, the human detection algorithm is applied, which gives the clusters corresponding to all moving people in the frame. Once the clusters have been generated, then its centroid is calculated. The centroid represents the location of the people in a 3D space, which is the input for the Kalman filter.

https://doi.org/10.1017/S0263574721000679 Published online by Cambridge University Press

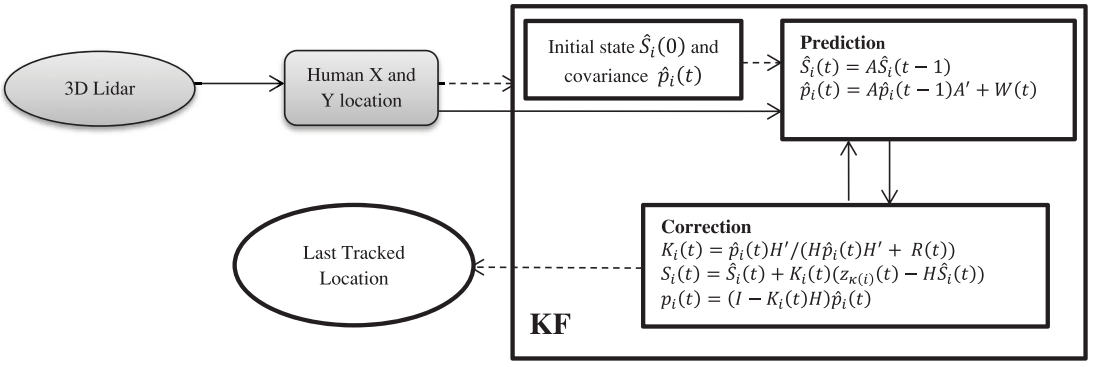

Figure 3. *Block diagram of people tracking using the Kalman filter.*

There is one tracker for every person. The person can arrive or disappear at any frame and occlusion may occur. Therefore, we can't estimate the number of trackers at the beginning of the algorithm. The number of trackers changes along with frames. Consequently, all trackers have to be managed in each frame so that each person is correctly tracked and one can extract their trajectory.

We proposed our own algorithm. When a person disappears for a few frames, then the tracker can predict their position to complete the trajectory. Correspondence is handled by assigning every observation to the closest predicted position in the people being tracked, subjected to a distance threshold (α). If a person is not observed in several consecutive frames (β), then the tracker stops tracking. If correspondence for an observation fails, a new tracker is initialized. The approach is explained stepwise in Algorithm 4.

Line 2 initializes the Kalman trackers (K) as an empty set and Line 3 initializes the output database (T) with another empty set. Both are incrementally populated. C is the set of initially visible people (Line 1), who are directly assigned a Kalman filter in Lines 4–6. Line 4 loops through all such people, Line 5 assign a Kalman track and Line 6 makes an entry in the output trajectory database (T). Line 7 is the main loop. Line 8 detects new people as the current observation. The first problem tackled therein is correspondence matching. Line 9 initializes Ψ to store the current predicted positions for all persons, which is calculated in Line 10. Loop in Line 11 tries to find a correspondence match for every observed person (c). Line 12 assigns a person (c) to the closest track (ψ) using the Euclidean distance between the observed position of a person to its expectation using the Kalman filter prediction. Line 13 additionally checks that the least distance should be less than α to avoid far-off people passing the correspondence matching in case of the detection of a new person. Line 14 computes the last time the same person (ψ) was observed by consulting the entries in the trajectory database (T). If it was seen at the immediately preceding frame, Lines 15–16 apply the standard Kalman filter correction and add the same to the trajectory database. Otherwise, if it has been a while that the person was previously detected, Lines 18–19 predict the motion of the person for all the missing frames and add it to the trajectory database, before applying a correction using the new observation in Line 20. Line 17 additionally necessitates that a person who has not been visible for a very long time (β) should be regarded as ceased to exist and such a person should not be tracked irrespective of the new observation. Lines 21–23 handle the situation wherein a new person enters the system, detected either by a failed correspondence matching (distance more than α) or that the system indicates a person was visible after time β, in which case it is probably a new person. Line 22 initializes a new Kalman filter for the person, while Line 23 appends the observation to the trajectory database.

The human detection logic is delicate and is prone to both False Positives and False Negatives, which are essentially handled by the Kalman filter. A False Positive may occur from the points on the wall detected by the lidar. As the human walks, there is some temporal change in segments of the wall as it gets occluded during the human motion. If a detected dynamic object consistently appears for some

https://doi.org/10.1017/S0263574721000679 Published online by Cambridge University Press

Algorithm 4.

Input: Detected people in all frames
Output: T, tracked locations of all people

$C \leftarrow$ Detect all people at first frame //pass all the centroids of the first frame calculated from the point cloud
K $\leftarrow \emptyset$ //Kalman trackers of all persons in the dataset, initially empty
T $\leftarrow \emptyset$ //time-annotated trajectories of all persons in the dataset, initially empty
for all $c \in C$, //for each centroid, a Kalman filter is instantiated
    append Kalman Filter (c) to K
    append {<c,0>} to T
while not end of recording, for all times t
    $C \leftarrow$ Detect all people at current frame
    $\Psi \leftarrow \emptyset$ //predicted positions of all people being tracked, temporarily stored for correspondence
    for all $\kappa \in$ K, add predict (κ) to Ψ
    for all $c \in C$,
        ψ = arg $\min_{\psi \in \Psi} d(\psi,c)$
        if $d(\psi,c) = \alpha$, // α is a distance threshold used for solving the correspondence problem
            $t_{last} \leftarrow \max_t$T($\psi$).$t$ //last time person was seen
            if current frame time – t_{last}=1, //no occlusionoccurred
                add correct(K(ψ),c) to T(ψ)
            if current frame time – $t_{last} = \beta$
                for t from t_{last} to current frame time //correct for occluded frames
                    add predict (<K(ψ),t>) to T(ψ)
            add correct (K(ψ),c) to T(ψ)
        if $d(\psi,c) > \alpha$ or current frame time – $t_{last} > \beta$\\a new person entered the system
            append Kalman Filter (c) to K
            append {<c, current frame time >} to T

seconds, only then it is tracked as a human, otherwise, it is taken as a False Positive. The parameter was set based on the observed data.

