Predicting Pedestrian Trajectories using Velocity-Space Reasoning

Sujeong Kim, Stephen J. Guy, Wenxi Liu, Rynson W. H. Lau, Ming C. Lin and Dinesh Manocha

Abstract We introduce a novel method to predict pedestrian trajectories using agent-based velocity-space reasoning for improved human-robot interaction. This formulation models the trajectory of each moving pedestrian in a robot's environment using velocity obstacles and learns the simulation parameters based on tracked data. The resulting motion model for each agent is computed using statistical inferencing techniques from noisy data. This includes the combination of Ensemble Kalman filters and maximum likelihood estimation algorithm to learn individual motion parameters at interactive rates. We highlight the performance of our motion model in real-world crowded scenarios. We also compare its performance with prior techniques and demonstrate improved accuracy in the predicted trajectories.

1 Introduction

Robots are increasingly used in everyday life. As more robots are introduced into human surroundings, it becomes imperative to develop reliable and safe human-robot interaction. The robots should be able to successfully navigate in a dynamic environment with multiple moving humans to reach their goal positions. In order to develop such systems, the robots should have a good understanding of its environment and the ability to sense, track, and predict the position of all moving humans in its workspace to perform collision-free navigation.

Sensing and tracking moving humans is a topic that has been studied in both robotics and computer vision, e.g. [18, 25, 15]. A key part of these methods is often to use a prior motion model fitted for the scenario in question. However, these

Sujeong Kim, Stephen J. Guy, Ming C. Lin and Dinesh Manocha
University of North Carolina at Chapel Hill, e-mail: {sujeong, sjguy, lin, dm}@cs.unc.edu

Wenxi Liu and Rynson W. H. Lau
City University of Hong Kong, e-mail: wenxiliu2@student.cityu.edu.hk, rynson.lau@cityu.edu.hk)

motion priors are usually generalized motions and not specific to individual's movement, therefore they do not perform well in tracking unusual (abnormal) behaviors or in capturing person-to-person differences. These behaviors, such as some people moving against the flow or quick movements to avoid collisions with other people, are common even in moderately crowded scenarios.

In this work, we introduce a novel motion model built on agent-based, velocity-space reasoning, combined with Bayesian statistical inference to provide a per-person motion model for each individual in a robot's environment. Unlike previous methods that use general motion priors or simple motion models, our method is able to reproduce an individual's motions with its own characteristics, also capable of dynamically adjusting the motion model for each individual in the presence of sensor noise and model uncertainty.

More specifically our approach works as follows. We assume at any given time, a robot has past observations on each person (represented as an agent) in the environment and wants to make some predictions about motion for each agent in the next timestep. We apply Ensemble Kalman Filtering (EnKF) to estimate the parameters for the human motion model based on *reciprocal velocity obstacle* (RVO) [2, 29] in a crowded scene that optimally matches the data observed thus far. Then, using these RVO parameters adaptively computed, we can simulate the crowd motion to estimate the future trajectory of all individuals in the scene and estimate factors, such as their current goal. We demonstrate that our per-agent learning method generates more accurate motion predictions than previous models, especially for complex scenarios, such as those with occlusions, low-resolution sensors, and missing data.

The rest of our paper is organized as follows. Section 2 reviews related work. Section 3 gives a brief overview of the RVO-based, per-agent motion model for a crowd. Section 4 describes our approach on incorporating an adaptive machine-learning framework with RVO-based multi-agent simulation, and Section 5 presents some results on observed (video recorded) data.

2 Related Work

In this section, we give an overview of prior work on motion models in robotics and crowd simulation and their application to pedestrian or people tracking.

2.1 Motion Models

There is an extensive body of work in robotics, multi-agent systems, crowd simulation, and computer vision on modeling pedestrians' motion in crowded environments. Here we provide a brief review of some recent work in modeling pedestrians' motion. In the fields of pedestrian dynamics and crowd simulation, many models have been proposed [26, 21]. These works could be broadly classified into a few main categories: potential-based works which models virtual agents in crowd as particles with potentials and forces [12], boid-like approaches which create simple rules to steer agents [24], geometric model which computes collision-free velocity[3, 29], and field-based works which generate fields based on continuum theory or fluid models [28]. Among these works, velocity-based motion models [13, 14, 3, 29, 23] have been successfully applied in the analysis and simulation of crowd behaviors, and been shown to have efficient implementations [11].

2.2 People Tracking with Motion Model

Most related works in tracking people include [8, 27, 7] that make simple assumptions on the pedestrian motion model, in order to improve the tracking quality. In recent years, better pedestrian motion models have been deployed in robotics and computer vision literature for tracking people. For example, long-term motion planning models have been proposed to combine with tracking. Bruce, et al. [5] and Gong et al. [9] estimate people's destinations and in turn predict the pedestrian motion along the path from the current position towards the estimated goal position. Ziebart et al. [31] considers conditioned action as well as the trajectory of people for the prediction. Liao et al. [17] extracts a Voronoi graph from environment and predicts people's motion along the edges following the topological shape of the environment. In contrast to these methods, people tracking techniques using short-term motion models have also been studied as well. Bennewitz et al. [1] apply Expectation Maximization clustering to learn typical motion patterns from trajectories of people and employs Hidden Markov Models to predict the motion of people for a mobile robot. Luber et al. [18] apply Helbing's social force model to track people based on a Kalman-filter based tracker. Mehran et al. [20] also apply social force model to detect people's abnormal behaviors from video. Pellegrini et al. [22] use an energy function to build up a goal-directed short-term collision avoidance motion model and Linear Trajectory Avoidance to improve their people tracking quality from video. [29].

