CODE: Crowd Optimized Design of Environments

Brandon Haworth
York University
brandon@cse.yorku.ca
www.eecs.yorku.ca/∼brandon

Muhammad Usman
York University
usman@cse.yorku.ca
www.eecs.yorku.ca/∼usman

Glen Berseth
University of British Columbia
gberseth@cs.ubc.ca
www.cs.ubc.ca/∼gberseth

Mahyar Khayatkhoei
Rutgers University
m.khayatkhoei@cs.rutgers.edu
www.rci.rutgers.edu/∼mk1391

Mubbasir Kapadia
Rutgers University
mubbasir.kapadia@rutgers.edu
www.cs.rutgers.edu/∼mk1353

Petros Faloutsos
York University
pfal@cse.yorku.ca
www.eecs.yorku.ca/∼pfal

Abstract

We present CODE: a *crowd-aware* computational tool for designing environments (e.g., building floor plans). Our system analyses the impact of newly added environment elements (e.g., pillars or doorways) on the resulting crowd flow, using current-generation crowd simulators. The results of the simulation are used to provide feedback to the designer in terms of aggregate statistics and heat maps. Additionally, our system is able to *automatically* optimize the placement of environment elements to maximize crowd flow in egress scenarios, while satisfying constraints that are imposed by the designer. Using CODE, architects and environment designers can iteratively refine upon their original design to quickly accommodate the dynamic properties of crowd simulations in an interactive fashion. CODE is modular and flexible so that designers may build environments, select from different crowd simulators, and specify varying crowd configurations.

Keywords: architectural optimization, user-in-the-loop design, crowd simulation

Introduction

The movement, and moreover the flow, of people through an environment is particularly difficult to explore in an arbitrary environment design. Exhaustively searching this environment configuration space while taking into account the subtle and aggregate effects on crowd flow is generally impractical and prohibitively expensive in terms of both computation and human labour. However, the placement, or arrangement, of features of an environment may facilitate or adversely affect the flow of crowds. To address this issue, the growing maturity of research in simulating both the microscopic as well as aggregate dynamics of crowds [1] has opened the possibility of using computational methods to facilitate the environment design process.

In this paper we propose CODE: a prototype *crowd-aware* computational tool for designing environments (e.g., building floor plans). Our system analyses the impact of newly added environment elements (e.g., pillars or doorways) on the resulting crowd flow, using current-generation crowd simulators. The results of the simulation are used to provide feedback to the designer in terms of aggregate statistics and heat maps. Additionally, our system is able to *automatically* optimize the placement of environment elements to maximize the flow of the crowd in egress scenarios, while satisfying constraints imposed by the designer. CODE affords architects interactive means to iteratively refine upon their original designs to quickly accommodate the dynamic properties of crowd simulations.

To our knowledge, CODE is the first approach to integrate crowd simulation, analysis,

and optimization by closing the feedback-design loop to provide a computer-assisted tool for environment design that intrinsically accounts for the dynamic flow of people. Current systems use simulations to provide visual feedback either in terms of trajectories, or other high-level features (flow), but require human experts to manually interpret these measures and integrate them into the design process. In these solutions, the term "optimization" is used to encompass the manual iterations of a designer. These systems provide means of analysis in that they have visual crowd simulators and may provide agent path renders, time metrics, heatmaps, and other visual analysis [2, 3, 4, 5, 6]. Our contributions can be summarized as follows: we leverage a body of previous contributions encompassing steering algorithms, crowd analysis and optimizations into an initial prototype a crowd-aware environment design tool. CODE is the first computational environment design tool that integrates dynamic crowd flow simulation, analysis, and automatic crowd-aware optimization of the environment; Second, to facilitate the user-in-the-loop authoring method, we propose an adaptive mesh refinement (AMR) approach to quickly guide the exploration of optimal placements of environment elements; Lastly, we present experiments to analyze the effectiveness of this approach including performance evaluation and comparison with manual authoring. We focus on the evaluation of density-critical portions of real world environments such as bottlenecks and egress points. Under serious scenarios, such as evacuation, the optimal crowd flow-density relationships between crowd and environment are crucial. Thus, while CODE may be applicable in other design spaces, we evaluate our approach under these conditions.

Related Work

Architectural Optimization. There is an established and growing interest in the use of architectural optimization to explore design spaces and provide optimal solutions with respect to problem criteria [7, 8].

Architectural optimization solutions generate new layouts or topologies for structures with respect to objective and/or subjective criteria, while providing a trade-off between automation and author precision. Data driven approaches [9] learn layout configurations from a database of prior architecture design constrained to a particular design space (e.g., residential homes). Another approach is to model objectives as relationships between features. Design objectives can be modelled as forces applied to a physical features to generate layout designs automatically [10]. Hierarchical and spatial relationships of furniture objects can be modelled to produce realistic human designer-like configurations with respect to their computed visibility and accessibility [11].

The modelling of physical phenomenon is most related to our approach to environment fitness. These methods may incorporate simulation of sunlight [12], materials, subsequent energy savings [13], or even acoustics [14]. The most closely related to our work is the optimization of egress environments using crowds [15, 16]. This work does not incorporate the user-in-the-loop processes and is presented as analyses of the affects of egress obstacles and shapes on crowd evacuation.

Since subjective criteria are difficult or impossible to quantify, many tools select an

optimization scheme to meet objective criteria then take a human-in-the-loop interactive approach to the subjective. The tools combine the aforementioned derivations for fitness with user guided optimization processes [17, 18, 19].

Crowd Simulation, Analysis, and Optimization. The maturity of research in simulating crowd dynamics [1, 20] has resulted in a wide variety of approaches including social forces [21] and predictive models [22, 23]. There has been a growing recent trend to use statistical analysis in the evaluation and analysis of dense crowd simulations. The work in [24, 25] measures the ability of a steering algorithm to emulate the behaviour of a real crowd dataset by measuring its divergence from ground truth. Crowd optimization techniques [26, 27] automatically fit the parameters of a crowd to meet different criteria, or even match real crowds.

Our Work. We leverage the wealth of research in crowd simulation to present a computational design tool for environments that uses features extracted from an underlying crowd simulator as criteria for architectural optimization.

Scenario Definition and Optimization

`CODE` allows designers to author scenarios that specify an environment layout along with different crowd configurations. Formally, we define a scenario $\mathbf{S} = \langle \mathbf{E}, \mathbf{C} \rangle$ that contains the specification of the environment $\mathbf{E}$ and the crowd $\mathbf{C}$. A crowd $\mathbf{C} = \langle \mathbf{A}, \mathbf{g}, \mathbf{P} \rangle$ defines a collection of agents $\mathbf{A}$ their desired goals $\mathbf{g}$ and the parameters $\mathbf{P}$ of the crowd simulator.

