**Paper title: Crowd Simulation: Queue-up Behavior**

1. I am here to present agent-based crowd simulation approach which allows agents to form waiting lines and do the security checking processes. In the simulation, agent could also move in and out of a waiting line based their own desire.
2. **We recorded videos during** events such as concerts or WWE and then study and analyze those videos and we have these 4 common behaviors:

* Stay pair while they are walking or waiting
* People form waiting line
* Queue up behavior
* People switching from a long line to a shorter one.

1. **For the architecture**, we have 3 layers:
   1. For bottom layer, we use the **recast library** to achieve the Local Obstacle avoidance.
   2. For medium layer, we use the **detour library** to do the global navigation.
   3. For the top layer, this layer is to handle the social queuing behavior that built upon those lower layers.
2. Each dot represents agent, for agent is circled by red line, they are security agents, others are normal agents. Normal agents are coming from the left and walking to the right to enter the event.
3. (simulation result) Let’s watch the simulation result, this video shows the comparison of two simulation results. The one on the left has all behavior features implemented. Another one just simply implemented the security checking feature. For the simulation: Before entering the event, all the agents have to finish two security processes. One is to check their ticket; another one is to do the body security check.   
   During these processes, we can see agents walk and line up in pair, and waiting lines formed, some of them will leave the current waiting line and move to the shorter one.
4. (evaluation). For the evaluation, participants indicated which crowd looks more natural after watching the simulation comparison, participants will watch the comparison video and then pick the video they think is more natural.

**Presentation Outline**

Name: Zhicheng Chen

Title: Research project – crowd simulation

**过场对白**

**Outline**

My presentation will be in three parts:

In the first part, I will introduce purpose of my project and briefly talk about related works

Then in second part, I will talk about the details of implementation

Finally, I’ll go on to talk about the project evaluation and result

**Please interrupt if you have any questions.**

**Introduction**

**Importance**:

Now let’s move to the first part : which is about introduction and related works.

So, there are 7.8 billion 🡪 World. What’s more, you might have seen many people 🡪 Street or Inside the building.

Big challenge to Urban planners/architects to have a safety design to handle a large amount of people. Asking a large amount of people to do the related experiments to observe the crowd movement is unrealistic and expensive.

But my simulation approach could efficiently solve this kind of problem.

**Contributions:**

Our simulation approach has the following contributions: agents could form waiting lines and process the security checking processes. Agent could also move in and out of a waiting line with their own desire. What’s more, agents could have social interaction while they are walking or waiting. Our approach finally has the statically significant result from the user surveys.

There are numbers of previous work:

**Previous works:**

Figure 1 shows an approach that designs agents as ellipses that have a sense of the environment and plan their own path ahead of time to avoid agent collisions.

Figure 2 provided a dynamic agent-based approach that allowed agents in a scene to have distinct goals and plan their own movements and collision avoidance. [A continuous cycle of sensing and acting is run, allowing the agent to adapt its behavior based on the feedback received from the environment after performing a certain action.]

Figure 3 extended state-of-art predictive approaches with social awareness, prediction, and social collision avoidance to achieve prediction in social path following behavior.

That’s all on the first part.

Lest’ move to the second part, **implementation**.

**Implement detail**

**Observation**

Before dive into the behavior implementations detail, let me introduce the source of behavior features I have. So, we study and analyze crowd videos we recorded at CenturyLink at different entrances during events such as concerts or WWE, below are common behaviors we observed from those videos:

* Stay pair while they are walking or waiting
* Queue up and form waiting line
* Queue up and form waiting line
* People switching from a long line to a shorter one.

[**Page 7**] Then I will describe the architecture and scenario

**Architecture**

let’s have a basic concept of the architecture of the application:

From the architecture perspective, we could divide the application into 3 layers: low, medium and high.

For low layer, we could call it dynamic avoidance, it uses the detour library to achieve the Obstacle avoidance and it calculates once per frame.

For medium layer, we could call it static avoidance, it uses the recast library to go to destination with wall avoidance and calculate the shortest path, calculate once per simulation for new destination.

For the high layer, this layer is handled by my approach that built upon those previous layers to achieve more complicated behaviors. And it will be discussed in the next slide.

Combine to the observations we have from the video records. we have the following scenario:

**Scenario**

The image is a screenshot of the simulation result. In the scene, we could assume there has a concert event. Then each circle represents people, and they are coming from the left and walking to the right to enter the event. However, before entering the event, they have to finish two security processes. One is checking their ticket; another one is to do the body security check. After finishing both processes. Agents could continuously enter the concert.

[在第九页]In order to make agents walking in the scene, we need to generate the agents and set up the environment. So we have two major initialization processs

**Agent initialization and Environment Setup**

For input data initialization

**Input Data Initialization**

This process determines agents’ basic information. Include Agent’s id, start time, start position, end position, and behavior mode. We can see digits on image below is divided into several groups by color. [介绍什么颜色代表什么]. The last features cannot directly showed by the just couple lines of data. However, the appearance of upcoming agents is determined by the number of agents enter the scene per second. The chart above illustrates the distribution of the appearance of the agents roughly follows the bell-shape pattern. This figure presents the number of new agents appear every certain amount of time. So we can see the number of upcoming agents reach the max when the simulation time is near 60 seconds.

Next is the environment setup process.

**Environment initialization**

Environment initialization process includes gates, pair relationship, agent’s checking time, anxiety and their gate choice. Figure on the left could clear demonstrate how the initializations effect the appearance of the simulation result. For example, gates initialization determines the position of gate and the direction the agent walking to. Agent’s checking time init determines how long the agent will stand in front of the gate.

