[[1]](#footnote-1)

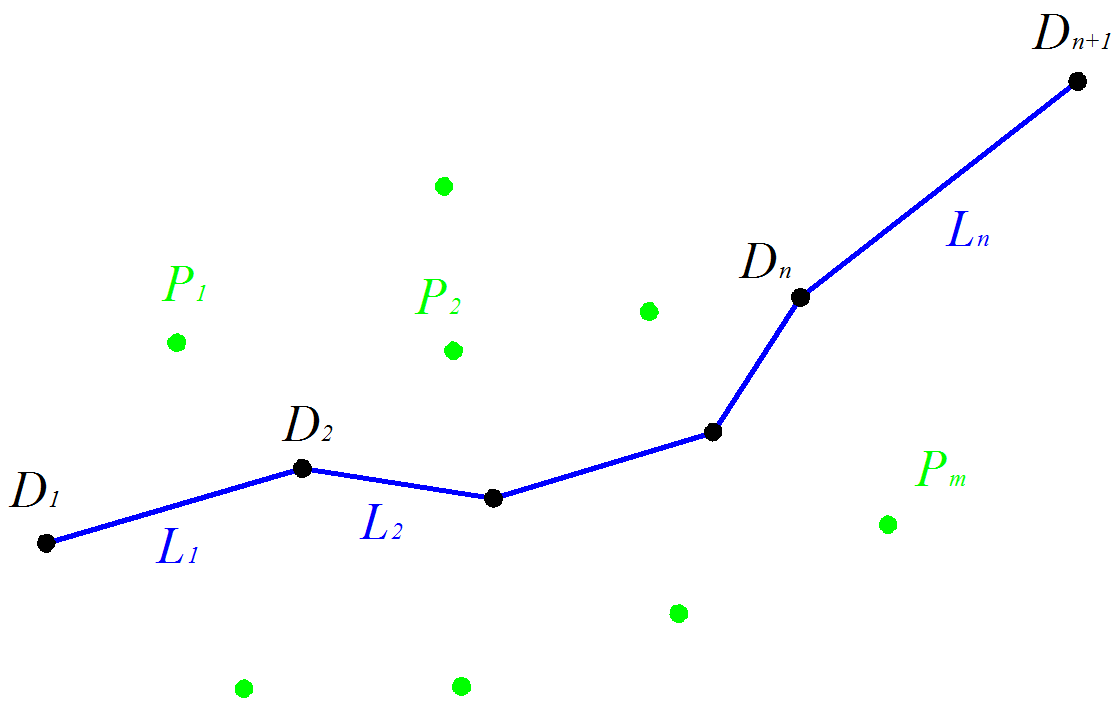
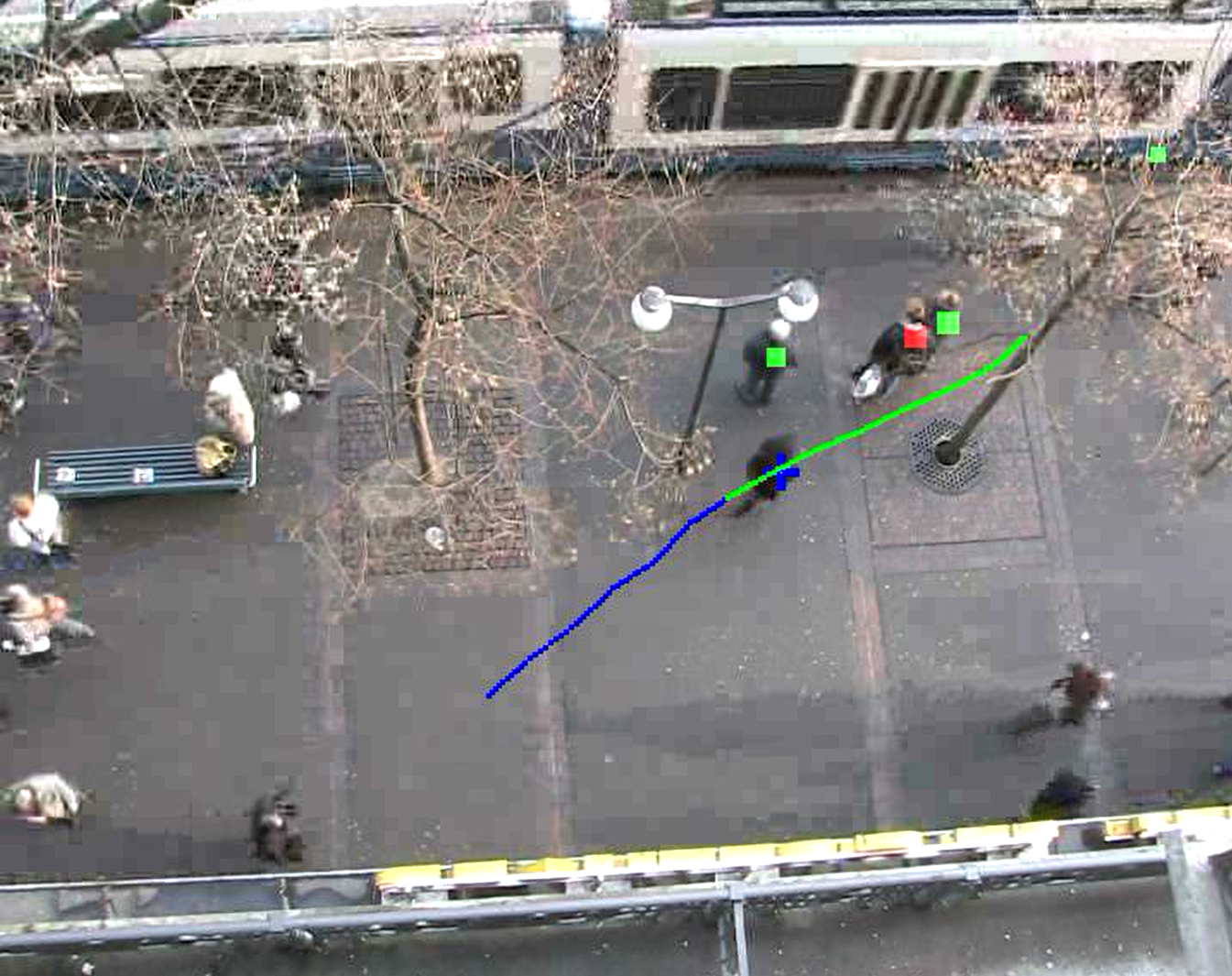