Every environment element $\mathbf{e} \in \mathbf{E}$ is defined as $\mathbf{e} = \langle \mathbf{p}, \mathbf{b}, s, \mathbf{r} \rangle$ using its position $\mathbf{p}$ bounds $\mathbf{b}$ and element shape s (e.g., walls, pillars, or obstacles). Additionally, the author may specify a region $\mathbf{r}$ which implies a constraint on the parametrized position of a set of $\mathbf{e}$ which may be optimized to lie anywhere within $\mathbf{r}$. If no region is specified, the element is considered a static element of the environment. For clarity, we modify the scenario definition to explicitly separate parametrized elements from the static elements $\mathbf{S} = \langle \mathbf{E}_f \cup \mathbf{E}_p, \mathbf{C} \rangle$ where $\mathbf{E}_f$ and $\mathbf{E}_p$ are the fixed and the parametrized elements in the environment design respectively.

Scenario Design

CODE allows designers to quickly design environments by providing controls for defining architectural layouts as well as constraint regions for optimizeable environment elements. Designers may specify crowd configurations by placing agents and their desired goals in the environment with control over defining groups and crowd densities. Furthermore, the designer may choose which steering simulator to use, as well as specifying a desired behaviour for the crowd. We currently support the following simulators: (a) **ORCA**: [22] uses reciprocal velocity obstacles for goal-directed collision avoidance, and (b) **SF**: [21] uses hypothetical social forces for resolving collisions between interacting agents in dense crowds. The system is predicated on a modular steering simulator and crowd evaluation suite and may be extended to support other simulators or methods. The addition of evacuation planning, group behaviour, and crowd coordination is outside the scope of this paper.

Optimization Formulation

Given a designer-authored scenario $\mathbf{S} = \langle \mathbf{E}_f \cup \mathbf{E}_p, \mathbf{C} \rangle$ our aim is to automatically generate a configuration of the environment elements $\mathbf{e} \in \mathbf{E}_p$ that optimizes the flow of the crowd under time sensitive conditions such as evacuation, while meeting constraints, imposed by the designer, on the valid regions where environment elements may be placed. This is formulated as the following optimization problem:

$$\begin{aligned} \mathbf{E}_p^{opt} &= \arg\max_{\mathbf{E}_p} \sum_{\mathbf{C} \in \mathbf{C}} f_{\mathbf{C}} \\ \text{s.t.} \quad & \mathbf{p} \in \mathbf{r} \;\; \forall \mathbf{e} \in \mathbf{E}_p \end{aligned} \tag{1}$$

In which we consider the simulation of a crowd $\mathbf{C}$ where $\mathbf{A_c} \subseteq \mathbf{A}$ reach their destination $\mathbf{g}$ within a maximum time threshold t_{sim} set at a conservative limit. Let $t_{\mathbf{a}}^f$ be the time for the last agent in $\mathbf{A_c}$ to reach its destination. The crowd flow $f_{\mathbf{C}}$ is computed as follows:

$$f_{\mathbf{C}} = \frac{|\mathbf{A_c}|}{t_{\mathbf{a}}^f} \tag{2}$$

This measures crowd flow as the average rate at which agents complete their goal $\mathbf{g}$ or the average agent flow at a goal area. Our formulation works well for degenerate cases, such as agents not completing their goals $\mathbf{A_{nc}} = \mathbf{A} \setminus \mathbf{A_c}$ when $|\mathbf{A_{nc}}| > 0 \wedge t_{\mathbf{a}}^f = t_{sim}$ and the numerator is reduced or denominator is inflated in the flow function. However, more complex formulations for crowd flow [28] may easily be used depending on the needs of the application.

Adaptive Mesh Refinement for Interactive Optimization

We apply an Adaptive Mesh Refinement (AMR) approach to discretize and adaptively sample the optimization search space of building layout optimization - a typically non-convex and continuous problem. AMR is a technique commonly used in the simulation of turbulent hydrodynamics [29], which we consider an isomorphic analogy to dense crowd simulations in arbitrary environments. The central idea of AMR to discretize the sampling space at finer granularity in areas of interest as defined by some heuristic. The heuristic guides mesh adaptation at each iteration to include, or enhance, information captured by the underlying data and affords the automatic pruning of areas which may be of low importance.

Assuming the isomorphic analogy to hydrodynamics, we apply the AMR technique to crowd aware building layout optimization. AMR affords tunable optimization results for crowd aware design of environments through subdivision depth and heuristic threshold. Based on results from previous literature [30], our granularity heuristic is crowd density, which focuses on sampling in dense areas of crowd flow. Furthermore, to meet the requirements of real time applications and iterative feedback in out proof-of-concept system, our algorithm employs a quadtree as the subdivision mesh. A quadtree is a simple yet effective structure that is fast to adapt, or subdivide, and affords easy tuning for the speed-granularity trade-off.

An illustration of our algorithm can be seen in Figure 2. The algorithm, which we refer to as Iterative Selection AMR with Backtracking (ISAB), is expressed in pseudocode

in Algorithm 1. The process, implemented in the CODE software system, can be seen in Figure 1. For a set of optimizable environment elements $\mathbf{e} \in \mathbf{E}_p$ the designer can specify a region $\mathbf{r}$ of valid locations where the elements might be placed in order to optimize the flow of the crowd. ISAB iterates through all parametrized environment elements, placing one in each iteration. At each iteration the **ComputeHistogram** function produces a histogram H and a resultant flow value $f_{\mathbf{C}}^{d}$ by executing a simulation the scenario $\mathbf{S}$ without the current element. This histogram is then used to adapt a space-partitioning structure, or mesh M. In this paper, the CODE system employs a quad tree for M but any space-partitioning structure may be used. The nodes n of this mesh, the centroids of the quad tree leafs, are used as sample points at which the **PlaceElement** function reinserts and moves the current element to produce a modified scenario $\mathbf{S}'$ while invalidating intersecting placements. The scenario $\mathbf{S}'$ is then simulated, and the flow value at the current node $\mathbf{F}$ (n) is stored. The index of $\mathbf{F}$ with the maximum value is then the optimal selection on this iteration. As a fail-safe for bad results, our algorithm incorporates backtracking functionality. This fail-safe ensures completeness in that we avoid being stuck in a local minima after an initial bad selection with probability ε_p. It should be noted, however, that the backtrack condition was not encountered in our experiments - more optimal flow values were always found. In larger more complex designs, the possibility to backtracking will reduce or avoid local minima and producing solutions with worse fitness. The amount of backtracking for any given element is also tunable (not shown here), in our current system this is set to two iterations.