3 Background and Overview

In this section, we briefly discuss the crowd simulation method we use, and provide overview of our method for training this motion model based on observations. We first provide some definitions of important terms used in this section.

A *pedestrian* is a human entity that shares an environment with the robot. We treat pedestrians as autonomous entities that seek to avoid collisions with each other. We denote n as the number of pedestrians in the environment. We assume a robot's *sensor* produces a noisy estimate of the position of pedestrians. Furthermore, we assume the amount of sensor noise is known. We also assume the robot has an estimate of the environment represented as a series of connected line segments.

3.1 RVO Crowd Simulations

As part of our approach, we need to use an underlying crowd simulation technique to model individual goals and the interactions between people. For this model, we choose a velocity-space reasoning technique based on Reciprocal Velocity Obstacles (RVO) [29]. RVO-based collision avoidance has previously been shown to reproduce important pedestrian behaviors such as lane formation, speed-density dependencies, and variations in human motion styles [11, 10].

Our implementation of RVO is based on the publicly available RVO2-Library (http://gamma.cs.unc.edu/RVO2). This library implements an efficient variation of RVO that uses a set of linear collision avoidance constraints on an agents velocities known as *Optimal Reciprocal Collision Avoidance* (ORCA) constraints [29]. Each agent is assumed to have a position, velocity, and a preferred velocity. If an agent's preferred velocity is forbidden by the ORCA constraints, that agent choses the closest velocity which is not forbidden. Formally:

$$\mathbf{v}^{new} = \underset{\mathbf{v} \notin OCRA}{\operatorname{argmin}} \|\mathbf{v} - \mathbf{v}_{pref}\|. \tag{1}$$

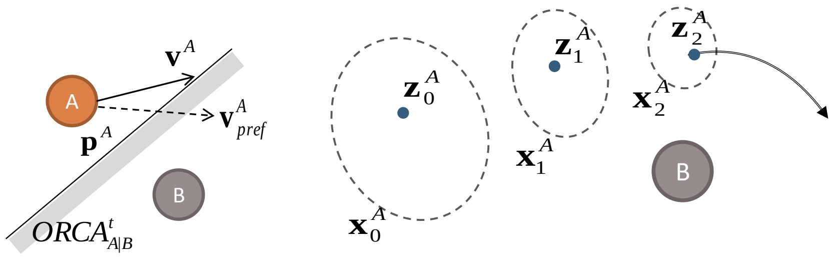
This process is illustrated in Fig. 1a.

An agents position is updated by integration of the new velocity computed in Eqn. 1. An agent's preferred velocity is assumed to change slowly, and is modeled by a small amount of noise each timestep. More details of the mathematical formulation is given in Sec 4. A through derivation of the ORCA constraints are given in [29].

We use the RVO2-Library to represent the state of each sensed pedestrian in a robot's environment. The state of pedestrian at timestep k is represented as a 6D vector $\mathbf{x}_k$ which consists of an agent's 2D position, 2D velocity, and 2D preferred velocity as defined by RVO2-Library. Our human motion model, called *Bayesian-RVO* or *BRVO*, seeks to adaptively find the RVO state that best represents all the previously observed motion data (with sensing uncertainty) for each pedestrian in the robot's environment.

3.2 Problem Definition

As discussed above, we define the computation of the BRVO motion model as a state estimation problem. Formally, the problem we seek to solve is as follows. Given a set of observations, denoted $\mathbf{z}_0 \cdots \mathbf{z}_t$, for each pedestrian, what is the RVO state $\mathbf{x}_t$ that best reproduces the motion seen so far. Given this estimate we predict the future



(a) RVO State ($\mathbf{p}$, $\mathbf{v}$, $\mathbf{v}_{pref}$) (b) Adaptive Refinement and Prediction

Fig. 1: (a) Overview of RVO Simulation illustrating an agent's position ($\mathbf{p}$), preferred velocity ($\mathbf{v}_{pref}$) and actual velocity ($\mathbf{v}$). If the ORCA collision-avoidance constraints prevent an agent from taking their preferred velocity, as shown here, the actual velocity will be the closest allowable velocity. These elements combine to form the RVO state $\mathbf{x}$. (b) As new data is observed (blue dot) BRVO refines its estimate of a distribution of likely values of the RVO states (dashed ellipse). These parameters are then used with RVO to predict likely paths (as indicated by arrow).

motion of each agent (person) by using the RVO simulation model to determine the likely future path of each pedestrian.

We propose an iterative solution to this problem. We assume a robot working under a sense-plan-act loop. During the sensing step, the robot measures new (noisy) positions of each pedestrian, updates its estimate of each person's state, and creates a new plan taking into account the expected motion of nearby pedestrians.