Figure 1. In a low-feature environment, a pedestrian’s turning is usually caused by other pedestrians (the person with red square in the left figure). The right figure shows this problem abstractly, Black dots *Di* are inflection points of focus agent and green dots *Pi* are other pedestrians when *D2* happens.

**Explaining Pedestrian Intention from Behavior**

Zhi Qiao and Ross A. Knepper, Member, IEEE

*Abstract*—Efficient and competent robot navigation among human pedestrians requires prediction of the intentions of other agents (humans or robots). Those intentions are unobservable, but under the rational actor assumption, a pedestrian’s behavior provides clues to their intentions. Those clues in turn permit an observer to infer circumstances that are known or observable by the pedestrian but are unknown to the observer, such as the existence of another pedestrian approaching from around a corner. In this paper, we focus on the question of identifying the cause of observed changes in trajectory. We construct a data-driven model to infer a distribution over possible causes of observed bends. We identify each possible influencer in the dataset using a 2-D kernel density estimator. With this model, we can calculate the relative influence likelihood of other pedestrians toward the target agent. We train and evaluate our model on recorded pedestrian datasets. Finally, we present two sample applications of this model to predict a pedestrian’s goal and infer features out of the observer’s view.

# INTRODUCTION

Social navigation can be characterized as the interplay of two objectives: reaching one’s goal and maintaining socially competent behavior. Competent pedestrian navigation requires robots to reason about human intentions [cite Mavrogiannis], which can be characterized as the pair of an eventual destination and a plan for getting there. Teleological reasoning [cite Csibra] describes the ability of a human observer to infer the intentions of another human based on the choice of actions they perform. Conversely, when the intention is known, humans anticipate a *predictable* action that minimizes energy [cite something] – a straight line path.

As the density of pedestrians in a space increases, the social competency objective increasingly conflicts with goal attainment. To reconcile the conflict, humans plan more complex paths comprising sequences of subgoals. When behaving predictably then, people follow a sequence of straight lines connecting subgoals. During social navigation in a pedestrian context, people employ teleological reasoning based on the observed sequence of motions in order to infer one another’s intentions. This inference, in turn, allows people to anticipate the motions of others and select compatible motions in advance.

In this paper, we address the problem of identifying subgoals of another agent based on observation of their path in context. We contribute:

* a data-driven, probabilistic, causative model for attributing bends in a pedestrian’s path,
* a predictive model that identifies obstacles or agents in the environment responsible for bends,
* a method of predicting where and when the causative agent is obscured from view, and
* results evaluating the predictive power of our approach.