The False Negatives are also handled by the Kalman filter. When humans walk in a queue and for some time, the human speed is very slow, and in that case, the spatial change detection cannot detect the near-stationery human. The Kalman filter continues tracking for some time even when the person is not visible, where the parameter is set based on the observed data. It is assumed that the slow speed of the humans is temporary, and the motion soon continues that gives observations to the Kalman filter.

5. Learning and simulation

When all people have been detected and tracked, then the trajectory of each person is analyzed. In this section, we propose our approach for performing future trajectory prediction in a natural scenario, especially when the human walks in a chain that is neither organized nor centrally managed. Everyone moves toward the same goal at the same time and thus it is their own decision of how much distance they maintain with the others.

Therefore, it is essential to understand the unorganized chaining behavior for performing the task of trajectory prediction, which will be useful for motion planning in this situation. Here, features are extracted from the trajectory of each person for behavior analysis and trajectory prediction.

5.1. Feature extraction

Kalman filter is applied for tracking and it also provides the trajectory of all persons individually. These trajectories are put under the process of feature extraction. The reactive motion of an individual is a

https://doi.org/10.1017/S0263574721000679 Published online by Cambridge University Press

function of the nearby obstacles and people, and the relative position with respect to the goal. The people far away do not affect the trajectory instantly, however, may affect the trajectory only when the person is closer to them. Similarly, the people at the back have a little influence on the trajectory, which is largely affected by the people ahead only. In an organized chain, the behavior of a person depends upon the person ahead only. However, here the chain is unorganized and there may be a lateral disarrangement between and hence the behavior is said to be affected by the nearest two people ahead. Taking more people will increase the model complexity, requirement of data, and is prone to overfitting where the system performs well only on scenarios similar to the training data. To find out the closest people ahead, the person's trajectory is compared with all others.

The features selected are the distance between the two closest persons, the angle to the nearest two people, the velocity of the closest two persons, and the angle to goal. These features are used for creating a training dataset so that the model can learn the behavior and can predict the future trajectory. The creation of a training dataset is one of the most important steps of trajectory prediction because the accuracy of any approach depends on the quality of the training data.

To calculate the features for the person S_i, the first requirement is to consider all the people who are *ahead* of p_i(Eqs. (7)–(8)), out of which close two closest people are identified (Eqs. (9)–(10)).

$$S^i_{ahead} = \left\{ S_j : \cos\left(\alpha_{ij}\right) > 0 \right\} \tag{7}$$

$$\begin{aligned} \cos\alpha_{ij} &= \left(S_i\left(t\right) - S_i\left(t-\Delta t\right)\right).\left(S_j\left(t\right) - S_i\left(t\right)\right)/N \\ &= \left(\left(S_i\left(t\right).x - S_i\left(t-\Delta t\right).x\right)\left(S_j\left(t\right).x - S_i\left(t\right).x\right)\right. \\ &\quad \left. + \left(S_i\left(t\right).y - S_i\left(t-\Delta t\right).y\right)\left(S_j\left(t\right).y - S_i\left(t\right).y\right)\right)/N \end{aligned} \tag{8}$$

$$S^i_{ahead,1} = \left(rank_{S_j \in S^i_{ahead} S_j \neq S_i} \left(d\left(S_j, S_i\right)\right) = 1 \right) \tag{9}$$

$$S^i_{ahead,2} = \left(rank_{S_j \in S^i_{ahead} S_j \neq S_i} \left(d\left(S_j, S_i\right)\right) = 2 \right) \tag{10}$$

Here, N is the norm of the two vectors in the dot product that does not affect the sign and hence neglected, i,j is the person and t is the time frame, d is the distance function on the X and Y values.

The distance that people maintain with each other while walking is an important aspect and plays a vital role while learning the trajectory. A person may not wish to be too close to another to maintain a social etiquette, while the distance may not be very large to make the chain inefficient. The distance features are given by $d\left(S_i, S^i_{ahead,1}\right)$ and $d\left(S_i, S^i_{ahead,2}\right)$

The direction of one person with respect to another person is another crucial information for a model while understanding the motion pattern of people in a particular scenario and trying to predict the future position of a person. The person typically attempts to go toward the closest person in a chain. This is unlike an obstacle avoidance behavior where the intent is to go away from the person aiming for an overtaking. The angle is given by Eqs. (11)–(12)

$$\theta_{i,1} = atan2\left(S^i_{ahead,1}\left(t\right).y - S_i\left(t\right).y, S^i_{ahead,1}\left(t\right).x - S_i\left(t\right).x\right) - \theta_i \tag{11}$$

$$\theta_{i,2} = atan2\left(S^i_{ahead,2}\left(t\right).y - S_i\left(t\right).y, S^i_{ahead,2}\left(t\right).x - S_i\left(t\right).x\right) - \theta_i \tag{12}$$

Here, θ_i is the current orientation of the person calculated from a derivative of the trajectory. The angle computed is modified to lie within the range $(-\pi, \pi]$.