Algorithm 1: `ISAB`

input : $\mathbf{E}_p$ - *optimizeable elements*

$\mathbf{E}_f$ - *fixed elements*

ε_p- *backtracking probability*

$\mathbf{S} \leftarrow \langle \mathbf{E}_f \cup \mathbf{E}_p, \mathbf{C} \rangle$- *scenario definition*

1 **foreach** $i \in |\mathbf{E}_p|$ **do**

2 $\quad \langle H, f^d_{\mathbf{C}} \rangle \leftarrow$ **ComputeHistogram(S)**

3 $\quad M \leftarrow$ **SubdivideMesh**(H)

4 $\quad \mathbf{F}\,[] \leftarrow -1$

5 $\quad$ **foreach** *Node* $n \in M$ **do**

6 $\quad\quad \mathbf{S}' \leftarrow$ **PlaceElement(S**, $\mathbf{E}_p[i]$, n**)**

7 $\quad\quad \mathbf{F}\,[n\,] \leftarrow$ **Simulate(S**$'$**)**

8 $\quad$ **end**

9 $\quad$ **if max(F)** $< f^d_{\mathbf{C}}$ *&* **rand()** $< \varepsilon_p$ **then**

10 $\quad\quad i \leftarrow i - 1$

11 $\quad$ **else**

12 $\quad\quad \mathbf{S} \leftarrow$ **PlaceElement(S**, $\mathbf{E}_p[i]$, *index(***max(F)**,**F***))*

13 $\quad$ **end**

14 **end**

Precomputed Flow Information

To understand how a crowd flows within a given scenario we pre-compute a two dimensional histogram describing the crowd density distribution in a given scenario. We generate this histogram using a simulation of the designer-authored current scenario state.

The histogram bins $H(\mathbf{p}_1, \mathbf{p}_2)$ store agent position counts over a series of simulation time steps at each grid location in the world $[\mathbf{p}_1, \mathbf{p}_2]$:

$$H(\mathbf{p}_1, \mathbf{p}_2) = \sum_{t \in t_{sim}} \sum_{\mathbf{a} \in \mathbf{A}} I(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_\mathbf{a}^t), \tag{3}$$

where the indicator function I increments a bin $H(\mathbf{p}_1, \mathbf{p}_2)$ if an agent's position at a sample time $\mathbf{p}_\mathbf{a}^t$ is within a histogram bin defined by upper left $\mathbf{p}_1$ and lower right $\mathbf{p}_2$ positions of the grid cell.

$$I(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_\mathbf{a}^t) = \begin{cases} 1 & \text{if } \mathbf{p}_\mathbf{a}^t \in [\mathbf{p}_1, \mathbf{p}_2] \\ 0 & \text{if else} \end{cases} \tag{4}$$

This histogram informs the adaptive procedure of the "importance" of a region by tracking the use of that region over time.

Adaptive Mesh Refinement

For obstacle features, whose parameter constraints enforce placement within a region, two dimensions are considered. As proof of concept and to meet the requirements of a real tie interactive user-in-the-loop system, a quad tree structure is employed as the sampling

mesh. The quad tree recursively subdivides the region $\mathbf{r}$ into finer quadrants given a suitable heuristic, increasing sampling resolution in regions as guided by this heuristic.

We adopt a subdivision heuristic which can be formalized as crowd flow density mass (the distribution of agents over time over $\mathbf{r}$), or the sum of the histogram values $h \in H$ weighted by the area of intersection between a node n represented using upper left $\mathbf{p}_1^n$ and lower right $\mathbf{p}_2^n$ points, and the area of the underlying bins h represented using the upper left $\mathbf{p}_1^h$ and lower right $\mathbf{p}_2^h$ points:

$$\mathbf{Mass}(n) = \sum_{h \in H} h \cdot \mathbf{Intersect}(\mathbf{A}(\mathbf{p}_1^h, \mathbf{p}_2^h), \mathbf{A}(\mathbf{p}_1^n, \mathbf{p}_2^n)). \tag{5}$$

Here **Intersect** is chosen by representation to be either line-rectangle or rectangle-rectangle intersection algorithms, and $\mathbf{A}(.)$ represents the spatial area of the bin. This allows us to focus sampling where the likelihood of impacting the crowd flow is significantly higher.

Evaluation

This section describes the experimental evaluation of the `ISAB` algorithm in two settings. First, the algorithm is compared with another offline optimization solution to asses performance and validity of the output. Second, the algorithm is exposed to participants in an effort asses the value of the approach in a realistic setting in Section: User Studies.

Apparatus and Materials

For each evaluation, a number of environmental elements, 1 and 2 pillars, are placed in the experimental scenarios described in this section. These scenarios represent common portions of environment designs which may incur turbulent crowd flow under high density, or panicked, conditions such as evacuations or crowd mergers.

To ensure fitness results generalize across a variety of initial crowd conditions, for each placement solution (performance comparison, `ISAB`, and participant), the subsequent authored scenario is simulated *ex post facto* 100 times with randomly chosen uniformly distributed crowd agents. We derive crowd flow performance from these simulations.

Uni-directional Hallway Egress: This scenario exhibits the basic features of a typical building egress point, Figure 4. The crowd begins within the hallway and moves toward the left egress point. The egress point serves as a representative bottleneck design found in some emergency exits.

Bi-directional Hallway: In this scenario, two crowds of agents must cross a hallway in opposite directions. The optimization region is defined at this central meeting point, Figure 5. The central meeting point results in the high density merger of the two crowds.

Bi-directional Hallway Side Egress: This modified version of the bi-directional hallway includes a centralized egress point. The optimization region is defined near this egress point, Figure 6. The centralized egress point results in a high density merger of two crowds during an egress and represents a potentially subtle bottleneck design.

Four-way Hallway Side Egress: In this scenario, four groups of agents must cross a hallway in opposite directions. The optimization region is defined in the central area, Figure 7. The central meeting point results in the high density merger of the four crowds.

Performance Experiment

To explore the performance of the `ISAB` method we compare it to the performance of CMA-ES, an evolutionary strategy optimization method used in non-convex problems. CMA-ES has been used previously in this domain as an offline method for optimizing architectural elements [30, 31, 32], and it serves as a state-of-the-art for analysis of sampling-based methods.

