Validation of Agent Based Crowd Egress Simulation (Extended Abstract)

Bikramjit Banerjee
School of Computing
The University of Southern Mississippi
118 College Dr. # 5106
Hattiesburg, MS 39406, USA
Bikramjit.Banerjee@usm.edu

Landon Kraemer
School of Computing
The University of Southern Mississippi
118 College Dr. # 5106
Hattiesburg, MS 39406, USA
Landon.Kraemer@eagles.usm.edu

Introduction

Crowd behavior simulation has been an active field of research because of its utility in several applications such as emergency planning and evacuations, designing and planning pedestrian areas, subway or rail-road stations, besides in education, training and entertainment. The most advanced and realistic simulation systems employ intelligent autonomous agents with a balance between individual and group intelligence for scalability of the architectures. Although several systems have even been commercialized, little attention has been accorded to the problem of validating the outcomes of these simulations in a generalized manner, against reality. The extent of validation fails to go much beyond visual matching between the simulation and the actual scenarios (with recordings of human crowds), which can lead to highly subjective and often questionable conclusions. The existing numerical measures often rely on ad-hoc *applications*, e.g., local crowd densities are measured to verify patterns, without a systematic procedure to identify *at what times* in the simulation and the scenarios can the densities be compared. Furthermore, if there are multiple systems that simulate crowd behavior in the same scenario in the same virtual environment, then no technique is currently known to quantitatively compare these systems in terms of realism. In this abstract, we present the first (to the best of our knowledge) principled, unified, and automated technique to quantitatively validate and compare the global performance of crowd egress simulation systems. We also evaluate a multi-agent based crowd egress simulation system (that we have recently developed, but we do not discuss this system here) using our technique and demonstrate a high degree of validity of that system as well.

The validation algorithm

Almost all of the simulation systems include a detailed geometry of the environment (such as a sport stadium, or a subway station), partitioned into a set of p convex polygons, $\{R_1, R_2, \ldots R_p\}$. These polygons are the surfaces on which the autonomous agents can move. The system simulates the navigation behavior of the agents in context of the environment's geometry. In an egress simulation, there is one or more *sink* of agents (i.e., where agents exit the virtual environment), but there is no *source* of agents (i.e., no polygon where agents are spawned). Instead, the agent distribution over the regions is pre-specified at the start of the simulation, and changes only by the simulated egress behavior.

In order to evaluate such an egress simulation system, we subscribe to the general idea of seeking macroscopic patters, but in a quantitative manner. We compare the simulation run in the virtual environment, and the actual scenario with real people navigating the corresponding real environment, with identical initial conditions. This comparison is accomplished by calculating the distance between the distributions of agents over the p poygonal regions in the simulation (sim), and the distribution in the actual scenario (scn) over the same regions, viz.,

$$D(sim_t, scn_{t'}) = \sum_{i=1}^{i=p} \left(\frac{C^t_{sim}(R_i) - C^{t'}_{scn}(R_i)}{X(R_i)} \right)^2$$

where $C^t_x(r)$ gives the count of the number of agents in region r at time t, and $X(R_i)$ is the maximum capacity of the region R_i. It is important to notice that the time-points at which these comparisons can be made might be on different scales, due to difference in the speeds between agents and people.

We assume that $C^t_{scn}(R_i)$ values are given for certain discrete time points $t_0, t_1, \ldots t_k$ at constant interval τ_{scn}, i.e., $t_i - t_{i-1} = \tau_{scn}$, $\forall i = 1 \ldots k$. These could be generated from the snapshots of crowd video at regular intervals. We call these time points *snap points*. At the start of validation, the distribution $C^{t_0}_{scn}(R_i)$ is replicated in the simulation and it is run for τ_{sim} units of simulation time. Then $D(sim_{\tau_{sim}}, scn_{t_1})$ is compared to a distance threshold δ ($0 \leq \delta \leq 1$). If the former is lower then the simulation continues to run, otherwise it "snaps". This means that the distribution $C^{t_1}_{scn}(R_i)$ is replicated in the simulation and it is run for another τ_{sim} units of time. This process continues with the comparison of $D(sim_{i*\tau_{sim}}, scn_{t_i})$ against δ in the ith step, until we reach t_k, and at this point the total number of "snaps" – represented as n_δ – is recorded as a function of δ.

The need to check for "snaps" at a regular interval stems from the fact that most sophisticated crowd simulation systems incorporate complex agent-based models, producing non-linear behavior as a function of time. Such emergent behavior can quickly diverge from the behavior being modeled, if small discrepencies are allowed to accumulate, due to chaotic effects. Hence, any discrepency that is determined to be sufficiently large needs to be reset, by re-matching the

Cite as: Validation of Agent Based Crowd Egress Simulation (Extended Abstract), Bikramjit Banerjee and Landon Kraemer, *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, van der Hoek, Kaminka, Lespérance, Luck and Sen (eds.), May, 10–14, 2010, Toronto, Canada, pp. 1551-1552

simulation distribution with the corresponding distribution from the scenario – a process that we call "snap". The main idea is that a simulation system that needs to snap fewer times for a given δ, compared to another simulation system in the same scenario, is more accurate for that scenario and for that δ. A metric for comparison between two simulation systems in terms of accuracy for any given scenario could be the area under the curve of n_δ vs. δ; the lower this area, the more accurate the simulation system. This is illustrated in Figure 1 (left).

