User Guide of CrowdEgress: A Simulation Tool for Multi-Agent Model of Crowd Evacuation

October 14, 2020

1 Introduction

The program mainly consists of four component: User Interface, Simulation Core, Data Tool, Visualization Tool

User Interface: The user interface is written in tkinter in ui.py. Please run ui.py to enable a graphic user interface (GUI) where one selects the input files, initialize compartment geometry, and configure or start a simulation. An alternative method is using evac-non-gui.py to directly start a simulation without GUI. Currently there is a simple version of GUI and it needs to be improved in several aspects. If you find any problems when using the user interface, please send me a message or direct start an issue here.

Simulation Core: The multi-agent simulation is implemented as simulation.py. The component is packed in a class called simulation class, and it computes interaction of four types of entities: agents, walls, doors and exits. The agent model is described in agent.py, while walls, doors and exits are coded in obst.py. The agent-based model is an extension of the traditional social force model by Helbing, Farkas, Vicsek and Monlar. The model aims at investigating protypes of pedestrian behavior in crowd evacuation. The core algorithm is still being studied and developed. This is an interesting study topic, which refers to Newton particles, complex systems and behavioral science. Your comments or contribution are much welcome.

Data Tool: This component reads in data from input files, and write data to output files. The input data is written by users in .csv files. Agents and exits must be specified in .csv file while walls and doors can be described either in .csv file or read in from standard .fds input file. In the future We plan to use a subroutine in FDS+Evac to output the agent movement data so that the agent movement can also be visualized by smokeview.

Visualization Tool: The visulization component is packed in draw_func.py and currently pygame is used to visualize the simulation result. We may develop another visualization tool together with smokeview such that users first run the simulation and get the output data, and then visualize the output data. Currently the visualization functions are packed up as an independent module

in draw_func.py. If any users are interested, please feel free to extend the module or try other graphic library to wrtie a visulization component.

2 About Simulation Model

In the simulation core there are four types of entities: agents, walls, doors and exits.

Walls: Walls are obstruction in a compartment geometry that confine agent movement, and they set up the boundary of a room or certain space. Just like walls in normal buildings, walls may be labeled with arrows that direct agent to move toward certain directions. In our program wall are either lines or rectangular areas. If any users are interested, please feel free to extend the wall types to circular or polyangular areas.

Doors: Doors are passageways that direct agents toward certain areas, and they may be placed over a wall so that agents can go through the wall by the door. Doors can also be placed as a waypoint if not attached to any walls, and they can be considered as arrows or markers on the ground that guide agent movement. In brief doors affect agent way-finding activities and they help agents to form a roadmap to exits. In current program doors are only specified as rectangular areas.

Exits: Exits are a special types of doors which finally evacuate agents to the safety. Thus they may be considered as safety areas, and computation of an agent terminates when the agent reaches the exits. An exit is usually placed over a wall like doors, but it can also be put anywhere independently without walls. Exits must be specified in .csv file. In the program exits are only defined as rectangular areas.

Agents: Finally and most importantly, agents are the core entity in computation. They interact with each other to form collective behavior of crowd. They also interact with above three types of entities to form egress motion toward exits. The resulting program is essentially a multi-agent simulation of pedestrian crowd. Each agent is modeled by extending traditional social force model. The model is further advanced by integrating several features including pre-evacuation behavior, group behavior, way-finding behavior and so forth.

3 How-To

In Tkinter Screen: When tkinter window (GUI) is activated, please select the input files for simulation. Choose .csv file for evac input data. Users can optionally use .fds file to create the compartment geometry, and the pedestrian features are described in .csv file. If both .csv and .fds files are presented, the compartment structure will be created by using .fds file. If .fds file is omitted, the compartment geometry is described in .csv file. The agents and exits must be specified in .csv file currently while the walls and doors can either described by .csv file or .fds file. Please take a look at the examples for details.

In Pygame Screen: When pygame screen is displayed, press keys to adjust the display features and visualize entity data. Use pageup/pagedown to zoom in or zoom out the entities in screen Use space key to pause the simulation Use arrows to move the entities vertically or horizonally in screen. Use 1/2/3 in number panel (Right side in the keyboard) to display the door or exit data on the screen.

There are two sections in pygame screen. One section is TestGeom, where users can visualize geometric settings and modify them manually. Users can also dump geometric data (e.g., wall data and door data) into .csv file by selecting the items in the menu bar.

The other section is RunSimulation, in which the evac simulation is started and visualized on the screen. User can pause the simulation, but cannot rewind it in current version (Version 2.0)

Try Examples: There are currently two examples in the repo. Please execute run.bat in the subfolders to run the example. The examples are run by using non-gui method. Users can also learn how to write the .csv files from the examples.