Results

Figure 8 shows the mean crowd flow and simulation costs with confidence intervals of CMA-ES and `ISAB`, based on a T-test. The expected crowd flow rates are relatively close to each other, and the 95% confidence intervals are very tight. The results generated by `ISAB` perform similarly (on par, close to, or better) to CMA-ES results. Moreover, Figure 8c shows that in terms of simulation costs (optimization fitness evaluations), `ISAB` greatly outperforms CMA-ES by requiring considerably less simulations. Therefore, `ISAB` can be useful for real-time interactive applications, providing results on par with CMA-ES in considerably less time.

User Studies

Two user studies, with 1 and 2 pillar placements, were performed to assess influence of the automated ISAB technique on authoring performance. The studies evaluate whether the automatic ISAB method outperforms user based manual placement of environment elements. Participants were asked to author optimal solutions for the four experimental scenarios using two tools: (1) CODE as a manual design tool; and (2) CODE with the automated ISAB technique. With both tools, the subject is supplied feedback on their current placement of environment elements based on crowd simulation. As a manual design tool, the subject can place the element and run simulations, but access to the ISAB technique is disabled. The scenarios have predefined agents with initial conditions and goals. All subjects were asked to carry out tasks on the same Desktop PC (Linux, 12GB RAM, AMD A 10-7800 Radeon R7, 12 Compute Cores, 3.5GHz). Subjects were given instructions on how the system works, interface elements, and the process of simulation, placement, and feedback.

Since the AMR approach may be repeatedly adapted (or subdivided), a preliminary mesh granularity experiment exploring quality versus performance revealed that a subdivision level between 2 and 4 of a quadtree space-partitioning data structure provided results on par with participants' best position selections with respect to flow and time, see Figure 3. The average of 10 participants' best selections out of 3 attempts is compared with ISAB at a max subdivision level ($l_{max} = 1 - 4$) bounded by $4^{l_{max}}$.

Independent variables. The authoring methods were the primary independent variables. These were the manual approach and the `ISAB` method.

Dependent variables. A number of metrics were captured during the use of the system. Those used to assess outcome performance are: (1) The final crowd flow rate given the participant's choice of placement for each scenario and method; and (2) The time in seconds, t_a needed to author a scenario, for each method.

1-Pillar Placement Study

Each subject was asked to optimize the placement of a single pillar with respect to crowd flow in the four experimental scenarios.

Results

Figure 9 shows the results of a T-test based on the single pillar placement user study based. The bars show the average results, and the error bars show the 95% confidence intervals. We analysed three separate cases, first, average and best selection Manual performance to compare participants' pillar placements with the `ISAB` assisted method. Note that a 0.5 unit difference in average crowd flow equates to approximately 30 more persons per minute being evacuated, resulting in significant improvement for evacuation tasks.

Figure 9a shows that, compared to the subject's first selection, `ISAB` average flow is significantly better in almost all scenarios. `ISAB` performance in the bi-directional hallway is as good as Manual, and in the four-way hallway `ISAB` average flow is slightly lower

than Manual but the difference is not statistically significant. ISAB also performs statistically significantly better than average performance of Manual placements, i.e. average flow achieved by the subject over three trials. Also, in almost all scenarios the best performance of each subject, i.e. maximum flow over three trials, is not significantly different from ISAB. Figure 9b, shows that average time of completion for ISAB is significantly lower than average total Manual pillar placement time of subjects.

In conclusion, the results confirm that the assistance of ISAB may improve average crowd flow in less time than users. In other words, participants who manually designed their environments achieved ISAB performance only after several iterations. In some cases, the Manual pillar placement has some advantages in performance. We hypothesize that this is due to single element selections with readily apparent, high value solutions. It is important to note that, while participants may perform better than ISAB on some scenarios, on average Manual performance is significantly lower than ISAB. This occasional result motivates our second user study with multiple pillars.

2-Pillar Placement Study

Each subject was asked to optimize the placement of two pillars with respect to crowd flow in the uni-directional scenario. This experiment assesses the relative performance of ISAB in a combinatorial design decision task.

Results

Figure 10 depicts how subjects had more difficulty with the more complex task of placing two pillars when comparing their performance to ISAB. As expected, when placement decisions become combinatorial subjects struggle to find optimal solutions. Figure 11 illustrates the placement of 2 pillars by the subjects and ISAB using **SF** in the uni-directional benchmark. Crowd flow values reveal that ISAB outperforms participant selections. ISAB finds solutions similar in performance to the 1-pillar user study, while participants manually perform worse and with more variability.

Conclusion

We have presented CODE: a crowd-aware computational design tool for designing environment building layouts that integrates crowd simulation, analysis, and environment optimization. CODE has potential as an enabling technology for practitioners that are involved with the design and layout of buildings, such as architects, fire marshals, etc. Our 2-pillar placement study suggests that as design decisions become more complex, tools like CODE provide significant improvements over intuitive designs. We will continue to work closely with experts in the area to improve and enhance the feature set of CODE, by conducting a usability study amongst experts in the future.

The use of one type of space-partitioning structure for AMR is a limitation of our current analyses. As well, region-only constraints for the environment parametrizations, and a lack

of integration with gold standard architectural design software limit the types of environment we can currently use our tool on. These particular limitations are to be addressed in future work. Our analysis is limited to flow-density critical portions of environments under stress, i.e. egress points in evacuation scenarios. However, the system supports fine tuned design of both the environment and the scenario including a crowd's behaviour, initial configurations, and goals. As such, a designer may optimize under a variety of conditions. In future work we plan to address the problem space of generalizable analysis of environments using dynamic synthetic crowds.

Acknowledgements

The authors wish to acknowledge the support of the NSERC Discovery Grant Program.

References

[1] Nuria Pelechano, Jan M. Allbeck, and Norman I. Badler. *Virtual Crowds: Methods, Simulation, and Control*. Morgan & Claypool Publishers, 2008.

[2] Shawn Singh, Mubbasir Kapadia, Petros Faloutsos, and Glenn Reinman. An open framework for developing, evaluating, and sharing steering algorithms. In *MiG*, pages 158–169. Springer-Verlag, 2009.

[3] Sean Curtis, Andrew Best, and Dinesh Manocha. Menge: A modular framework for simulating crowd movement. http://gamma.cs.unc.edu/Menge/, 2015.

[4] Golaem. Golaem crowd. http://www.golaem.com/crowd, 2016.

[5] Oasys Software. Massmotion: Crowd simulation and pedestrian modelling software. http://www.oasys-software.com/products/engineering/massmotion.html, 2016.