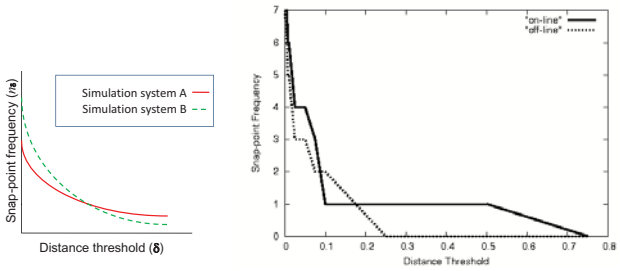

Figure 1: Left: Illustration of n_δ vs. δ curves for two simulation systems. Right: Actual n_δ vs. δ plots from the two methods for computing τ_{sim} for a simulation system.

The main challenge underlying the above validation algorithm is in calculating the appropriate value of τ_{sim}, since the timescale may differ between the simulation and the scenario, e.g., the agents may be navigating too fast compared to real people in the scenario, or too slow. Moreover, these speed-discrepencies may not stay constant, but vary from region to region and also with time. Any endeavor at such synchronization may also be affected by micro-level occurrences in reality that may be hard to reproduce spatially and temporally in simulation, such as people stopping to chat (in a zero-panic scenario), which ultimately affects the count distribution, and thus D. The premise of our validation approach is that the inevitable mismatches in micro-level occurrences can be overlooked as long as the macro-level patterns (e.g., agent count distribution over regions) match well enough. Clearly, the problem of finding appropriate times to match the simulation with the scenario at the macro-level is the key to this approach. We now describe two approaches – an off-line and an on-line approach – to calculating τ_{sim}.

Off-line (pre-)calculation of τ_{sim}

Here we describe an off-line strategy to find the best value of τ_{sim} that will enable the closest possible match between the simulation and the scenario. Let $C^t_x = \sum_i C^t_x(R_i)$. Then $L^j_{scn} = C^{t_0}_{scn} - C^{t_j}_{scn}$ is the number of people that have left the environment (i.e., the set of p polygons) between time t_0 and t_j. The sequence $\{L^j_{scn}\}_{j=1}^{j=k}$ must be a non-decreasing sequence to match with the assumption of there being no agent source in the simulation.

In order to pre-calculate τ_{sim} before the validation runs, we run the simulation once, populated with the distribution $C^{t_0}_{scn}(R_i)$, and record the total agent counts C^t_{sim} at small intervals Δ, thus producing a sequence $C^0_{sim}, C^\Delta_{sim}, C^{2\Delta}_{sim}, \ldots, C^{m\Delta}_{sim}$, where $C^{m\Delta}_{sim} \leq C^{t_k}_{scn}$ and $C^{(m-1)\Delta}_{sim} > C^{t_k}_{scn}$. The existence of m is guaranteed by the assumption that there is no agent source in the simulation. As in the case of the scenario, we can then generate another non-decreasing sequence $\{L^j_{sim}\}_{j=1}^{j=m}$, where L^j_{sim} is the number of agents that have egressed the virtual environment by time $j\Delta$ from the start of the simulation. If Δ is chosen sufficiently small, then it can be ensured that $m \gg k$.

Given the above set-up, τ_{sim} can now be estimated as

$$\tau_{sim} \approx h^* \Delta$$

where integer h^* is found by solving the discrete optimization problem $h^* = \arg\min_h \sum_{s=1}^{k} (L^{sh}_{sim} - L^s_{scn})^2$ under the constraint that $kh \leq m$.

On-line adjustment of τ_{sim}

Another technique for calculating τ_{sim} is to adapt it on-line during evaluation of each segment (between snap-points). In particular, the information generated during the run through the segment t_{j-1} to t_j is used to adapt τ_{sim} for the segment t_j to t_{j+1}. Let τ^j_{sim} be the value of τ_{sim} calculated at snap-point t_j, and used for this segment. This is calculated as

$$\tau^j_{sim} = \tau^{j-1}_{sim} \cdot \left(\frac{C^{t_{j-1}}_{scn} - C^{t_j}_{scn}}{C^x_{sim} - C^y_{sim}} \right)$$

where $x = \sum_{t=0}^{j-2} \tau^t_{sim}$, and $y = x + \tau^{j-1}_{sim}$, for $j = 1, \ldots, k-1$, with $\tau^{-1}_{sim} = 0$, and $\tau^0_{sim} = t_1 - t_0$. The ratio $\frac{C^{t_{j-1}}_{scn} - C^{t_j}_{scn}}{C^x_{sim} - C^y_{sim}}$ gives the relative number of egresses between the scenario and the simulation during the segment t_{j-1} to t_j, and acts as an estimate of how much faster (if < 1) or slower (if > 1) the simulation must be interrupted (and possibly snapped) for the current segment, to produce a better (expected) match with the following segment of the scenario.

Although this method is computationally cheaper than the off-line calculation of τ_{sim}, the estimate is expected to alternate between overcompensation and undercompensation for the discrepency in the number of egresses between the simulation and the scenario. Hence, unless k is large, the estimate may not find enough time to stabilize. Therefore we expect the off-line estimation method to produce greater accuracy for validation with a few snap-points.

Results

We have implemented both techniques for estimating τ_{sim}, and evaluated a single segment of 7 minutes duration (with 7 possible snap-points at 1 minute intervals) from the video footage of spectators from an actual football game. The segment starts with the final declaration of scores in the game, at which point spectators start to leave in large numbers. The segment ends (roughly 7 minutes later) when almost all spectators have left. We have recently developed a multi-agent based crowd egress simulation system which we used in the current evaluation against this video segment. The plots of n_δ vs. δ resulting from the application of our validation algorithm is shown in Figure 1 (right). As expected, with just 7 snap-points, the off-line estimation produces a better snap-point frequency curve. This off-line plot also shows that the simulation system does not need to snap even once for $\delta > 25\%$, which clearly establishes a high accuracy for this system for the given segment.

Acknowledgment: This work was supported in part by the U.S. Dept. of Homeland Security through the SERRI (Southeast Region Research Initiative) Program at the Oak Ridge National Laboratory.