Learning behavior patterns from video for agent-based crowd modeling and simulation

Presenter: Sudeep Basnet

2018-10-18

Outline

• Introduction
• Model Design – Abstract
• Dual-Layer Agent-based Crowd Modeling Architecture
• Top layer
  1. Behavior of new pedestrians appearing in SRs
  2. Goal selection patterns of pedestrians, using probability matrix
  3. Path navigation patterns of pedestrians
• Crowd simulation procedure
• Construction of model components
• Case Study 1 (New York Grand Central Terminal)
• Case Study 2 (ETH Walking Pedestrians)
Introduction

• Data-driven modeling framework to construct agent-based crowd model based on real-world video data.

• Can be used to predict trajectories of pedestrians in the same scenario as the video.

• Dual-layer architecture:
  • **Bottom Layer** models the microscopic collision avoidance behaviors.
  • **Top layer** models the macroscopic behaviors such as goal selection patterns and the path navigation patterns.

• Automatic learning algorithm to learn behavior patterns.
Model Design – Abstract

• Objective. Simulate crowd behaviors in a well-defined area with a number of source regions (SRs) and destination regions (DRs).

• Pedestrians enter the area via SRs and leave the area via DRs.

• First Step – Determine:
  1. How frequently pedestrians enter the SRs (arrival rate)
  2. How pedestrians select their DRs.

• Second Step – After DRs are selected, determine:
  • How pedestrians move towards their DRs (i.e., path navigation).
  • How pedestrians avoid collisions with other pedestrians and obstacles
Dual-Layer Agent-based Crowd Modeling Architecture

Fig. 1 The proposed dual-layer agent-based crowd modeling architecture
Dual-Layer Agent-based Crowd Modeling Architecture (cont.)

• In the bottom layer (microscopic), various collision avoidance models can be used.
  • Social-Force crowd Model (SFM) \[1\]
  • Reciprocal Velocity Obstacles (RVO2) \([2]\). This paper uses RVO2.

• Top layer (macroscopic) uses three components:
  1. First component describes how new pedestrians enter the simulated area.
  2. The second component models the goal selection patterns of pedestrians by using a probability matrix.
  3. The last component models the path navigation patterns of pedestrians.
Behavior of new pedestrians appearing in SRs

- Each SR (or DR) is approximately represented by a rectangle region.
- The behaviors of new pedestrians appearing in each SR is modeled as a Poisson process [3]:

$$P\{N_i(t + \tau) - N_i(t) = k\} = \frac{(\lambda_i \tau)^k \exp(-\lambda_i \tau)}{k!}$$  \hspace{1cm} (1)

- $N_i(t + \tau) - N_i(t) = k$ is the number of new pedestrians entering the $i^{th}$ SR during time interval $(t, t + \tau]$.
- $\lambda_i$ is the expected arrival rate of the $i^{th}$ SR
- Starting positions of pedestrians in each SR are modelled by using a Gaussian model.
Goal selection patterns of pedestrians, using probability matrix

• The probability of pedestrians moving from the \(i^{th}\) SR to the \(j^{th}\) DR is labelled as \(p_{i,j}\)

• The goal selection patterns of pedestrians can be represented by the following matrix:

\[
P = \begin{bmatrix}
p_{1,1} & p_{1,2} & \cdots & p_{1,M} \\
\vdots & \ddots & \cdots & \vdots \\
p_{N,1} & p_{N,2} & \cdots & p_{N,M}
\end{bmatrix}
\]  \hspace{1cm} (2)

• \(N\) is the number of SRs and \(M\) is the number of DRs.
Path navigation patterns of pedestrians

• A **Velocity field** \([4,5]\) consists of a series of position-direction pairs.
• Each position-direction pair determines the future moving directions of pedestrians near the position.
• Each DR is associated with one velocity field.
• Entire region is divided into discrete grids. Suppose \(H \times W\) grids.

\[
\mathbf{V}_i = \begin{bmatrix}
  v_{i,1,1} & v_{i,1,2} & \ldots & v_{i,1,W} \\
  \ldots & \ldots & \ldots & \ldots \\
  v_{i,H,1} & v_{i,H,2} & \ldots & v_{i,H,W}
\end{bmatrix}
\] (3)