# Related Work

The social force model proposed by D Helbing and P Molnar is an early and classical method to take pedestrian sociality into account [1]. That method builds a forcing relation between pedestrians, and others also improve it by many ways. For example, Y Tamura assumes a goal in the front of pedestrian and combine social force model to predict possible trajectory of pedestrian [2]. However, the methods derived from social force model are usually partial for pedestrian behavior, and they ignore some valuable information showed by trajectory.

For conscious pedestrians, their historical trajectories and velocities are important information and we can infer many reliable results from historical trajectories, even predict future trajectory. In an indoor environment, B Ziebart and his colleagues collect pedestrians’ trajectories over a period of time and analyze them with machine learning [3]. The method can obtain a cost-map about current environment, and infer possible future trajectory from historical trajectory with Markov decision process. But this method is limited in common environment because the learning in a special indoor environment will be influenced by environment feature. Besides, S Chung proposes a method to collect trajectories in various conditions with pedestrian ego-graph (PEG) [4]. They build a framework to predict pedestrian’s future trajectory by general spatial effects (GSEs) and discover specific spatial effects (SSEs) by abnormal trajectories.

The problem of two methods above is analyzing the whole trajectories to infer potential information. So we divide the trajectory to different parts and try to explain the reason that change of trajectory. There is an interesting question: Why pedestrians change their trajectories rather than walking as a straight line in a wide and low-features environment? Most pedestrians will to choose walk as a straight line in an open space, because we always choose the shortest path to get our destination. But in a crowded environment, pedestrians will change their trajectories because of interaction with others. In most cases, there are two explanations for this, pedestrians change their trajectories to avoid someone in front of them (such as the left of Fig. 1) or return to their original trajectories after avoiding a pedestrian. Certainly, sometimes pedestrians will take some small actions for other reason, such as keeping safe distance from others or walking along environment feature. And some pedestrians just appear incompetent while walking.

# Problem Statement

While focus agent *F* is walking in an open space, the trajectory can be divided for a few parts *Li = {L1, L2, … Ln}* as the changes of its mind. Then we can get a few point *Di = {D1, D2, … Dn+1}* for division at different time *Ti = {T1, T2, … Tn+1}* (right in Fig. 1). The first problem is how to divide the trajectory properly. There are various methods for trajectory fitting, C Lee presented a derivation for a spline smoother which considers local velocity information [5]. But fitting trajectory with straight line is more suitable when we take mind into account, because it’s easy to observe the changes of mind with straight line.

Meanwhile, there are other pedestrians *Pi = {P1, P2, … Pm}*, the position of pedestrian *Pi* at time *T2* is *Wi*. In a low-feature environment, pedestrians’ turnings usually cause by other pedestrians. So the other question here is which pedestrian is influencing focus agent’s trajectory when the first turning happens. The research about relation between will

# Method

The reason we can’t just regard pedestrians as moving obstacles is that pedestrians have their own destinations to take uncertain trajectories. Their trajectories have a few of inflection points to divide trajectories to different parts. There is an obvious question here: why pedestrians change their original trajectories instead of keeping them? We think the change of pedestrian’s trajectory arise from exist of general spatial effects (moving pedestrians or fixed obstacles) exclude the environment characteristic. So the question is simplified to confirm which pedestrian is effect of focus agent in a no-feature environment.

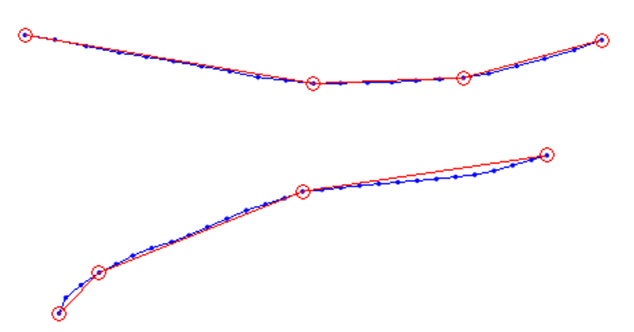


Figure 2. Two fitting-lines of pedestrians’ trajectories. Blue dots are tracking points and red circles are inflection points calculated by dividing method. When the trajectory has a part with small radian, this method will divide it inaccurately.

We labeled the effects of focus agents by watching the video with two rules. These wandering agents, staying agents, incompetent agents and groups aren’t considered by us. After the transformation, we obtain a distribution of influencers’ point in the view of focus agent by 2-D KDE. The two dimensions are bearing and range. By this model, we make a video to show the likelihood of effect with another dataset. In the video, blue cross is the focus agent, red square means highest likelihood effect and green squares mean other effects (left in Fig. 1). The length of square means the relative likelihood whether this agent is influencing the trajectory of focus agent. We can see the reason clearly why pedestrians yield while walking by these squares.