[6] INCONTROL Simulation Solutions. Pedestrian dynamics crowd simulation software. http://www.pedestrian-dynamics.com/, 2016.

[7] Philippe Block, Jan Knippers, Niloy J Mitra, and Wenping Wang. *Advances in Architectural Geometry 2014*. Springer, 2014.

[8] Helmut Pottmann, Michael Eigensatz, Amir Vaxman, and Johannes Wallner. Architectural geometry. *Computers & Graphics*, 47:145–164, 2015.

[9] Paul Merrell, Eric Schkufza, and Vladlen Koltun. Computer-generated residential building layouts. *ACM Transactions on Graphics*, 29(6):181:1–181:12, December 2010.

[10] Scott A Arvin and Donald H House. Modeling architectural design objectives in physically based space planning. *Automation in Construction*, 11(2):213–225, 2002.

[11] Lap-Fai Yu, Sai Kit Yeung, Chi-Keung Tang, Demetri Terzopoulos, Tony F. Chan, and Stanley Osher. Make it home: automatic optimization of furniture arrangement. *ACM Transactions on Graphics*, 30(4):86, 2011.

[12] Hwang Yi and Yun Kyu Yi. Performance based architectural design optimization: Automated 3d space layout using simulated annealing. In *ASHRAE/IBPSA-USA Building Simulation Conference*, 2014.

[13] Luisa Gama Caldas and Leslie K Norford. A design optimization tool based on a genetic algorithm. *Automation in construction*, 11(2):173–184, 2002.

[14] Alban Bassuet, Dave Rife, and Luca Dellatorre. Computational and optimization design in geometric acoustics. *Building Acoustics*, 21(1):75–86, 2014.

[15] Anders Johansson and Dirk Helbing. Pedestrian flow optimization with a genetic algorithm based on boolean grids. In *Pedestrian and evacuation dynamics 2005*, pages 267–272. Springer, 2007.

[16] Li Jiang, Jingyu Li, Chao Shen, Sicong Yang, and Zhangang Han. Obstacle optimization for panic flow-reducing the tangential momentum increases the escape speed. *PloS one*, 9(12):e115463, 2014.

[17] Xing Shi and Wenjie Yang. Performance-driven architectural design and optimization technique from a perspective of architects. *Automation in Construction*, 32:125–135, 2013.

[18] Michela Turrin, Peter von Buelow, and Rudi Stouffs. Design explorations of performance driven geometry in architectural design using parametric modeling and genetic algorithms. *Advanced Engineering Informatics*, 25(4):656–675, 2011.

[19] Jeremy Michalek and Panos Papalambros. Interactive design optimization of architectural layouts. *Engineering Optimization*, 34(5):485–501, 2002.

[20] Daniel Thalmann and Soraia Raupp Musse. *Crowd Simulation, Second Edition*. Springer, 2013.

[21] Dirk Helbing, Illes Farkas, and Tamas Vicsek. Simulating dynamical features of escape panic. *Nature*, 407(6803):487–490, 2000.

[22] Jur van den Berg, Stephen J. Guy, Ming Lin, and Dinesh Manocha. Reciprocal n-body collision avoidance. In *Robotics Research*, volume 70, pages 3–19. 2011.

[23] Shawn Singh, Mubbasir Kapadia, Billy Hewlett, Glenn Reinman, and Petros Faloutsos. A modular framework for adaptive agent-based steering. In *ACM I3D*, pages 141–150, 2011.

[24] Stephen J Guy, Jur van den Berg, Wenxi Liu, Rynson Lau, Ming C Lin, and Dinesh Manocha. A statistical similarity measure for aggregate crowd dynamics. *ACM Transactions on Graphics*, 31(6):190, 2012.

[25] Julien Pettré, Jan Ondřej, Anne-Hélène Olivier, Armel Cretual, and Stéphane Donikian. Experiment-based modeling, simulation and validation of interactions between virtual walkers. In *ACM SIGGRAPH/EG SCA*, pages 189–198, 2009.

[26] David Wolinski, Stephen Guy, Anne-Helene Olivier, Ming Lin, Dinesh Manocha, and Julien Pettré. Parameter estimation and comparative evaluation of crowd simulations. *Computer Graphics Forum*, 33(2), 2014.

[27] Glen Berseth, Mubbasir Kapadia, Brandon Haworth, and Petros Faloutsos. SteerFit: Automated Parameter Fitting for Steering Algorithms. In Vladlen Koltun and Eftychios Sifakis, editors, *ACM SIGGRAPH/EG SCA*, pages 113–122, 2014.

[28] Dirk Helbing, Lubos Buzna, Anders Johansson, and Torsten Werner. Self-organized pedestrian crowd dynamics: Experiments, simulations, and design solutions. *Transportation Science*, 39(1):1–24, 2005.

[29] Marsha J Berger and Phillip Colella. Local adaptive mesh refinement for shock hydrodynamics. *Journal of computational Physics*, 82(1):64–84, 1989.

[30] Brandon Haworth, Muhammad Usman, Glen Berseth, Mubbasir Kapadia, and Petros Faloutsos. Evaluating and optimizing level of service for crowd evacuations. In *ACM SIGGRAPH MiG*, pages 91–96. ACM, 2015.

[31] Glen Berseth, Muhammad Usman, Brandon Haworth, Mubbasir Kapadia, and Petros Faloutsos. Environment optimization for crowd evacuation. *Computer Animation and Virtual Worlds*, 26(3–4):377–386, 2015.

[32] Glen Berseth, M Brandon Haworth, Mubbasir Kapadia, and Petros Faloutsos. Characterizing and optimizing game level difficulty. In *ACM SIGGRAPH MiG*, pages 153–160. ACM, 2014.