4 Bayesian-RVO

In this section, we provide the mathematical formulation of our Bayesian-RVO motion model, or **BRVO**. This model combines an Ensemble Kalman Filtering approach with the Expectation-Maximization algorithm to estimate the best approximated RVO state of each agent, as well as the uncertainty in the model.

4.1 Model Overview

Our model performs Bayesian learning for each agent. We assume each agent can be represented with their own RVO state vector $\mathbf{x}$. Given an agent's RVO state $\mathbf{x}_k$ at timestep k, we use the RVO collision-avoidance motion model, denoted here as

f, to predict the agent's next state $\mathbf{x}_{k+1}$. We denote the error f has in predicting the RVO state at each timestep as $\mathbf{q}$. This leads to our motion model of:

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k) + \mathbf{q} \tag{2}$$

Additionally, we assume the sensing of the robot can be represented by a function h that projects the predicted state $\mathbf{x}_k$ to an observed state which we will denote as $\mathbf{z}_k$. In general, sensing devices produce noisy readings leading to an error between the observed state and the ground truth, we denote this error as $\mathbf{r}$. This assumption leads to our sensing model as:

$$\mathbf{z}_k = h(\mathbf{x}_k) + \mathbf{r} \tag{3}$$

An overview of this adaptive motion prediction model is given in Figure 2.

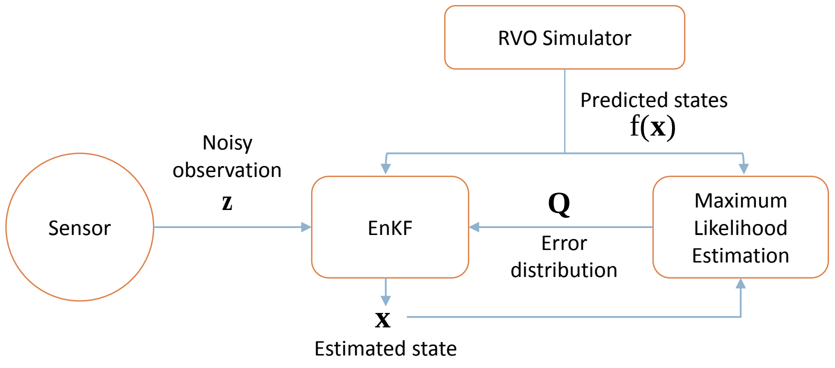


Fig. 2: **Overview of the Adaptive Motion Model.** We estimate current state $\mathbf{x}$ via an iterative process. Given noisy data observed by the sensor, RVO as a motion model, and the error distribution matrix $\mathbf{Q}$, we estimate current state. The error distribution matrix $\mathbf{Q}$ is recomputed based on the difference between the observed state and the prediction $f(\mathbf{x})$ and used to refine current estimation of $\mathbf{x}$.

Simplifying Assumptions The model given by Eqns. 2 and 3 is very general. In order to efficiently estimate the state $\mathbf{x}_k$ from the observations $\mathbf{z}_k$ we must make some simplifying assumptions, which allow us to develop a suitable learning approach for our adaptive model. First we assume that the error terms q and r are independent at each timestep, and follow a zero-meaned Gaussian distribution with covariances Q and R respectively. That is:

$$\mathbf{q} \sim N(0,Q) \tag{4}$$

$$\mathbf{r} \sim N(0,R) \tag{5}$$

We make a further assumption that the sensor error r is known or can be well estimated. This is typically possible by making repeated measurements of known data points to establish an average error. In many cases, this error will be provided by the manufacture of the sensing device. This error will fully specify the matrix R.

To summarize, the h function is specified by the robot's sensors, and the matrix R characterizes the estimated accuracy of these sensors. The f function is a motion model will be used to predict the motion of each agent and Q is the accuracy of this model.

Our BRVO framework uses the RVO-based simulation model to represent the function f and Ensemble Kalman Filtering to estimate the simulation parameters which best fit the observed data. In addition, we adapt the EM-algorithm to estimate the model error Q for each agent. Better estimating Q improves the Kalman Filtering process, which in turn improves the predictions given by BRVO. This process is used iteratively to predict the next state and refine the state estimation for each agent, as depicted in Fig 1b. More specifically, we perform EnKF and EM step for each agent, separately, but taking account all the agents in the motion model f(x). It gives more flexibility in cases like dynamically changing scenes, such as agents entering and leaving the scene in the middle of the sequences, because the computations are done with fixed size per-agent matrix.

4.2 State Estimation

Optimal estimation of the simulation state $\mathbf{x}_k$ is possible when f and h are linear functions by using Kalman Filtering. However, our BRVO approach uses RVO for the simulation model f and allows an arbitrary sensing model h which creates a non-linear system with no known method of finding an optimal estimate.

The BRVO motion model uses an extension to Kalman Filtering known as Ensemble Kalman Filter (EnKF) as a model for state estimation. The motivation for this algorithmic choice is two-fold. First, EnKF makes the same assumptions we laid forth in Sec 4.1. That is, a (potentially) non-linear f and h combined with a Gaussian representation of error. Secondly, as compared to methods such as particle filters, EnKF is very computationally efficient, providing more accuracy for a given number of samples. This is an important consideration for low-to-medium power onboard computers commonly found on a mobile robot.