## A. Trajectory Dividing

Because the trajectories in dataset are represented by discrete points, we can’t confirm where pedestrians change their trajectories exactly. So fitting trajectory to straight line is necessary to check pedestrians’ intention. The tracking frames of the dataset we use is 2.5, we make straight line every three points to get rid of subtle angle differences between adjacent points and get a vector *A={A1,A2,…Am}* made by the angle between the line and *X* axis. All the angle difference *Vi,j*between every two elements *{Ai, Aj}* (*i*, *j* < m) in vector *A* form a matrix *U*. By this matrix, we can divide the trajectory to broken lines easily.

If the trajectory needed to divide has only one infection point, finding the single maximal absolute value in *U* is enough to solve it. It means the maximal angle difference between every two lines, so the two lines can be taken to fit trajectory as two broken lines. If the problem is fitting trajectory with two or more infection points, finding all the local maximal (minimal) values in the matrix *U* is necessary for robustness. Every row or line of local maximal (minimal) value in the *U* is a potential infection point. But sometimes this method will obtain incorrect infection points because of subtle noise in dataset, there is another scenario to filter incorrect infection points.

After obtaining all the infection points, we put the absolute angles (at absolute coordinate system) of these lines into a vector *S={S1,S2,… Sn} (𝑆∈𝐴)* and set a threshold *Q* = 6º with experience. Filtering vector S according to threshold will build a new vector *G (G∈S)*, where every two adjacent points’ angle difference less than *Q*. Finally *G* is the reference vector to fit pedestrians’ trajectories.

This approach solves the trajectory’s fitting-line problem successfully if the trajectory behaves like a few of broken lines (the upper result in Fig. 2). As a global fitting method, it won’t be influenced by subtle noise and we can adjust fitting degree by changing the value of threshold Q. But if the trajectory behaves like an arc with small curvature, the approach won’t work well because it’s difficult to fit arc with a few lines (the lower result in Fig. 2).

## B. Labeling

The primary work of labeling is to observe the reason that why pedestrians change their trajectories. For every pedestrian, it maybe cause by a special pedestrian, environment feature or self-distraction. If there’s a special pedestrian influences the trajectory of focus agent, record the number of the pedestrian and take him as effect.

The hardest part of labeling process is how to ensure the objectivity and uniformity of this process. If the turning is cause by pedestrian, two special situations happen mostly: a. pedestrian turns when he find another pedestrian who’s in front of him (avoiding process); b. pedestrian turns after he passing another pedestrian who blocks his way (returning process). Actually these two cases are two different parts of a same situation, people usually avoid first when they find pedestrians blocking in the front of them and then return to origin trajectories after passing the effect.

From these two cases, we can conclude two rules about judging effects (Fig. 3): a. If focus agent don’t turn and keep the direction of first fitting-line *L1*, focus agent will collide the front pedestrian, we regard the front pedestrian as an effect; b. At the same time when turning happens, if another pedestrian block the straight line from infection point *D1* to *D3*, we regard the blocked pedestrian as an effect.

According the two rules above, we label the video of ETH by watching the behaviors of every pedestrian in the dataset. There are 420 agents in the dataset, but 242 of them are lost or have no interaction with others, so we have 178 valuable samples totally.