Accomplishing the task of future trajectory prediction of a person requires the direction of the goal. The first person in the queue only moves toward the goal, while the others tend to align toward the goal in case the people ahead highly deviate. The angle to goal is given by Eq. (13)

$$\theta_{i,goal} = atan2\left(G_y - S_i\left(t\right).y, G_x - S_i\left(t\right).x\right) - \theta_i \tag{13}$$

https://doi.org/10.1017/S0263574721000679 Published online by Cambridge University Press

While understanding the motion pattern of any situation, the velocity (linear and angular) is one of the most important pieces of information. Here, the velocity of the closest two people ahead is calculated in each frame and fed into the model so that it can learn the trajectory. The velocity of the person ahead directs the ideal speed that the person should keep. A similar reasoning can also be given for the angular speed.

The target output is the linear and angular speed of the person being planned. The outputs are supplied to the simulated person, who implements the linear and angular velocities, which make it move suitably in the simulation setup.

For any person i in the simulation system, the inputs are outputs of SVR are given as follows:

Inputs:

- Distance to closest person and second closest person ahead, $d\left(S_i, S^i_{ahead,1}\right)$ and $d\left(S_i, S^i_{ahead,2}\right)$,
- Angle to closest person and second closest person ahead ($\theta_{i,1}$ and $\theta_{i,2}$),
- Linear speed of closest person and second closest person ahead ($v_{i,1}$ and $v_{i,2}$),
- Angular speed of closest person and second closest person ahead ($\omega_{i,1}$ and $\omega_{i,2}$),
- Angle to goal ($\theta_{i,goal}$).

Outputs:

- Linear speed of the person (v_i) and
- angular speed of the person (ω_i).

5.2. Support vector regression

Support Vector Machine (SVM) was designed as a pattern recognition tool and later, a few functionalities were added such as the ability to solve regression problems, building a multidimensional strip, and solving a linear equation. Considering its importance in statistical learning, SVMs were made more generic for statistical learning. When SVMs solve pattern recognition problems, it can also find decision rules in feature space with great generalization. It can find optimal separating hyperplane, by using a subset of training data, referred to as *Support Vectors (SVs)*. Increasing the range between classes and training data can be an alternative approach for optimizing the cost function, such as mean square error and it provides useful features to SVMs so that make it can automatically tune the classification function, using a small subset of data representation for classification. Therefore, SVMs able to summarize the information contained in the dataset.

Support Vector Regression (SVR) takes multiple inputs and produces single output that is, many to one system. It can support several inputs as a feature vector and output a single target value calculated by the decision function. Training algorithm for SVR requires f_{motion} motion features as a parameter of objective function, which is shown by Eq. (14)

$$y = O\left(f_{motion}\right) = < w, f_{motion} > + b \tag{14}$$

The task of the learning algorithm is to find the value weights w and bias b, which represents the optimal separating hyperplane. SVR also provides an alternative loss function which includes distance measure, known as $\in -$insensitive loss function. It is a penalty tool used to define the range between the actual value and predicted value. If $|y - O\left(f_{motion}\right)| > \vartheta$ then a penalty will be added, which is shown by Eq. (15). ϑ parameter can affect the smoothness of the response and the number of support vectors.

$$L_{\vartheta}\left(y, O\left(f_{motion}\right)\right) = \begin{cases} 0 & |y - O\left(f_{motion}\right)| \leq \vartheta \\ |y - O\left(f_{motion}\right)| - \vartheta & otherwise \end{cases} \tag{15}$$

Here, SVR is used to model the human walking behavior in a natural scenario so that robot can use it for motion planning. As a feature, several walking characteristics are computed from the person's trajectory, which is used to understand the motion pattern.

https://doi.org/10.1017/S0263574721000679 Published online by Cambridge University Press

Once the SVR learns these features, it is used to predict the future trajectory of all persons. Hence, a robot can use this model for its own path planning. After predicting the person's future trajectory, a robot can calculate the optimal and collision-free paths.

The learned model is then used for the simulation of the behavior. The initial state is loaded from the observed natural trajectories of the humans. The features are extracted from the trajectory and given to the SVR for the prediction of the future state. The future state is again used to compute the new features, which guide the next motion of the humans. The SVR hence forms a reactive controller that can be used to estimate the human motion in unorganized chains starting from any hypothetical situation.

Most of the recent works in the literature use deep learning approaches. Deep learning can perform this task, but we know that if we want to perform some task by deep learning, then training the deep learning model requires a huge amount of data. In our model, the data are observed from real-life trajectories of humans and the data are extremely limited. Further, it is not possible to collect a lot of data from different scenarios. Therefore, generalizing ability needs to be high. This is only possible when well-crafted features are used that can generalize to new scenarios, without requiring large training data. Unlike many approaches in the literature, we do not resort to using simulation systems for the creation of a large amount of training data that can be non-socialistic. Therefore, in our approach, we tracked the people using a Kalman filter followed by preparing a trajectory database, and after this few important features are calculated separately which are used by the SVR.

6. Results

In this paper, we analyze human motion in a natural environment and extract the individual trajectory, which is used for the training of our model. Here, we collected the data using a Quanergy's M8 outdoor 3D LiDAR scanner for data collection, and it has a range capacity of 200 m, 5–20 Hz frame rate, 360° and 20° horizontal, vertical field of view respectively, range of angular resolution is $0.03° - 0.2°$.

First, we had placed this sensor in a big classroom where people move randomly or any specific pattern. We collected the data after that the trajectory of each person was extracted. But when these trajectories were analyzed, then it was found that these trajectories are free from natural characteristics, which are crucial for understanding human walking behavior in the real-world scenario. The subjects largely moved as instructed. That is why the next time we placed it in an open space in our campus, where moving people were not aware of this data collection, they all moved in their own way and self-decided where to go.