At a high level, EnKF works by representing the potential state of an agent at each timestep as an ensemble or collection of discrete samples. Each sample is updated according to the motion model f. A mean and standard deviation of the samples is computed at every timestep for use in the predictor-correction equations. For a more detailed explanation of EnKF, we refer readers to a standard textbook in statistical inference such as [6].

State Representation We represent the state of each agent as the set of RVO parameters as discussed Sec 3.1. Therefore the state of an agent, $\mathbf{x}$, is six dimensional:

$$\mathbf{x} = \begin{bmatrix} \mathbf{p} \\ \mathbf{v} \\ \mathbf{v}_{pref} \end{bmatrix}, \tag{6}$$

where $\mathbf{p}$ is the agent's position, $\mathbf{v}$ the velocity, and $\mathbf{v}_{pref}$ the preferred velocity. To summarize the crowd dynamics model f:

$$f(\begin{bmatrix} \mathbf{p} \\ \mathbf{v} \\ \mathbf{v}_{pref} \end{bmatrix}) = \begin{bmatrix} \mathbf{p} + \mathbf{v}\Delta t \\ \operatorname{argmin}_{\mathbf{v} \in OCRA} \|\mathbf{v} - \mathbf{v}_{pref}\| \\ \mathbf{v}_{pref} \end{bmatrix}. \tag{7}$$

For the experiments in this paper, we assume that the robot has the ability to sense the relative positions of pedestrians. This assumption leads to a simple h function of

$$h(\begin{bmatrix} \mathbf{p} \\ \mathbf{v} \\ \mathbf{v}_{pref} \end{bmatrix}) = \mathbf{p}. \tag{8}$$

In general, our BRVO framework makes no assumption about the linearity of the sensing model. More complex sensing functions can be represented (for example, integrating multiple sensors) by modifying the function h in accordance with the new sensing model.

4.3 Maximum Likelihood Estimation

The accuracy of the states estimated by the EnKF algorithm is a function of the parameters defined in Eqns 2-5: f, Q, h, and R. While the sensing function h and the error distribution R are determined by the sensor's specifications, f is determined by the motion model chosen. However, the choice of motion prediction error distribution Q is still a free parameter. We propose a method of estimating the optimal value for Q based on the Expectation Maximization or EM-algorithm [19].

The EM-algorithm is a generic framework for learning (locally) optimal parameters by following a gradient decent approach. The algorithm alternates between two steps: an E-step which computes expected values and an M-step which computes the parameters which maximize the likelihood of the values computed during the E-step.

In our case, the E-step is exactly the EnKF algorithm. This step estimates the most likely state given the parameters Q. For the M-step, we need to compute the Q which maximizes the likelihood of values estimated from EnKF. This probability will be maximized with a Q that best matches the observed error between the predicted state and the estimated state. We can compute this value simply by finding the average error for each sample in the ensemble at each timestep for each agent.

By iteratively performing the E and M steps we will continuously improve our estimate of Q which will in turn improve the quality of the learning and the predictiveness of the method. In theory, one should iterate over the E and M steps until convergence. In practice, the process converges fairly rapidly. Due to the online process nature of our approach, we limit the number of iterations to three, which we found to be empirically sufficient. We analyze the resulting improvement produced by the EM algorithm in Section 5.1.

4.4 Implementation

A pseudo code for our BRVO algorithm is given in Algorithm 2. We represent the distribution of likely RVO states as an ensemble of m samples. We set $m = 1000$, for the results shown in section 5. Lines 2 through 6 preform a stochastic prediction of the likely next state. Lines 7 through 10 correct this predicted value based on the observed data from the robot's sensor. Lines 11 through 14 preform a maximum likelihood estimation of the uncertainly in the prediction.