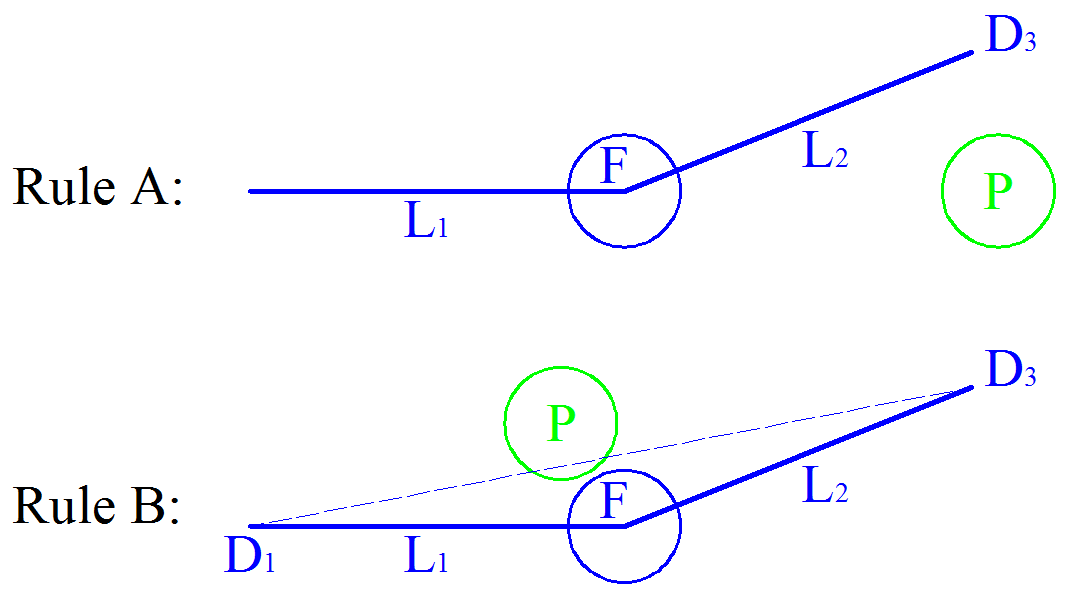


Figure 3. Two Rules about labeling. Rule *A* comes from avoiding case and Rule *B* comes from returning case.

## C. Coordinate Transformation

To observe the relative position between focus agent and effect at the same time with first inflection point, we need to transfer all the focus agents with one infection point to a same coordinate system. We transform the first fitting line *L1* to negative *Y*-axis to make sure focus agent come from -*Y* to +*Y* and move infection point *D2* to (0,0). Effects’ positions will be transformed with the same matrix and we mirror the agents which turn left to make them turn right.

Now it’s a map about focus agents’ trajectories and the effects’ positions when the inflection point *D2* happens (Fig. 4). There are a lot of effects gather around (2,0). People always turn after passing another pedestrian in the direction they’re going to turn, because they don’t want to confuse others when they turn. Some other effects appear in the front of focus agent with obvious explanation. If we explain these two phenomena according to the avoiding and returning cases in labeling process, they should behave comparative quantity as they come from a same situation. The critical reason is the limited view of our sample video, the effect’s position in the avoiding process may be out of screen while the effect in the returning process always appears in the screen. It’s also an inevitable defect of this model caused by finite view of fixed camera.

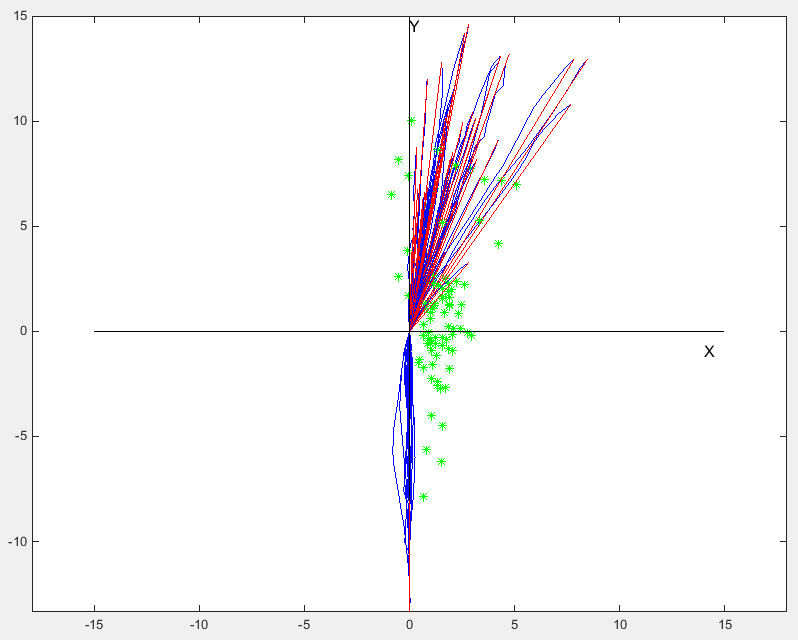
****

Figure 4. The positions of effects when the corresponding focus agents turn. Green star is effect, blue line is focus agent’s trajectory and red line is fitting-line. Most effects gather around (2,0) and others cluster in front of focus agent.