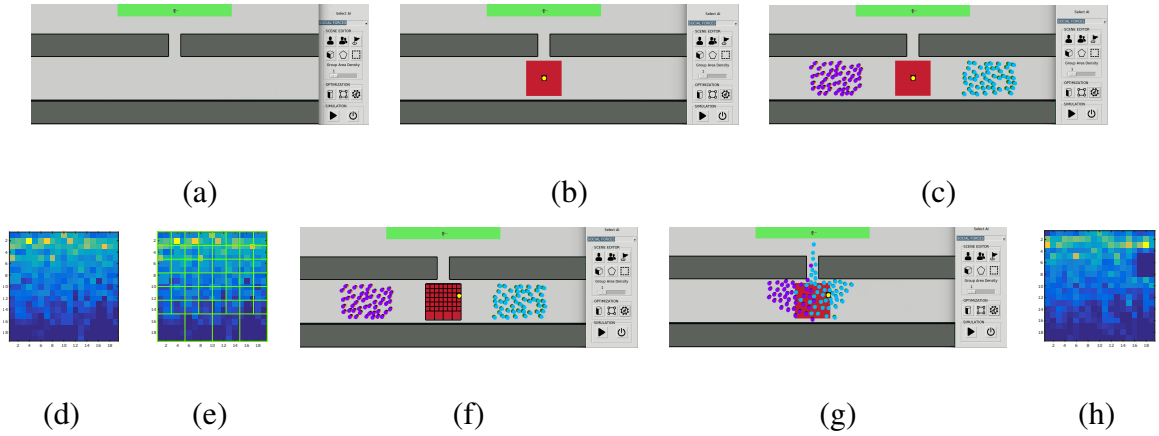
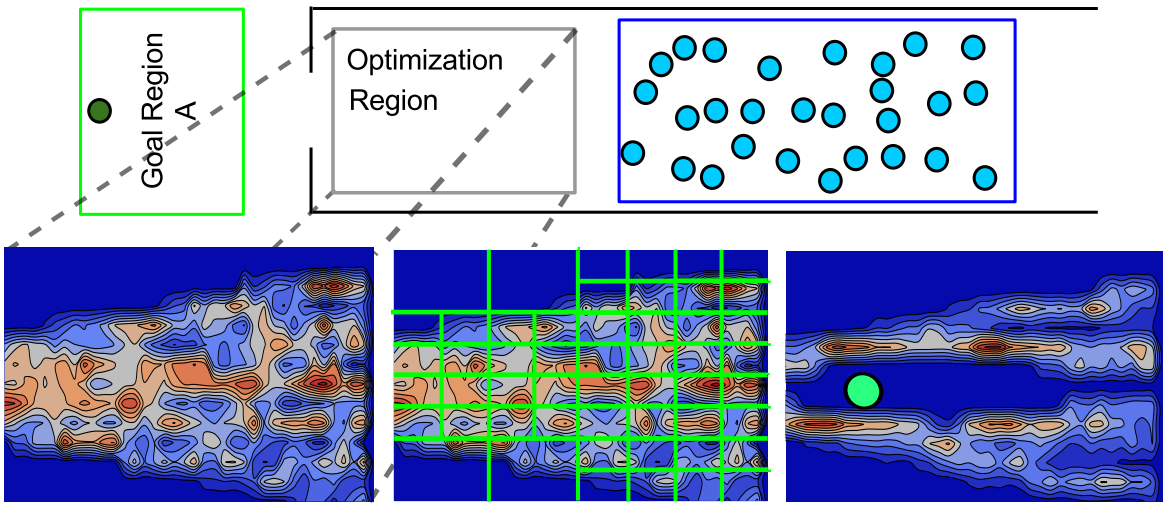


Figure 1: (a) The initial scenario with walls (dark grey) and target location (bright green). (b) Addition of the optimization region (red) and initial best guess for the pillar location (yellow). (c) Addition of crowds to the scenario. (d) A heat map of the flow in the optimization region for the scenario in (c). (e) The computed `ISAB` optimal pillar location. (f) The `AMR` visualized in the scenario. (g) The new crowd simulation with increased flow and new flow as shown in (h).



(a) (b) (c)

Figure 2: (a) The crowd flow density in the optimization region. (b) The AMR for (a). (c) The best pillar position according to the AMR sampling, and new crowd flow.