Algorithm 1: Bayesian-RVO

```
Input: Observed positions over time $\mathbf{z}_1 ... \mathbf{z}_t$, crowd motion simulator $f$, estimated initial
    error variance $\mathbf{Q}$, sensor function $h$, sensor noise $\mathbf{R}$
, and the number of samples $m$. Output: Estimated agent's state distributions $\mathbf{x}_1 ... \mathbf{x}_t$    1
foreach $k \in 1 \dots t$ do    2
    // EnKF Predict Step
    foreach $i \in 1 \dots m$ do    3
        Draw $\mathbf{q}_{k-1}^{(i)}$ from $\mathbf{Q}$, $\hat{\mathbf{x}}_k^{(i)} = f(\hat{\mathbf{x}}_{k-1}^{(i)}) + \mathbf{q}_{k-1}^{(i)}$ ;    4
        Draw $\mathbf{r}_k^{(i)}$ from $\mathbf{R}$, $\hat{\mathbf{z}}_k^{(i)} = h(\hat{\mathbf{x}}_k^{(i)}) + \mathbf{r}_k^{(i)}$;    5
    $\bar{\mathbf{z}}_k = \frac{1}{m} \sum_{i=1}^m \hat{\mathbf{z}}_k^{(i)}$;    6
    $Z_k = \frac{1}{m} \sum_{i=1}^m (\hat{\mathbf{z}}_k^{(i)} - \bar{\mathbf{z}}_k)(\hat{\mathbf{z}}_k^{(i)} - \bar{\mathbf{z}}_k)^T$;    7
    // EnKF Correct Step
    $\bar{\mathbf{x}}_k = \frac{1}{m} \sum_{i=1}^m \hat{\mathbf{x}}_k^{(i)}$;    8
    $\Sigma_k = \frac{1}{m} \sum_{i=1}^m (\hat{\mathbf{x}}_k^{(i)} - \bar{\mathbf{x}}_k)(\hat{\mathbf{x}}_k^{(i)} - \bar{\mathbf{x}}_k)^T$;    9
    foreach $i \in 1 \dots m$ do    10
        $\hat{\mathbf{x}}_k^{(i)} = \hat{\mathbf{x}}_k^{(i)} + \Sigma_k Z_k^{-1} (\mathbf{z}_k - \hat{\mathbf{z}}_k^{(i)})$;    11
    // Maximum Likelihood Estimation
    $\mathbf{Q}_{k-1} = \mathbf{Q}$;    12
    foreach $i \in 1 \dots m$ do    13
        $\mathbf{Q}_k += (\hat{\mathbf{x}}_k^{(i)} - f(\hat{\mathbf{x}}_{k-1}^{(i)}))(\hat{\mathbf{x}}_k^{(i)} - f(\hat{\mathbf{x}}_{k-1}^{(i)}))^T$;    14
    $\mathbf{Q} = \frac{k-1}{k} \mathbf{Q}_{k-1} + \frac{1}{k} \mathbf{Q}_k$    15
```

5 Results

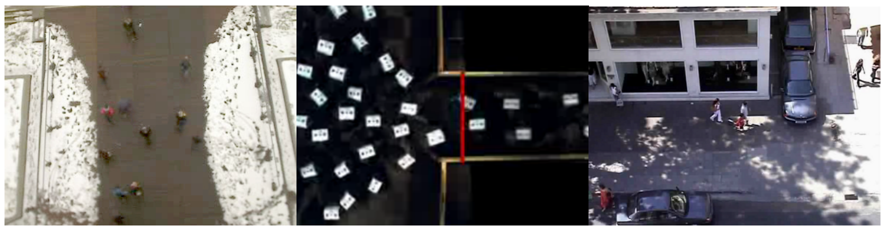
In this section, we show some comparisons and results that show the advantage of our BRVO model.

We analyze various attributes of our model across a verity of datasets. First, we isolate the effect of the EM loop for estimating the prediction uncertainty **Q**. Next, we demonstrate the ability of our approach to cope with sensor noise. Additionally, we analyze how varying density and varying sampling rates each affect the results. Finally, we provide a quantitative comparison to two recent techniques.

We focus our analysis on three different datasets, illustrated in Fig. 3.
Campus Video data of students on campus recorded from the top of the ETH main building in Zurich was extracted by manual notation every 0.4 second [22]. We extract three sequences from this data, each containing about 10 seconds of pedestrian interaction: Campus-1 (7 pedestrians), Campus-2 (18 pedestrians), Campus-3 (11 pedestrians).

Bottleneck Optical tracking equipment capture the motion of participant in a lab-environment [4]. Participants moved through a bottleneck opening into a narrow passage. Data was collect with a passage width of 1.0 meter and 2.5 meter, denoted Bottleneck-100 and Bottleneck-250 respectively. Both studies have about 170 participants, and we use the data subsampled at a rate of 0.4 second.

Street This video is recorded from a street view, of low density pedestrian traffic, with manual motion tracking [16]. The dataset contains motion of 148 pedestrians over a period of roughly 5 minutes.



(a) Campus (b) Bottleneck (c) Street

Fig. 3: **Benchmarks used for our experiments** (a) In the Campus dataset, a sparse set of students walk pass a fixed camera. (b) In the Bottleneck dataset, multiple cameras track participants walking into a narrow hallway. (c) In the Street dataset, a fixed camera tracks pedestrian motion on a street.

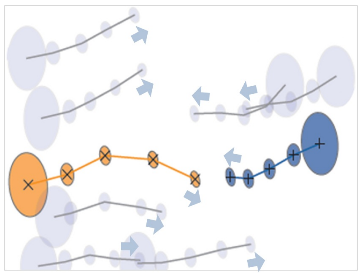
We include comparison with Constant Velocity and Constant Acceleration models as they are simple to implement, and provide a common benchmark to recent approaches with similar comparisons. For constant motion models, we used the last observed velocity and acceleration, respectively, for prediction. In other words, states are updated using the last observed velocity for Constant Velocity model, and the last observed acceleration for Constant Acceleration model.