## D. Effect Model

In statistics, kernel density estimation (KDE) is a non-parametric way to estimate the probability density function of a random variable. KDE is applied to statistics widely as its property which not presume anything to distribution. The formula of one dimensional KDE is as followed:

However, the data what we are dealing with are positions, which are two-dimensional variables. For two-dimensional KDE, the hardest problem is how to select the metric *(x-xi)*. We can divide this problem to two parts: selecting coordinate system and how to design equivalent two-dimensional metric. Rectangular coordinate system is excellent for position problem as conciseness and understandability. But there are many effects’ positions close to original point in our result, if we take Euclidean distance as metric it will estimate incorrect probability around original point on account of neglecting bearing. For example, sometimes the model in rectangular coordinate system will calculate points in third quadrant as effects while our experience and result indicate points in third quadrant shouldn’t be effect. So we choose the equivalent distance of polar radius and polar angle as two-dimensional metric as followed. Here *ρi* and *θi* are polar radius and polar radius of grid points, and *ρh* and *θh* are polar radius and polar radius of sample points.

Meanwhile, the selecting of kernel function *K* and smoothing parameter *h* also influences KDE greatly. We choose Gaussian function as kernel function with experience and set smoothing parameter to 0.05 with observation about result. The two-dimensional KDE formula is as followed:

(3)

After putting sample points to formula above, we can obtain effects’ distribution model in polar coordinate system and its contour. In Fig. 5, the horizontal axis is the angle difference between original direction of focus agent and the vector from original point to effect’s position. And vertical axis is the distance between original point and effect’s position. Most possible area is around two meter and 45 degree, the contour’s behavior is compliant to the cases we have discussed before.

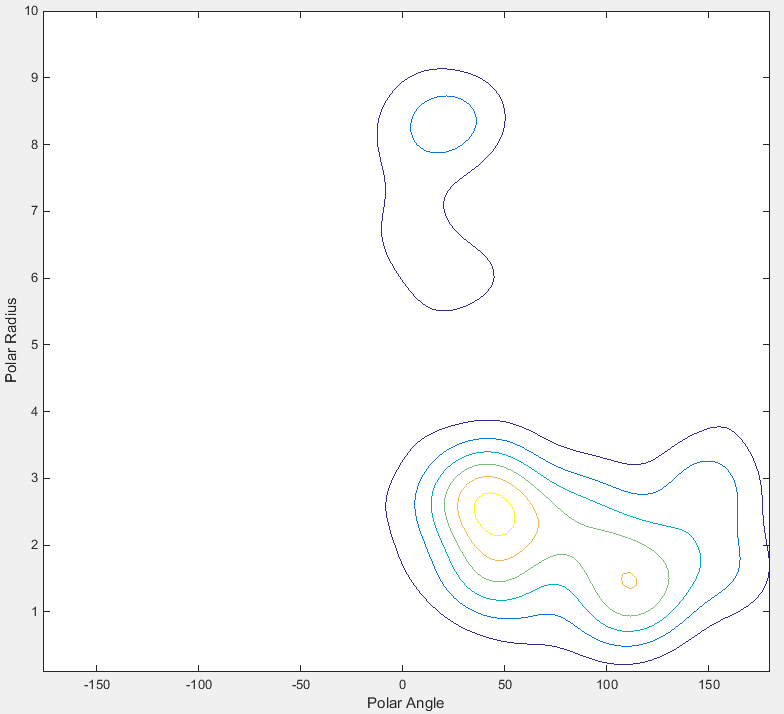


Figure 5. The contour of effect distribution when focus agent turns. The horizontal axis is polar angle and vertical axis is polar radius. Most effects gather in the turning inner side and others gather in the front of focus agent.

# Evaluation and Applications

## Evaluation

To calculate the detecting accuracy of model, we contrast another dataset of EWAP with this model. First we select eligible pedestrians as focus agents with following condition:

a. The distance between their starting point and ending point should bigger than 4 meter in case of wandering;

b. Their turning angle should be bigger than 5 degree;

c. They shouldn’t gather to a group;

d. Their fitting line should be correct.

After filtering ineligible pedestrians, we transform the trajectory of focus agent and the positions of others with the method of part III and then put the positions into Eq. 3. We could get the probability of every pedestrian which appears in the dataset. The pedestrian with biggest probability should be the effect to focus agent. There are 99 eligible samples totally. 46 samples are predicted successfully; 21 samples are unpredictable as distraction; 6 samples fail because they are influenced by environment feature; 7 samples’ real effects are not in the view or traced by dataset; 24 samples are predicted incorrectly as model defect. The evaluated result is shown as followed form.