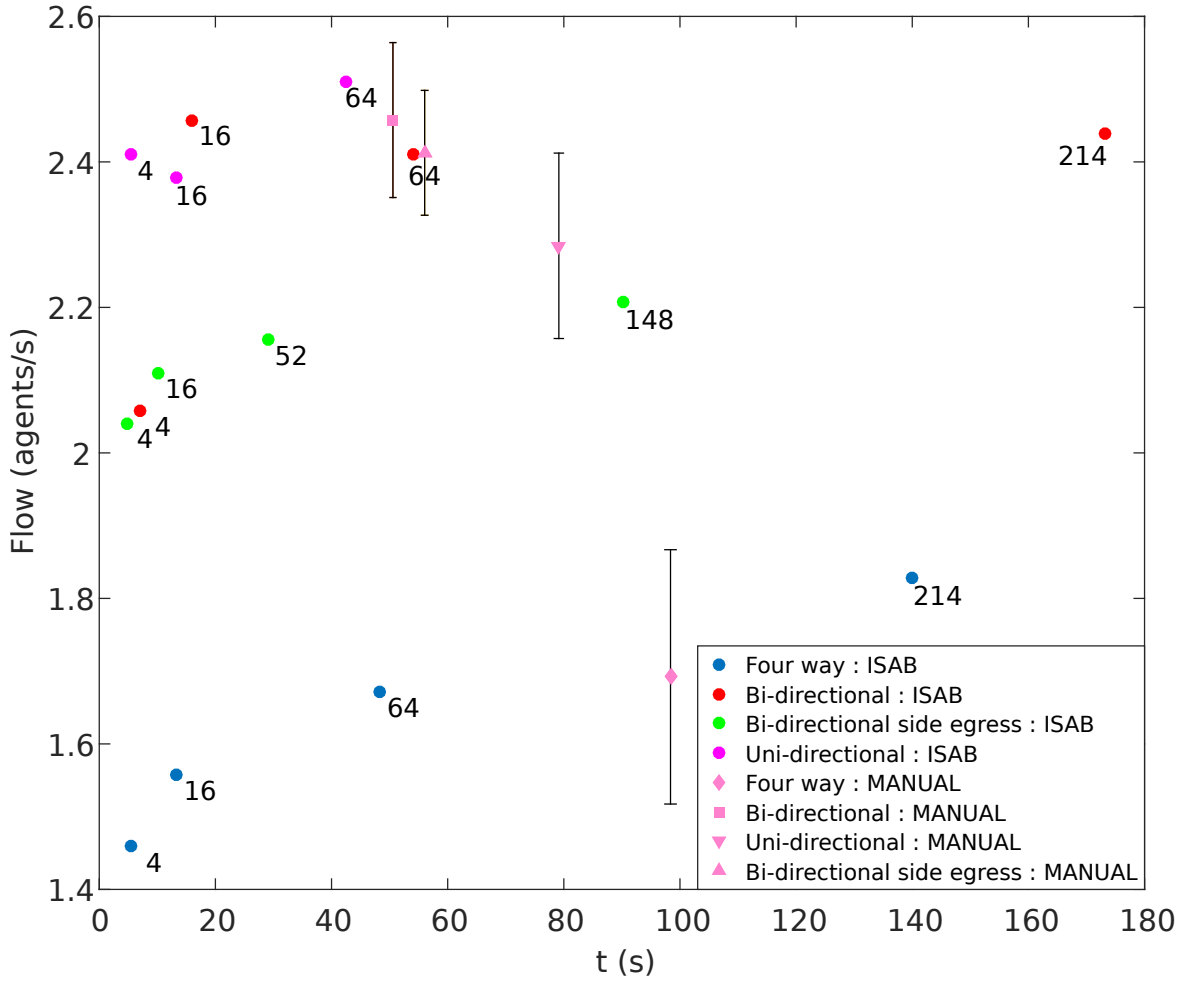


Figure 3: AMR mesh granularity experiment exploring quality (flow) versus performance (time) of ISAB and user based optimizations of pillar placements in different scenarios. The number of actual samples for each ISAB optimization is shown as the number next to the point. The error bars for the user based manual optimizations are shown.

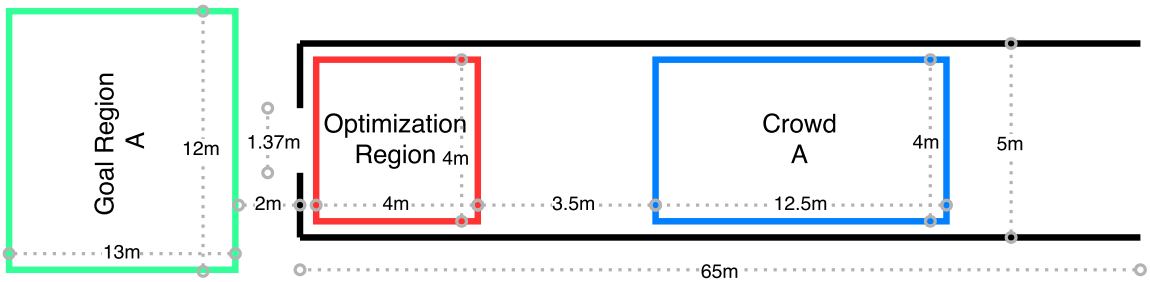


Figure 4: Egress scenario.

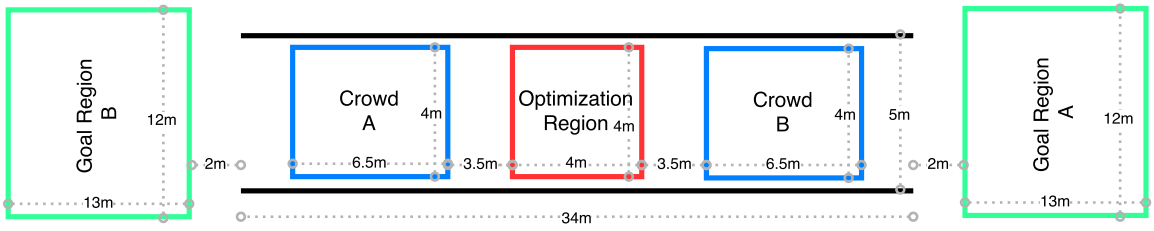


Figure 5: Bi-directional traffic in a hallway.

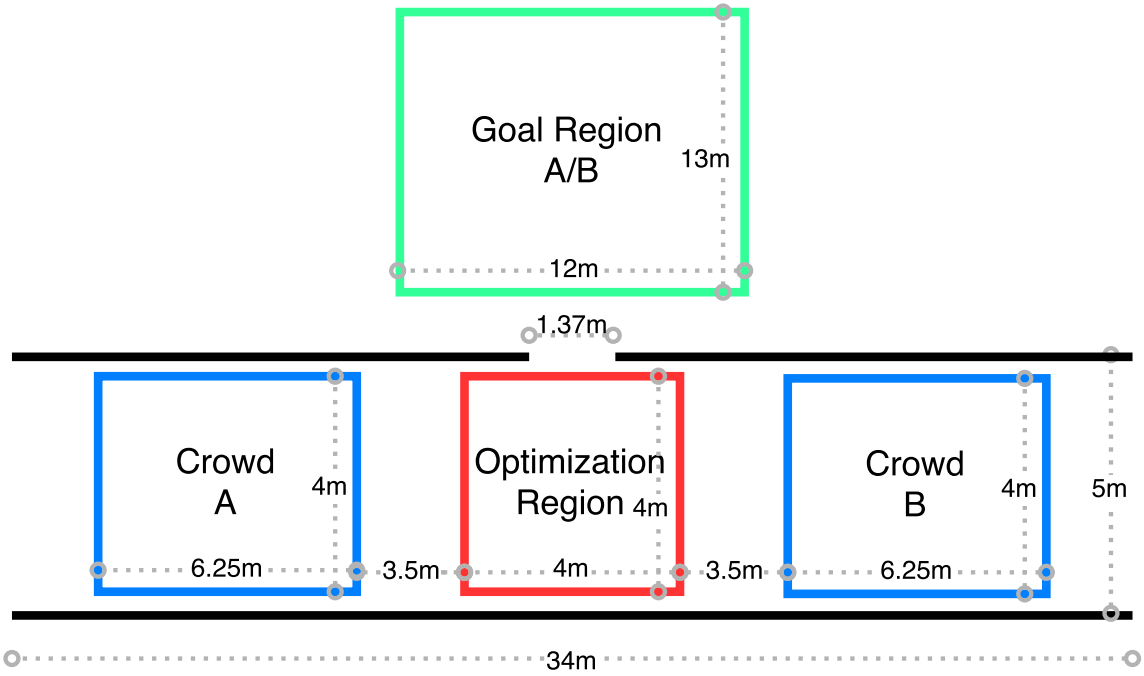


Figure 6: Bi-directional traffic with side egress.