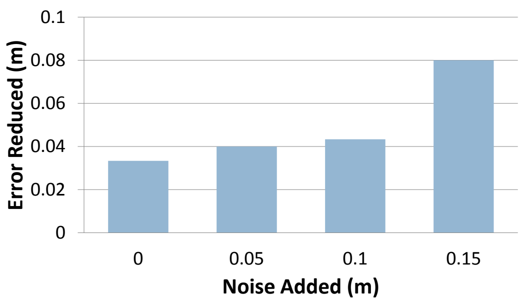
5.1 Learning maximum likelihood (EM)

We show that our method can refine itself by using the EM algorithm. As discussed in Section 4.3, EM step reduces the difference between the observed state and the prediction based on our motion model. Figure 4a visually illustrate the effect of the EM algorithm; as new data is presented and pedestrians interact with each other, the estimated uncertainty, **Q**, decreases. Without EM step, the same, initially given estimated uncertainty is used as **Q** without refining its values.

We can analyze the effect of estimating **Q** (as discussed in Section 4.3) by removing this step from our approach. We compare the performance of BRVO on the Campus sequences with and without this EM feedback loop.

Similar to the above experiments, BRVO learns for the first 5 frames, and predicts position of later frames. We test the improvement on varying level of noise. The same initial estimated error Q is used for all cases. We use Gaussian noise with standard deviation 0.05m, 0.1m, 0.15m to the ground truth data. Figure 4b show the measured improvement in term of reduced prediction error. As the chart shows, using the EM loop to estimate the uncertainty leads to more accurate predictions. We also observe that the improvement increases under larger amounts of noise.


(a) Estimation Refinement



(b) Improvement from EM feedback Loop

Fig. 4: **The effect of EM algorithm.** (a) This figure shows the trajectory of each agent and the estimated error distribution (ellipse) for the first five frames of Campus-3 data. The estimated error distributions gradually reduces as agents interact. (b) The improvement provided by the EM feedback loop for various amounts of noise. As the noise increases, this feedback becomes more important.

5.2 Noisy Data

Sensor noise is an important concern in many robotic systems. To analyze how BRVO responds to noise in the data, we compare its performance across varying levels of synthetic noise. Again, BRVO learns for first 5 frames, and predicts positions on later frames. Figure 5 shows average prediction error across all the Campus sequences for BRVO, Constant Acceleration, and Constant Velocity models. In all cases, adding noise increases the prediction error. However, BRVO is minimally imprecated compared to other methods.

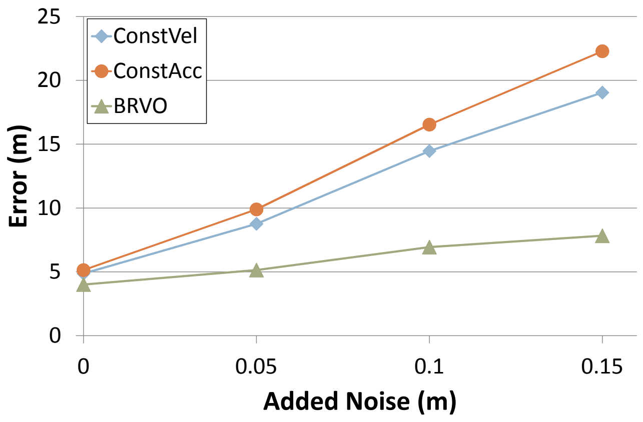
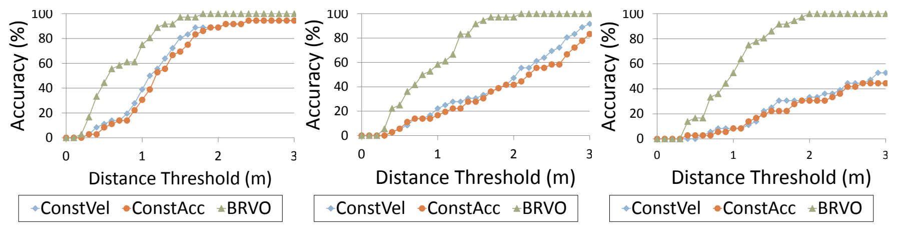


Fig. 5: **Mean prediction error (lower is better).** Prediction error after 7 frames (2.8s) on Campus dataset. As compared to constant velocity and constant acceleration models, BRVO can better cope with varying amount of noises in the data.

Figure 6 shows the percentage of correctly predicted path within varying accuracy threshold. At an accuracy threshold of 0.5m, BRVO far outperforms ContAcc and ConstVel models (44% vs 8% and 11% respectively) even with little noise. With larger amounts of noise, these difference is magnified further.

5.3 Varying Density Scenario

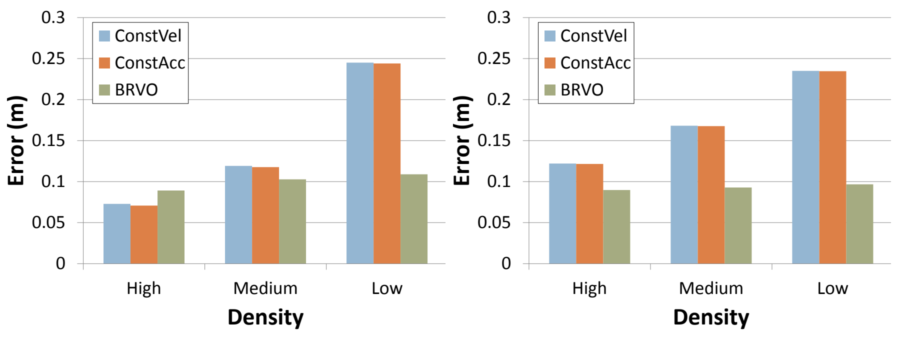
We use the Bottleneck scenarios to analyze the effect of varying densities. This scenario shows a variety of densities: before the passage, there is a high density of crowd from people backing up and waiting, as people pass through the entrance, the density drops to a more medium level, only a few people at a time enter the hallway resulting in a low density. We compute the error of BRVO for each of these regions of the scenario, for both the 1m and 2.5m hallway.