While collecting data, an interesting behavior was obtained as an *unorganized chain* of moving people. This situation occurs when the lectures get over or begin. Then all students are found coming in or out. All of them maintain distance from each other. None of the students was intimated or briefed about the recording so as not to bias the behavior. The lidar scanner was placed at a distance, where unorganized chaining often happens due to a large number of students attempting to leave the building as quickly as possible after the classes. The density typically varies from sparing initially and at the end, to a high density in the middle phases, as the students leave. A challenge was that most places where such unorganized chaining happens are narrow and congested and placing a lidar in the vicinity is not possible. Further, many sites have mostly occluded data and visibility is primarily at the exit point alone, where the behavior analysis is not possible. The identification of the site was based upon the metrics of a large field of view, low occlusion, and getting enough visibility to track and study the behavior.

The detection algorithm was applied to the recorded data for finding out the human in the 3D point cloud. Once the humans were detected in each frame, then the tracker was applied for tracking all persons and also for extracting the trajectories of all persons. Now, these trajectories were used by our model for learning to perform this some important features was extracted from the trajectory.

Once the model gets trained, then it was used for the person's future trajectory prediction. We compare our approach with a widely used approach, the artificial potential field. The potential field is a very natural and intuitive way to model the problem. The algorithm has a fewer number of parameters that

https://doi.org/10.1017/S0263574721000679 Published online by Cambridge University Press

can be set to replicate a dataset based on a few iterations of trial and error. The parameters are themselves intuitive to set. Hence, the approach has a large generalization ability. However, the method may not represent all possible interactions, which have to be modeled as forces.

6.1. Trajectory prediction

The prediction of the trajectory is shown in Fig. 4. The simulation was done based on the initial settings taken from the recorded data. For each person, three different trajectories appear. The first trajectory is of the color blue and denotes the ground truth. The second one is the trajectory predicted by the proposed approach. It is shown by the green color. The last trajectory is represented by the red color, which is produced by the artificial potential field method. The direction of motion in the simulation is denoted by an arrow. From the simulation, it is observed that the predicted trajectory by our approach is much closer than the artificial potential field method, and the shape of taking the turn also similar to the actual trajectory. These simulations give an idea about the environment and direction of the persons in the future.

In scenario 2, the average distance error is maximum when compared to the other scenarios because the number of persons in this chaining is the maximum. When there are multiple people spread in a characteristic manner, humans must choose primarily which human they are following. In an organized chaining behavior, one only follows the immediate leader. In this unorganized chaining behavior, we enabled the model to consider two people ahead. In scenario 2; however, there was enough gap that exposed three people ahead. The person used that gap and displayed an aggressive behavior of following a distant person ahead, while the SVR did not see that person as it is restricted to two people only. For such exceptional cases, considering three people ahead will solve the problem, which will however affect the generalizability. It must be stressed that such gaps are rarely seen in unorganized chains in a small corridor.

The artificial potential field method is based on a goal. All people using the potential field algorithm try to go straight to the goal and feel repulsion from each other which avoids collisions. The approach hence cannot model the behaviors where a follower sees a leader (or a few leaders) as the goal, along with the ultimate goal.

6.2. Performance

Using the proposed method, the future location of the person is predicted using support vector regression. Its performance is measured by comparing the predicted location with the ground truth. The error between the actual and the predicted trajectory for each scenario is used as a performance metric. We have the predictions for the person i as $\tau_{i,predicted} = \left[\tau_{i,predicted}\ (s)\right]$ and actual route available as a ground truth as $\tau_{i,actual} = \left[\tau_{i,actual}\ (s)\right]$.The distance error between them is measured as given by Eq. (16)

$$d\left(\tau_{i,predicted}, \tau_{i,actual}\right) = \frac{1}{len\left(\tau_{i,predicted}\right)} \sum_{s=1}^{len\left(\tau_{i,predicted}\right)} d\left(\tau_{i,predicted}\ (s), \tau_{i,actual}\ (s)\right) \tag{16}$$

Prediction accuracy is another performance metric. It is the ratio of correctly predicting the location to the total number of predictions in each trajectory. Prediction accuracy for trajectories $\tau_{i,predicted}$ and $\tau_{i,actual}$ is defined as Eq. (17).

$$\chi\left(\tau_{i,predicted}, \tau_{i,actual}\right) = \frac{1}{len\left(\tau_{i,predicted}\right)} \sum_{s=1}^{len\left(\tau_{i,predicted}\right)} E\left(\tau_{i,predicted}\ (s), \tau_{i,actual}\ (s)\right) \tag{17}$$

$E(p_i, g_i)$ is used to match the predicted location with the actual one given by Eq. (18)

$$E\left(\tau_{i,predicted}\ (s), \tau_{i,actual}\ (s)\right) = \begin{cases} 1 & if\ \left(\tau_{i,predicted}\ (s), \tau_{i,actual}\ (s)\right) < \epsilon \\ 0 & otherwise \end{cases} \tag{18}$$

https://doi.org/10.1017/S0263574721000679 Published online by Cambridge University Press