• \(v_{i,j,k}\) is a two-dimensional vector, represents the future moving directions of the pedestrians in the grid cell \((j, k)\).
• The value of \(v_{i,j,k}\) is scenario specific, learned from video input data.
Crowd simulation procedure (Algorithm 1)

**Algorithm 1:** CROWD SIMULATION PROCEDURE.

1. for step = 1 to max_steps do
2.     for i = 1 to N do
3.         Sample a number of new agents according to $\lambda_i$;
4.         Set the starting position and velocity of each new agent;
5.         Set the global goal of each new agent according to $P$
6.     for i = 1 to total number of agents in the scene do
7.         Suppose the global goal of the $i$th agent ($A_i$) is the $j$th DR;
8.         Update the local goal of $A_i$ according to $V_j$
9.     Use RVO2 to update the velocities and positions of all agents.

$\lambda_i$ is the expected arrival rate of the $i^{th}$ SR.
$P$ is the probability matrix for goal selection.
$V_j$ is the velocity field of the $j^{th}$ DR.
Construction of model components

- The data extracted from videos are considered trajectories.
- Each trajectory is represented by a sequence of vectors.

Fig. 2: The procedures of constructing an agent-based crowd model based on video data
Initialization

• Each trajectory is divided into small number of segments.
• Each grid cell contains at most one segmentation point.
• After that, the velocity field of each DR (i.e., \(V_i\)) is then initialized by assigning each grid with a vector that directly points to the DR.

\[
v_{i,j,k} = \begin{cases} 
\Gamma(\overrightarrow{AB}), & \text{if grid cell (j, k) contains no obstacles.} \\
(0,0), & \text{if grid cell (j, k) contains obstacles.} 
\end{cases}
\] (4)

• \(A\) is the center point of the grid cell (j, k), \(B\) is the destination.
• \(\Gamma(\hat{a})\) normalizes the input vector \(\hat{a}\) in such a way that the total length of \(\hat{a}\) is much smaller than the size of a single grid.
Construction of model components based on off-line video

- The K-means clustering algorithm is used to cluster all trajectories into $M$ groups, where $M$ is the total number of DRs.
- The trajectories in the same group are expected to terminate at the same DR.
- In each iteration of the K-means clustering process, each trajectory is classified into the nearest cluster.
- After clustering all trajectories, each cluster $C_i$ is then used as the temporary velocity field of the corresponding $i^{th}$ DR.
- Temporary velocity field is further revised by considering neighboring influences.
Construction of model components based on off-line video (cont’d. 1)

• Vector closer to the grid \((j, k)\) will have a larger influence on \(v_{i,j,k}\)

• If obstacles are seen, all eight neighborhood of \((j, k)\) are checked, the best grid that is not inside obstacle and has the least turning angle is chosen.

Fig. 3: Revising an infeasible velocity
Construction of model components based on off-line video (cont’d. 2)

• Once the velocity fields of all DRs are obtained, they can then be utilized to learn the state transition rates between SRs and DRs (i.e., the probability matrix).

• Choose trajectories that start at $i^{th}$ SR and end in other regions.

• $S_{i,j}$ is the total number of trajectories that have the $i^{th}$ SR as source and the $j^{th}$ DR as destination.

• Transition probability from the $i^{th}$ SR to the $j^{th}$ DR is:

$$ p_{i,j} = \frac{S_{i,j}}{\sum_{k=1}^{M} S_{i,k}} $$

(5)
Case Study 1 (New York Grand Central Terminal) - Trajectories

Fig 4: A frame of the video and the transformed trajectories used in the first case study. (a) A frame of the video; (b) the transformed trajectories
Case Study 1 – Velocity Fields

Fig 5: Two examples of the learned velocity fields in the first scenario
Case Study 1 – Crowd Density

- Objective crowd density calculated based on the video data.
- Density values of a narrow region from (20, 0) to (37, 15) are the largest.
- Density values in the center circle are very small, because there is a counter located in that region.