The dash line rectangle determines the range of agents appear and disappear positions. Then, diff roles will be assigned to the agents. Faculty agents are black, and they stay static and help others to do security check. Normal agents are grey, and walking left 🡪 right. Among those normal agents, some of them walk in pair. In this initialization process, it will also determine how long the agents will do for the security check. Value will be assigned to each agent to represent their anxiety which could help make line change decision that I will mention later. Then when agents appear, they are going to pick the closest gate to go to, this feature also handled by this initialization process.

After the agents and the environment are setup, agents start to move in the scene. And we allow agents to achieve the following behavior features.

**Agent behaviors implementation**

* Pair walking behavior
* Queue up behavior
* Re-consider behavior

**Pair walking behavior**

How the agents achieve the pair walking.

In this simulation, we allow at most 2 agents to do pair walking. For agent in pair relationship, both agents have their own role. One is leader, one is follower.

So how they adjust their speed?

Let’s look at the image, follower agent might walk at left or right side of the leader agent. Both agents’ speed will be determined by angle between 2 vectors: one is from leader to its follower (leader🡪follower), another one is from leader to the leader’s next destination (leader🡪leader’s next destination). When the angle is less than 90, leader will slowly increase its speed, its follower will slowly decrease its speed until the angle is 90. If the angle is larger than 90, it works the opposite, leader will slowly decrease its speed, and its follower will slowly increase its speed.

Next part is the implementation of the queue up behavior

**Queue up behavior**

Agent’s queue up behavior is determined by its current state. So I use this state diagram to illustrate how agent’s behavior transforms from one to another.

**After agent is initialized**, it starts to walk to it first checking spot. Because the target gate it chooses might be occupied by the other agents already. So, the agent might directly reach the empty gate or queue up the tail of a waiting line. Once agent reaches its current destination, its state **will be changed to waiting1 state**.

Then for agents that in waiting state, if they are not the head of the waiting line, it will stay at waiting line and then slowly moving forward to the checking spot. During this process, agent could move out of the current waiting line and queue up another shorter waiting line.

For the head agent, it immediately starts the internal timer to count the time to simulate the ticket checking process, which means agent will stand at the checking spot for a few seconds. After that, agent will check if there have space to move. Why it needs to check, because the space between the first checking spot and the second checking spot is limited. If there has no available space, it will wait at the first checking spot until it has space. If it has space, it will pass through **the first checking spot and set its state to checked1**.

Then the second checking process will work in the same way. Once agents finish both checking processes, **they will be in checked2 state** and continuously move to the final destination, which means walk out of the scene.

As we mentioned on previous page, agent could move out of the waiting line and queue up on a shorter line, this is the re-consider behavior.

**Re-consider behavior:**

How we achieved this: we allow agents to have “anxiety”. Agents are assigned a value to represent it anxiety degree.

Agent makes line changing decision based on two factors: one is its **anxiety degree**, another one **is its related position**.

The anxiety degree is effect by the related position. How the related position effect the anxiety degree: as we can see in the image below, the related position is determined by the difference between two distances: A and B.

A is length of the other waiting line; B is the length of agent’s current position to the target checking spot. If diff of A and B reach certain limit, agent’s anxiety degree will be increase.   
  
The maximum of the anxiety degree is 4. When it reaches the maximum and the diff we just mention reaches the certain number, it will leave the current waiting line and move to another short one.

Finally, lest’ move to the second part, **Evaluation**

**Evaluation**

Overall, the evaluation process includes three parts:

* Input data evaluation
* Behavior evaluation
* Mechanical Turk survey

The first part is the input data evaluation.

**Input data evaluation**

The input data follows certain patterns. For example, agent’s id, enter time should be in ascending order, start position and end position should locate at correct position, agents in pair relationship should have close enter time, start position and end position, etc.

After the input data is properly generated, we start to do the simulation.

未完

**Behavior evaluation**

The second evaluation process is the behavior evaluation. In this step, we will watch the simulation result. Based on the simulation result, we mark items in following check list.

**Mechanical Turk survey**

What is Mechanical Turk survey: Amazon Mechanical Turk is a crowdsourcing marketplace that makes it easier for people to outsource their works to a distributed workforce who can perform tasks virtually. Figure is the interface of Mechanical Turk survey. After reading the question: ”indicate which crowd looks more natural”, participants will watch the comparison video and then pick the video they think is more natural. The video shows the comparison of two simulation results. One has all behavior features implemented. Another one just simply implemented the security check feature. Participants make their own decision after watching the whole videos. In order to avoid the influence of the video position, those two videos will flip for different tests.

Next let’s see the survey result.

**MT survey table:** The following chart shows survey result.

There are 108 participants in total. When all features implementation video is on the left, there are 33 participants vote for this, and 18 participants voted for the other. For another survey, there are 37 participants voted for all features video, and other video had 20 votes. After the calculation, we find out no matter the all features video is on the left or right, we still could get the statically significant, which means the simulation approach is XXXXX

**Future work**

So far we only allow at most 2 agents to have social relationship, in the future, we will make the social relationship to be more dynamic, to allow it to have various number of agents in a group. Then, so far, the wait line is roughly straight, in the future, we will let the agents to form waiting lines that could adjust more complex circumstance, such as fense..

**Thank you!**

**Any question?  
- “No.”**

**- “No.”**

**Thank you!**