(a) 0.05m (b) 0.1m (c) 0.15m

Fig. 6: **Prediction Accuracy (higher is better)** (a-c) Shows prediction accuracy across various accuracy thresholds. The analysis is repeated at three noise levels. For all accuracy thresholds, for all noise levels BRVO produces more accurate predictions than constant velocity of acceleration models. The advantage is most significant for large amounts of noise in the sensor data as in (c).

Figure 7 compares our error to Constant Velocity and Acceleration models. Both Constant Velocity and Constant Acceleration model have large variations in error for different density, but BRVO performs about equally well across all the densities as it dynamically adapts the parameters each frame.



(a) Bottleneck-100 (b) Bottleneck-250

Fig. 7: **Error at Various Densities (lower is better).** In high-density scenarios there is little motion and simple models such as constant velocity perform about as well as BRVO. However, in low and medium densities where there is more motion BRVO provides more accurate motion prediction than the other models. In general, BRVO performs consistently well across various densities.

5.4 Varying Sampling Rates

Our method also works well with vey large timesteps. To demonstate this aspect, we show the results on the Street dataset with varying sampling intervals to sub-sample the data. We chose Street scenario,the highest framerate at 0.04s per frame. Figure 8 shows the graph of the mean error verses sampling interval. From the result, we can see that our method performs very well compared to Constant Velocity and Constant Acceleration model in every sampling interval.

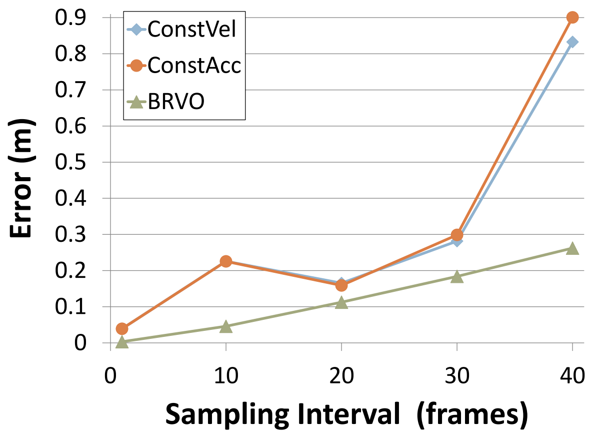


Fig. 8: **Error vs Sampling Interval** As the sampling interval increases the error of Constant Velocity and Constant Acceleration estimations grows much larger than that of BRVO.

6 Conclusion

We have presented a method to predict pedestrian trajectories using agent-based velocity-space reasoning. The BRVO model we introduced is an online motion prediction method, which learns per-agent parameters even without prior knowledge about the environment. We demonstrated the ability of this approach to perform well on several different datasets, with varying amounts of sensor noise, interaction levels, and density. Specifically, we showed our approach performs very well with noisy data and low framerate scenarios. We also showed that we can handle dynamic scenarios with temporal and spatial variance in density and speed. BRVO assumes no prior knowledge of the scenario to which the model is applied to. As such, it is well suited for a mobile robot that may frequently encounter new obstacles.

In the future, we would like to integrate our BRVO motion model for online/offline tracking. The improved motion model should increase tracking performance in typical environments but also in challenging scenarios, such as occluded scenes, low-resolution input, and data with missing frames, where the predictions from BRVO can help cope with these challenging issues. Additionally, we would like to incorporate our motion prediction model with an online planner for robots navigating in crowded scenes.

Acknowledgements

We are grateful to the reviewers for their comments, we would like to thank Jia Pan and David Wilkie for their help. This research is supported in part by ARO Contract W911NF-04-1-0088, NSF awards 0917040, 0904990, 100057 and 1117127, and Intel.