1. Evaluation Results

| Result & Reason | Numbers |
| --- | --- |
| Successful prediction | 46 |
| Distraction | 21 |
| Environment Feature | 6 |
| Lost by tracking | 7 |
| Incorrect calculation | 24 |

The result is not ideal as various reasons. First, most error caused by calculation is derived from the difference between avoiding process and returning process. Our method takes these two processes as same situation to label sample and builds model based on that, but avoiding process happens when interaction begins while returning process happens when interaction ends. The result of the two processes may be influenced by condition changing (finite view of camera), so it will make the model inaccurate and non-adaptive in different environment. Second, pedestrians always predict the future trajectories of front pedestrians in the avoiding process. Actually avoiding pedestrian is affected by the future position of front pedestrians, our model doesn’t take time domain into consider and it will lost some effects in the front sides of focus agent. Last, the training and testing datasets of our model have to be at a low-feature environment. We haven’t found sufficient dataset for our method, and it brings inadequate robustness for our model.

We also visualize this model on testing video with symbols. The focus agent is shown by blue cross and different parts of his trajectory are shown by different colors. The squares mean other pedestrians and their lengths correspond to their probabilities to be effects. Finally, the biggest probability pedestrian is labeled by red square and others are labeled by green squares (left of Fig. 6).

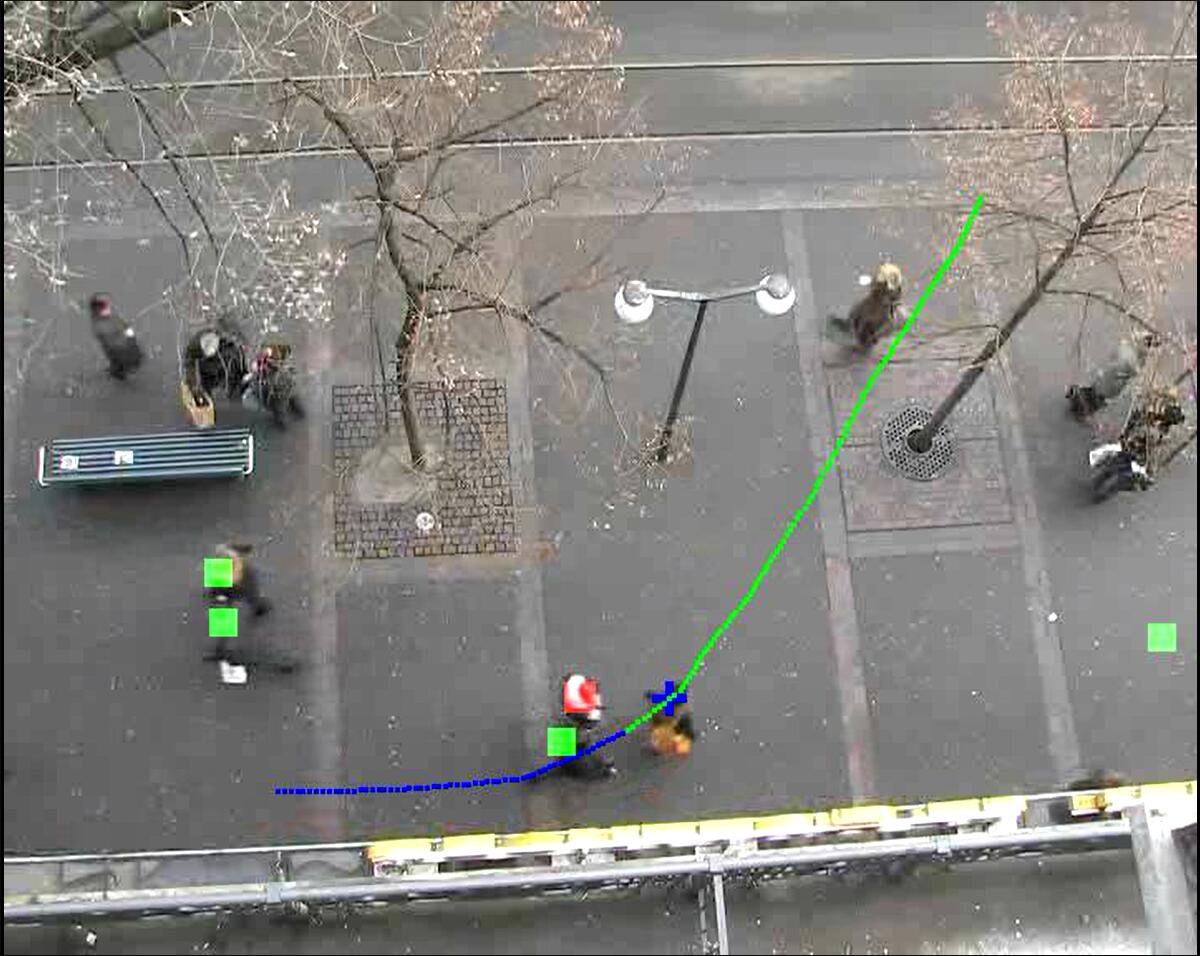
 

Figure 6. The visualization of the effect model. The blue and green lines indicate two segments of the trajectory with consistent intention. The red square indicates the most likely influencer as calculated by effect model (left). Sometimes, the model predicts an out-of-view pedestrian as designated by a red square at the edge of the screen (right).