Fig 6: The objective crowd density distributions in the first scenario
Case Study 1 – Crowd Density – Comparisons

Fig 7: The crowd density distributions of the six methods in the first scenario
Case Study 1 – Predicted Trajectories – Comparisons

Fig 8: Examples of the predicted trajectories found by the D-ABC and the ConVelocity in the first scenario
Case Study 1 – Error Analysis

<table>
<thead>
<tr>
<th>Method</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>D-ABC</td>
<td>274.8</td>
</tr>
<tr>
<td>D-ABC/T</td>
<td>295.3</td>
</tr>
<tr>
<td>D-ABC/V</td>
<td>292.0</td>
</tr>
<tr>
<td>D-ABC/VT</td>
<td>318.1</td>
</tr>
<tr>
<td>RVO2-random</td>
<td>336.0</td>
</tr>
<tr>
<td>ANN</td>
<td>290.4</td>
</tr>
</tbody>
</table>

Table 1: The $\rho_{error}$ results of the six methods in the first scenario

<table>
<thead>
<tr>
<th>Method</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConVelocity</td>
<td>2.58$^a$</td>
</tr>
<tr>
<td>D-ABC</td>
<td>1.79</td>
</tr>
</tbody>
</table>

Table 2: Average of the final prediction errors in the first scenario

Fig 9: Average prediction errors of the D-ABC and the ConVelocity in the first scenario
Case Study 2 (ETH Walking Pedestrians)

Fig 10: The video and the trajectories used in the second case study. (a) A frame of the video; (b) the trajectories used
Case Study 2 – Velocity Fields

- velocities do not always directly point to the destinations.
- In regions near obstacles, they point to other directions so as to guide agents to avoid obstacles.

Fig 11: Examples of the learned velocity fields of the second scenario
Case Study 2 - Crowd Density

- crowd density of the video data.

Fig 12: The objective crowd density distributions in the second case study
Case Study 2 – Crowd Density – Comparisons

Fig 13: The crowd density distributions of the six methods in the second case study
Case Study 2 – Predicted Trajectories – Comparisons

Fig 14: Examples of the predicted trajectories found by the D-ABC and the ConVelocity in the second scenario
Case Study 2 – Error Analysis

<table>
<thead>
<tr>
<th>Method</th>
<th>perror</th>
</tr>
</thead>
<tbody>
<tr>
<td>D-ABC</td>
<td>1.97</td>
</tr>
<tr>
<td>D-ABC/T</td>
<td>2.40</td>
</tr>
<tr>
<td>D-ABC/V</td>
<td>2.12</td>
</tr>
<tr>
<td>D-ABC/VT</td>
<td>2.57</td>
</tr>
<tr>
<td>RVO2-random</td>
<td>2.86</td>
</tr>
<tr>
<td>ANN</td>
<td>2.04</td>
</tr>
</tbody>
</table>

Table 3: The perror results of the six methods in the second scenario

<table>
<thead>
<tr>
<th>Method</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConVelocity</td>
<td>1.48°</td>
</tr>
<tr>
<td>D-ABC</td>
<td>1.23</td>
</tr>
</tbody>
</table>

Table 5: Average of the final prediction errors second scenario

Fig 15: Prediction errors of the D-ABC and the ConVelocity in the second scenario
Conclusion – Authors

• Proposed a generic data-driven crowd modeling framework to generate realistic crowd behaviors that can match the video data.
• Both the microscopic collision avoidance behaviors and multiple macroscopic behaviors.
• Results have shown that our proposed framework is effective to generate crowd behaviors, in terms of crowd density.
• Future work could include modelling for different types of agents.
• Using machine learning techniques to distill generic behavior instead of modelling specific scenarios, is promising future work.
My Conclusions

• It would be interesting to apply this concept in a chaotic scenario. To track and study trajectories of people in the presence of danger or during an unrest.

• Agent are of uniform characteristics, variables such as age or gender could not be considered with this method.

• The method of considering shortest paths from point A to point B, may only be true during rush hours and in places such as transport stations.

• While some limitations have been discussed in the paper, there have been no experiments to test exactly where the limitation lie in finding complicated trajectories such as a “W” shaped trajectory.

• It is not clear how obstacles are generated in the simulation models. The paper only shows how the model avoids obstacles.

• A general relationship of the error and grid size is needed, the choice of grid sizes used for experiments are not explained. It seems to be too scenario-specific.

• In case study 1 (New York), 40000 trajectories are extracted from the video. The amount of computation seems to be very large, yet the paper doesn’t discuss such challenges.
References