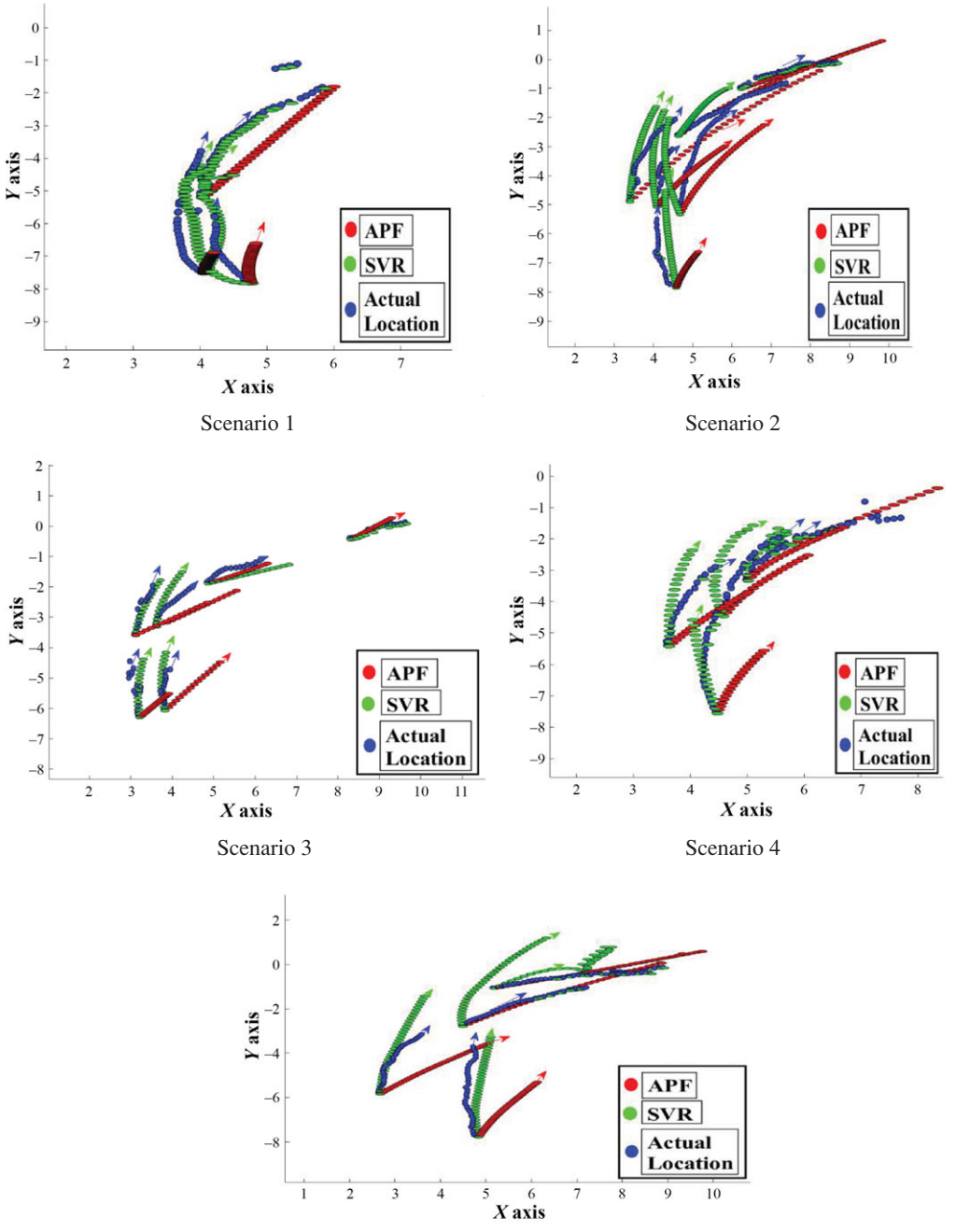

Figure 4. *Simulation of predicted future trajectory of unorganized chaining behavior.*

Our method of future trajectory prediction is compared with another approach that is, artificial potential field method and it is observed that our approach gives a natural trajectory in which natural characteristics of human walking also exist. To compare this approach, average distance error and prediction accuracy is computed for both methods and the outcome of this comparison reveals that our approach

https://doi.org/10.1017/S0263574721000679 Published online by Cambridge University Press

Table II. *Comparison of the results.*

Scenario	Avg distance error (Proposed)	Avg distance error (APF)	Prediction accuracy (Proposed)	Prediction accuracy (APF)
1	**0.33**	0.60	**0.96**	0.27
2	**0.54**	2.45	**0.84**	0.19
3	**0.25**	0.61	**0.94**	0.37
4	**0.25**	0.47	**0.923**	0.62
5	**0.44**	0.48	**0.88**	0.69