References

1. Bennewitz, M., Burgard, W., Cielniak, G., Thrun, S.: Learning motion patterns of people for compliant robot motion. The International Journal of Robotics Research **24**(1), 31–48 (2005)
2. Van den Berg, J., Lin, M., Manocha, D.: Reciprocal velocity obstacles for real-time multi-agent navigation. In: Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on, pp. 1928–1935. IEEE (2008)
3. van den Berg, J., Patil, S., Sewall, J., Manocha, D., Lin, M.: Interactive navigation of individual agents in crowded environments. In: Symp. on Interactive 3D Graphics and Games (I3D 2008) (2008)
4. Boltes, M., Seyfried, A., Steffen, B., Schadschneider, A.: Automatic extraction of pedestrian trajectories from video recordings. Pedestrian and Evacuation Dynamics 2008 pp. 43–54 (2010)
5. Bruce, A., Gordon, G.: Better motion prediction for people-tracking (2004)
6. Casella, G., Berger, R.: Statistical inference. Duxbury advanced series in statistics and decision sciences. Thomson Learning (2002)
7. Cui, J., Zha, H., Zhao, H., Shibasaki, R.: Tracking multiple people using laser and vision (2005)
8. Fod, A., Howard, A., Mataric, M.: A laser-based people tracker (2002)
9. Gong, H., Sim, J., Likhachev, M., Shi, J.: Multi-hypothesis motion planning for visual object tracking
10. Guy, S., Kim, S., Lin, M., Manocha, D.: Simulating heterogeneous crowd behaviors using personality trait theory. In: Symp. on Computer Animation, p. 4352 (2011)
11. Guy, S.J., Chhugani, J., Curtis, S., Dubey, P., Lin, M., Manocha, D.: PLEdestrians: a least-effort approach to crowd simulation. In: Symp. on Computer Animation, p. 119128 (2010)
12. Helbing, D., Molnar, P.: Social force model for pedestrian dynamics. Physical review E **51**(5), 4282 (1995)
13. Karamouzas, I., Heil, P., van Beek, P., Overmars, M.: A predictive collision avoidance model for pedestrian simulation. Motion in Games pp. 41–52 (2009)
14. Karamouzas, I., Overmars, M.: A velocity-based approach for simulating human collision avoidance (2010)

15. Kratz, L., Nishino, K.: Tracking pedestrians using local spatio-temporal motion patterns in extremely crowded scenes. Pattern Analysis and Machine Intelligence, IEEE Transactions on (99), 11 (2011)
16. Lerner, A., Fitusi, E., Chrysanthou, Y., Cohen-Or, D.: Fitting behaviors to pedestrian simulations. In: Symp. on Computer Animation, p. 199208 (2009)
17. Liao, L., Fox, D., Hightower, J., Kautz, H., Schulz, D.: Voronoi tracking: Location estimation using sparse and noisy sensor data (2003)
18. Luber, M., Stork, J., Tipaldi, G., Arras, K.: People tracking with human motion predictions from social forces. In: Robotics and Automation (ICRA), 2010 IEEE International Conference on, p. 464469 (2010)
19. McLachlan, G.J., Krishnan, T.: The EM Algorithm and Extensions (Wiley Series in Probability and Statistics), 2 edn. Wiley-Interscience (2008). URL http://www.worldcat.org/isbn/0471201707
20. Mehran, R., Oyama, A., Shah, M.: Abnormal crowd behavior detection using social force model. In: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, p. 935942 (2009)
21. Pelechano, N., Allbeck, J., Badler, N.: Virtual crowds: Methods, simulation, and control. Synthesis Lectures on Computer Graphics and Animation **3**(1), 1176 (2008)
22. Pellegrini, S., Ess, A., Schindler, K., Van Gool, L.: You'll never walk alone: Modeling social behavior for multi-target tracking. In: Computer Vision, 2009 IEEE 12th International Conference on, p. 261268 (2009)
23. Pettr, J., Ondej, J., Olivier, A.H., Cretual, A., Donikian, S.: Experiment-based modeling, simulation and validation of interactions between virtual walkers. In: Symp. on Computer Animation, p. 189198 (2009)
24. Reynolds, C.W.: Steering behaviors for autonomous characters. In: Game Developers Conference. http://www. red3d. com/cwr/steer/gdc99 (1999)
25. Rodriguez, M., Ali, S., Kanade, T.: Tracking in unstructured crowded scenes. In: Computer Vision, 2009 IEEE 12th International Conference on, p. 13891396 (2009)
26. Schadschneider, A., Klingsch, W., Klüpfel, H., Kretz, T., Rogsch, C., Seyfried, A.: Evacuation dynamics: Empirical results, modeling and applications. Arxiv preprint arXiv:0802.1620 (2008)
27. Schulz, D., Burgard, W., Fox, D., Cremers, A.: People tracking with mobile robots using sample-based joint probabilistic data association filters. The International Journal of Robotics Research **22**(2), 99 (2003)
28. Treuille, A., Cooper, S., Popovi, Z.: Continuum crowds. In: ACM SIGGRAPH 2006 Papers, p. 11601168 (2006)
29. Van Den Berg, J., Guy, S., Lin, M., Manocha, D.: Reciprocal n-body collision avoidance. Robotics Research p. 319 (2011)
30. Yamaguchi, K., Berg, A., Ortiz, L., Berg, T.: Who are you with and where are you going? In: Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, pp. 1345 –1352 (2011). DOI 10.1109/CVPR.2011.5995468
31. Ziebart, B.D., Ratliff, N., Gallagher, G., Mertz, C., Peterson, K., Bagnell, J.A., Hebert, M., Dey, A.K., Srinivasa, S.: Planning-based prediction for pedestrians. In: Proceedings of the 2009 IEEE/RSJ international conference on Intelligent robots and systems, IROS'09, pp. 3931–3936. IEEE Press, Piscataway, NJ, USA (2009). URL http://dl.acm.org/citation.cfm?id=1732643.1732694