## Destination Prediction

The question which our model solved is why pedestrian turns rather than walks straightly in low-feature environment. After solving this question, there is a deeper question behind that: why pedestrians turn on account of a special pedestrian or obstacle? If the trajectory of focus agent is influencing by a special pedestrian, then this pedestrian must block the way of focus agent. It means that focus agent wants to get the area blocked by the pedestrian.

Combining this idea and effect distribution model, it’s possible that we can predict the destination by the position of effect. But there are still some problems need to be solve, such as the effect of environment feature or finite view of camera.

## To Discover GSEs or SSEs

In an environment with pedestrians, there are two kinds of effects which interact with pedestrians: general spatial effects (GSEs) and specific spatial effects (SSEs) [4]. GSEs are detectable and concrete obstacles (trees) while SSEs is abstract implication in people’s mind (sidewalk). If we analyze abnormal trajectories with no proper reason to turn, maybe some latent information will be obtained.

After observing a focus agent with changing trajectory, assume an invisible GSE in the front of focus agent and out of the view. If this invisible GSE has biggest probability under effect distribution model, we can infer that there may be an invisible pedestrian influencing focus agent’s trajectory. It’s shown in the right of Fig. 6.

Furthermore, some SSEs’ position or feature can be analyzed by long-term learning about abnormal trajectories. But it’s difficult to conduct long-term learning only with effect distribution model, and we need another method dealing with environment feature.

# Conclusion and Future Work

In this paper, we propose a method to explain why pedestrian change its trajectory in low-feature environment based on sample learning and KDE. As this method confuses two different cases without time series, its evaluation is not satisfactory. At the end of article, we also propose two promising application: analyze GSEs and SSEs by abnormal trajectories and predict possible destination of focus agent by this model and the position of effect.

There are many future work need to be expanded. The most primary goal is collecting pedestrian dataset and tracking trajectories. Because this model need some low-feature environments to collect data, it’s easy to choose a proper condition to make sure data quality if we perform experiment by ourselves. In model improvement, the returning and avoiding processes must be analyzed apart, and introduce some other variables such as the turning angle of focus agent. In future extensional application, prediction of focus agent’s destination is the most promising work with easiness to be implemented. Although the analysis of environment feature based on this model will be more difficult if we can’t find a long-term learning way, it’s still a research direction in the future.

References

1. Helbing, Dirk, and P. Molnár. "Social force model for pedestrian dynamics." *Physical Review E Statistical Physics Plasmas Fluids & Related Interdisciplinary Topics* 51.5(1998):4282.
2. Tamura, Y, et al. "Development of pedestrian behavior model taking account of intention." *IEEE/RSJ International Conference on Intelligent Robots and Systems* IEEE, 2012:382-387.
3. Ziebart, Brian D, et al. "Planning-based prediction for pedestrians." *IEEE/RSJ International Conference on Intelligent Robots and Systems, October 11-15, 2009, St. Louis, Mo, Usa* DBLP, 2009:3931-3936.
4. Chung, Shu Yun, and H. P. Huang. "A mobile robot that understands pedestrian spatial behaviors." *IEEE/RSJ International Conference on Intelligent Robots and Systems, October 18-22, 2010, Taipei, Taiwan* DBLP, 2010:5861-5866.
5. Lee, C., and Y. Xu. "Trajectory fitting with smoothing splines using velocity information." *IEEE International Conference on Robotics and Automation, 2000. Proceedings. ICRA* IEEE Xplore, 2000:2796-2801 vol.3.
6. Mavrogiannis, Christoforos I. and Ross A. Knepper, “Decentralized Multi-Agent Navigation Planning with Braids”, *Workshop on the Algorithmic Foundations of Robotics*, San Francisco, December, 2016.
7. Gergely, György, and G. Csibra. "Teleological reasoning in infancy: The naıve theory of rational action." *Trends in cognitive sciences* 7.7 (2003): 287-292.

1. \*This material is based upon work supported by HIT Robot Group and by the National Science Foundation under Grant No, 1526035.

   Z. Qiao is with the Harbin Institute of Technology, Harbin, China qz709430186@163.com

   R. A. Knepper is with Cornell University, Ithaca, USA rak@cs.cornell.edu [↑](#footnote-ref-1)