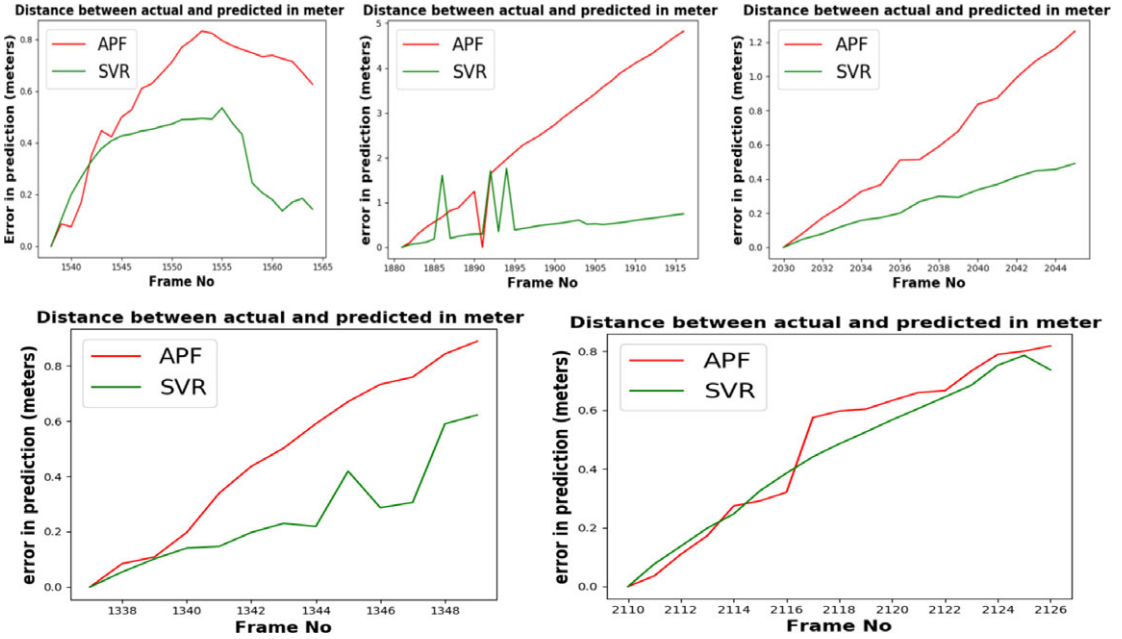

Figure 5. *Comparing average distance error in all scenarios.*

is better than the artificial potential field due to a less average distance error and a higher prediction accuracy. The results are shown in Table II.

Trajectory predicted by both the methods is also simulated and it is found that the trajectory of the proposed method is similar to ground truth when compared to the artificial potential field method. Comparison and performance of both methods can be understood by a graph between time and distance error with the actual location of all scenarios, as shown in Fig. 5. In the graph, red represents the artificial potential field approach and the green line denotes the proposed approach. For each time frame, the distance error is plotted and from the graph, one can derive that the performance of the proposed approach is closer to the ground truth and less error than the artificial potential field approach.

For completeness of the study, a macroscopic variable of density is also analyzed, which is most interesting. Here, we considered a point, where a large number of people move. We counted the number of persons who are passing per second from this point on different distance ranges. Figure 6 shows the average number of persons in different distance ranges (in meters) from the exit point. The crowd density is lowest at the exit as the outflow rate is very high while the inflow rate is limited. The crowd density is highest at the end, where queuing starts. The intermediate density is nearly constant.

https://doi.org/10.1017/S0263574721000679 Published online by Cambridge University Press

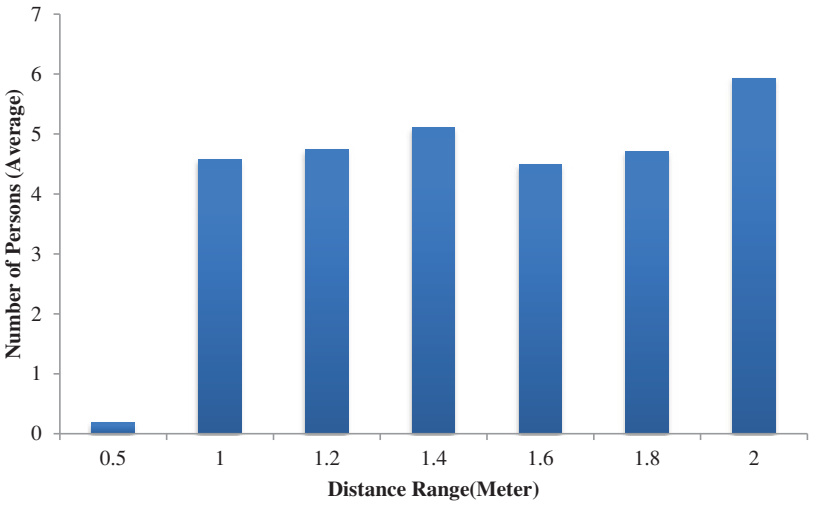

Figure 6. *Crowd density per second of our datasets.*

7. Conclusion

The unorganized chaining behavior is a dominant behavior widely found especially because of situations like corridors, where the difference between the rates of entry and exit can cause queuing, while there may be no physical infrastructure to maintain a strict queue. Humans have a peculiar behavior of slowly following the people ahead and moving toward a common goal. The behavior has not been appreciably studied in the literature and this paper made efforts in the same direction.

We considered dense human crowds and use a 3D LiDAR for capturing the movement of the humans. Here, a moving people detection algorithm in point clouds is applied. A Kalman filter is used to perform tracking, which also solves the problem of occlusion and correspondence and generates the trajectory database. The trajectories were used to learn a motion model that imitates the behavior of the people when displaying the unorganized chaining behavior.

We compared the performance with the artificial potential field method and our approach provided better results when compared to the potential field. The learning algorithm aims to model the behavior without requiring a large amount of training data and not overfitting the data. The features were carefully chosen to meet the same requirements. In contrast to the potential field, the algorithm was able to model latent intents not modeled by the potential field, while generalizing to the testing sequences.

There are some limitations of the paper that need to be addressed in the future. The paper analyzes a single behavior only and the work needs to be extended to all behaviors that near exhaustively cover all perspectives of human motion. The cases where multiple behaviors interact are of special interest. Similarly, it is important to cover different diverse scenarios, where unorganized chaining is found to exhaustively learn the model. Similarly, it needs to be seen if different personal attributes and social groups affect the behavior of the human, in which case the same may have to be included in the modeling. Finally, the paper aims to make a robot that navigates socially. The translation to a real robot needs to be done in the future. The model may need adaptation as the robots may have different constraints and social acceptance in contrast to humans.

Acknowledgments. This work is supported by the Indian Institute of Information Technology, Allahabad through the seed money project initiative.

https://doi.org/10.1017/S0263574721000679 Published online by Cambridge University Press

References

[1] L. Guo, S. Manglani, Y. Liu and Y. Jia, "Automatic sensor correction of autonomous vehicles by human-vehicle teaching-and-learning," *IEEE Trans. Veh. Technol.* **67**(9), 8085–8099 (2018).

[2] C. Hu, R. Wang, F. Yan and N. Chen, "Output constraint control on path following of four-wheel independently actuated autonomous ground vehicles," *IEEE Trans. Veh. Technol.* **65**(6), 4033–4043 (2015).

[3] J. Broch, D. A. Maltz, D. B. Johnson, Y. C. Hu and J. Jetcheva, "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols," *Proceedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking* (1998) pp. 85–97.

[4] J. Ariyakhajorn, P. Wannawilai and C. Sathitwiriyawong, "A Comparative Study of Random Waypoint and Gauss-Markov Mobility Models in the Performance Evaluation of Manet," *2006 International Symposium on Communications and Information Technologies* (IEEE, 2006) pp. 894–899.

[5] K. Lee, S. Hong, S. J. Kim, I. Rhee and S. Chong, "SLAW: Self-similar least-action human walk," *IEEE/ACM Trans. Networking* **20**(2), 515–529 (2011).

[6] D. Z. Wang and I. Posner, "Voting for Voting in Online Point Cloud Object Detection," **In**: *Robotics: Science and System,* vol. 1(3) (2015) pp. 10–15607.

[7] M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong and I. Posner, "Vote3deep: Fast Object Detection in 3D Point Clouds Using Efficient Convolutional Neural Networks," *2017 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2017) pp. 1355–1361.

[8] S. Song and J. Xiao, "Sliding Shapes for 3D Object Detection in Depth Images," *European Conference on Computer Vision* (2014) pp. 634–651.

[9] S. Song and J. Xiao, "Deep Sliding Shapes for Amodal 3D Object Detection in RGB-D Images," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016) pp. 808–816.

[10] B. Li, "3D Fully Convolutional Network for Vehicle Detection in Point Cloud," *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2017) pp. 1513–1518.

[11] B. Li, T. Zhang and T. Xia, "Vehicle detection from 3D lidar using fully convolutional network," *RSS* (2016).

[12] J. Ku, M. Mozifian, J. Lee, A. Harakeh and S. L. Waslander, "Joint 3D Proposal Generation and Object Detection from View Aggregation," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2018) pp. 1–8.

[13] X. Chen, H. Ma, J. Wan, B. Li and T. Xia, "Multi-View 3D Object Detection Network for Autonomous Driving," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017) pp. 1907–1915.

[14] K. Li, X. Wang, Y. Xu and J. Wang, "Density enhancement-based long-range pedestrian detection using 3D range data," *IEEE Trans. Intell. Transp. Syst.* **17**(5), 1368–1380 (2015).

[15] A. Börcs, B. Nagy and C. Benedek, "Instant object detection in lidar point clouds," *IEEE Geosci. Remote Sens. Lett.* **14**(7), 992–996 (2017).

[16] Y. Choi, N. Kim, S. Hwang, K. Park, J. S. Yoon, K. An and I. S. Kweon, "KAIST multi-spectral day/night data set for autonomous and assisted driving," *IEEE Trans. Intell. Transp. Syst.* **19**(3), 934–948 (2018).

[17] Z. Yan, T. Duckett and N. Bellotto, "Online learning for 3D LiDAR-based human detection: Experimental analysis of point cloud clustering and classification methods," *Auto. Rob.* **44**(2), 147–164 (2020).

[18] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang and O. Beijbom, "Pointpillars: Fast Encoders for Object Detection from Point Clouds," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2019) pp. 12697–12705.

[19] C. R. Qi, W. Liu, C. Wu, H. Su and L. J. Guibas, "Frustum Pointnets for 3D Object Detection from RGB-D Data," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018) pp. 918–927.

[20] R. E. Kalman, "A new approach to linear filtering and prediction problems," *J. Basic Eng.* **82**(1), 35–45 (1960).

[21] Y. Bar-Shalom, *Tracking and Data Association* (Academic Press Professional Inc., San Diego, CA, USA, 1987).

[22] S. Rezatofighi Hamid, A. Milan, Z. Zhang, Q. Shi, A. Dick and I. Reid, "Joint Probabilistic Data Association Revisited," *Proceedings of the IEEE International Conference on Computer Vision* (2015) pp. 3047–3055.

[23] A. Ess, B. Leibe and L. Van Gool, "Depth and Appearance for Mobile Scene Analysis," *2007 IEEE 11th International Conference on Computer Vision* (IEEE, 2007) pp. 1–8.

[24] M. Bajracharya, B. Moghaddam, A. Howard, S. Brennan and L. H. Matthies, "A fast stereo-based system for detecting and tracking pedestrians from a moving vehicle," *Int. J. Rob. Res.* **28**(11–12), 1466–1485 (2009).

[25] D. M. Gavrila and S. Munder, "Multi-cue pedestrian detection and tracking from a moving vehicle," *Int. J. Comput. Vis.* **73**(1), 41–59 (2007).

[26] J. Dequaire, P. Ondrúška, D. Rao, D. Wang and I. Posner, "Deep tracking in the wild: End-to-end tracking using recurrent neural networks," *Int. J. Rob. Res.* **37**(4–5), 492–512 (2018).

[27] P. Ondruska and I. Posner, "Deep Tracking: Seeing Beyond Seeing Using Recurrent Neural Networks," *Thirtieth AAAI Conference on Artificial Intelligence* (2016).

[28] S. Gao, Z. Han, C. Li, Q. Ye and J. Jiao, "Real-time multipedestrian tracking in traffic scenes via an RGB-D-based layered graph model," *IEEE Trans. Intell. Transp. Syst.* **16**(5), 2814–2825 (2015).

[29] D. Held, J. Levinson, S. Thrun and S. Savarese, "Robust real-time tracking combining 3D shape, color, and motion," *Int. J. Rob. Res.* **35**(1–3), 30–49 (2016).

https://doi.org/10.1017/S0263574721000679 Published online by Cambridge University Press

[30] K. H. Lee and J. N. Hwang, "On-road pedestrian tracking across multiple driving recorders," *IEEE Trans. Multimedia* **17**(9), 1429–1438 (2015).
[31] M. Ozuysal, V. Lepetit and P. Fua, "Pose Estimation for Category Specific Multiview Object Localization," *2009 IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2009) pp. 778–785.
[32] A. Ghodrati, M. Pedersoli and T. Tuytelaars, "Is 2D Information Enough for Viewpoint Estimation?," *Proceedings BMVC 2014* (2014) pp. 1–12.
[33] D. Teney and J. Piater, "Multiview feature distributions for object detection and continuous pose estimation," *Comput. Vis. Image Understanding* **125**(1), 265–282 (2014).
[34] K. Hara and R. Chellappa, "Growing Regression Forests by Classification: Applications to Object Pose Estimation," *European Conference on Computer Vision* (2014) pp. 552–567.
[35] S. Danielsson, L. Petersson and A. Eidehall, "Monte Carlo Based Threat Assessment: Analysis and Improvements," *2007 IEEE Intelligent Vehicles Symposium* (IEEE, 2007) pp. 233–238.
[36] J. Ariyakhajorn, P. Wannawilai and C. Sathitwiriyawong, "A Comparative Study of Random Waypoint and Gauss-Markov Mobility Models in the Performance Evaluation of Manet," *2006 International Symposium on Communications and Information Technologies* (IEEE, 2006) pp. 894–899.
[37] S. Qiao, D. Shen, X. Wang, N. Han and W. Zhu, "A self-adaptive parameter selection trajectory prediction approach via hidden Markov models," *IEEE Trans. Intell. Transp. Syst.* **16**(1), 284–296 (2014).
[38] Q. Lv, Y. Qiao, N. Ansari, J. Liu and J. Yang, "Big data driven hidden Markov model based individual mobility prediction at points of interest," *IEEE Trans. Veh. Technol.* **66**(6), 5204–5216 (2016).
[39] C. Wang, L. Ma, R. Li, T. S. Durrani and H. Zhang, "Exploring trajectory prediction through machine learning methods," *IEEE Access* **7**(1), 101441–101452 (2019).
[40] L. Wang, L. Zhang and Z. Yi, "Trajectory predictor by using recurrent neural networks in visual tracking," *IEEE Trans. Cybern.* **47**(10), 3172–3183 (2017).
[41] W. Hu, D. Xie, Z. Fu, W. Zeng and S. Maybank, "Semantic-based surveillance video retrieval," *IEEE Trans. Image Process.* **16**(4), 1168–1181 (2007).
[42] B. T. Morris and M. M. Trivedi, "Trajectory learning for activity understanding: Unsupervised, multilevel, and long-term adaptive approach," *IEEE Trans. Pattern Anal. Mach. Intell.* **33**(11), 2287–2301 (2011).
[43] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei and S. Savarese, "Social LSTM: Human Trajectory Prediction in Crowded Spaces," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016) pp. 961–971.
[44] M. Luber, J. A. Stork, G. D. Tipaldi and K. O. Arras, "People Tracking with Human Motion Predictions from Social Forces," *2010 IEEE International Conference on Robotics and Automation* (IEEE, 2010) pp. 464–469.
[45] L. Leal-Taixé, M. Fenzi, A. Kuznetsova, B. Rosenhahn and S. Savarese, "Learning an Image-Based Motion Context for Multiple People Tracking," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2014) pp. 3542–3549.
[46] W. Choi and S. Savarese, "A Unified Framework for Multi-Target Tracking and Collective Activity Recognition," *European Conference on Computer Vision* (2012) pp. 215–230.
[47] W. Choi and S. Savarese, "Understanding collective activities of people from videos," *IEEE Trans. Pattern Anal. Mach. Intell.* **36**(6), 1242–1257 (2014).
[48] T. Kruse, A. K. Pandey, R. Alami and A. Kirsch, "Human-aware robot navigation: A survey," *Rob. Auto. Syst.* **61**(12), 1726–1743 (2013).
[49] J. Rios-Martinez, A. Spalanzani and C. Laugier, "From proxemics theory to socially-aware navigation: A survey," *Int. J. Soc. Rob.* **7**(2), 137–153 (2015).
[50] C. P. Lam, C. T. Chou, K. H. Chiang and L. C. Fu, "Human-centered robot navigation—towards a harmoniously human–robot coexisting environment," *IEEE Trans. Rob.* **27**(1), 99–112 (2010).
[51] H. Kretzschmar, M. Spies, C. Sprunk and W. Burgard, "Socially compliant mobile robot navigation via inverse reinforcement learning," *Int. J. Rob. Res.* **35**(11), 1289–1307 (2016).
[52] X. T. Truong and T. D. Ngo, "Toward socially aware robot navigation in dynamic and crowded environments: A proactive social motion model," *IEEE Trans. Autom. Sci. Eng.* **14**(4), 1743–1760 (2017).
[53] S. S. Paliwal and R. Kala, "Maximum clearance rapid motion planning algorithm," *Robotica* **36**(6), 882–903 (2018).
[54] V. Malviya, A. K. Reddy and R. Kala, "Autonomous social robot navigation using a behavioral finite state social machine," *Robotica*, **38**(12), 2266–2289 (2020).
[55] A. Sinha and P. Papadakis, "Mind the gap: Detection and traversability analysis of terrain gaps using LIDAR for safe robot navigation," *Robotica* **31**(7), 1085–1101 (2013).

Cite this article: A. Malviya and R. Kala (2022). "Learning-based simulation and modeling of unorganized chaining behavior using data generated from 3D human motion tracking", Robotica **40**, 544–569. https://doi.org/10.1017/S0263574721000679