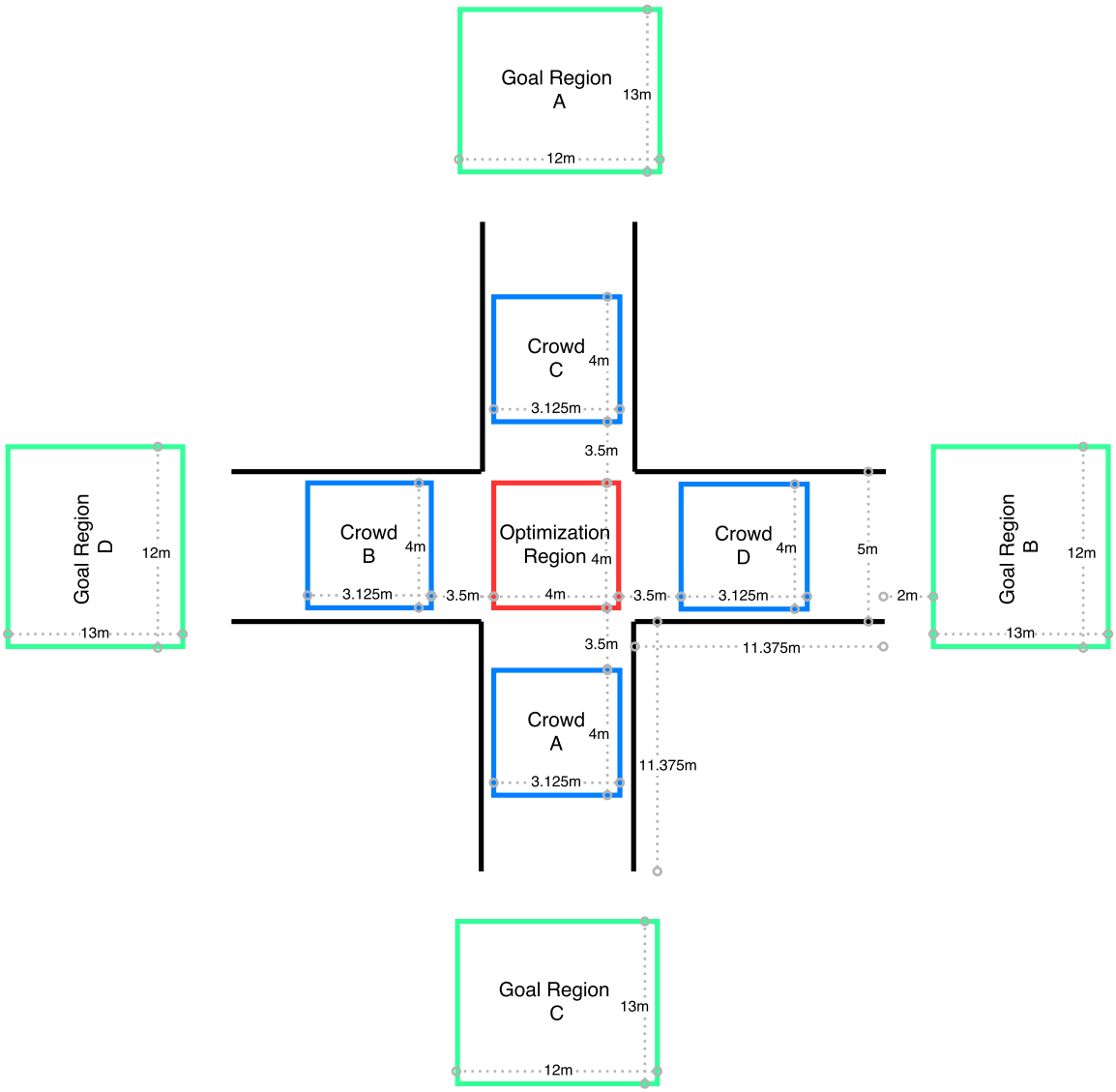
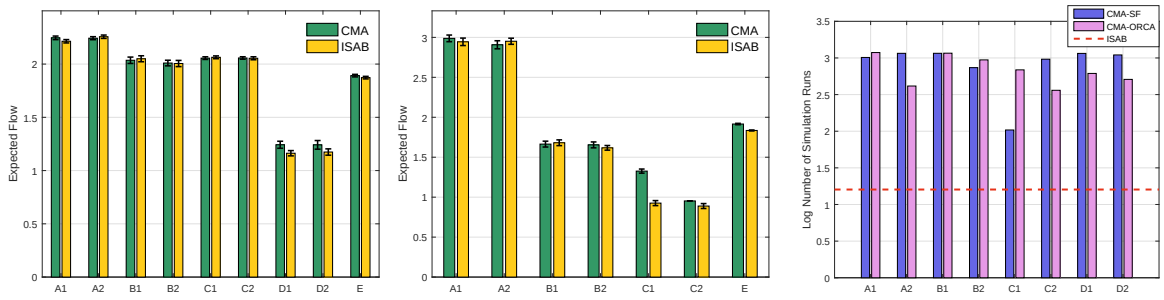
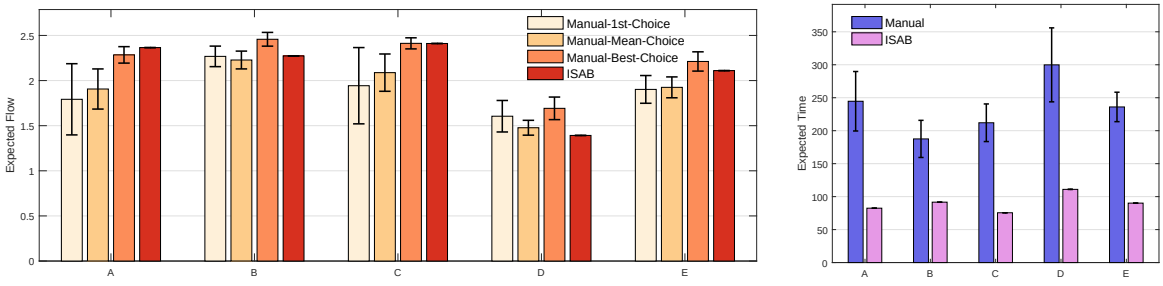


Figure 7: Four-way traffic in a hallway.



(a) (b) (c)

Figure 8: Comparison between `ISAB` and CMA-ES results, where each bar represents expected flow for each group with 95% confidence intervals shown. A is the uni-directional hallway scenario, B is the bi-directional hallway scenario, C is the bi-directional hallway side egress scenario, D is the four-way hallway and E is over all scenarios - the numbers 1 and 2 correspond to single and two pillar placements. (a) and (b) Expected flow results for **SF** and **ORCA** algorithms respectively (higher flow is better). (c) Log_{10} number of simulation runs till convergence (best fitness value).



(a) (b)

Figure 9: Average flow of Manual and `ISAB` assisted single pillar placement with 95% confidence intervals. The labels A to E represent scenarios as defined in Figure 8. (a) Comparison of the first, best, and mean manual choices with `ISAB` selections. (b) Expected time of completion of user average time and `ISAB`.

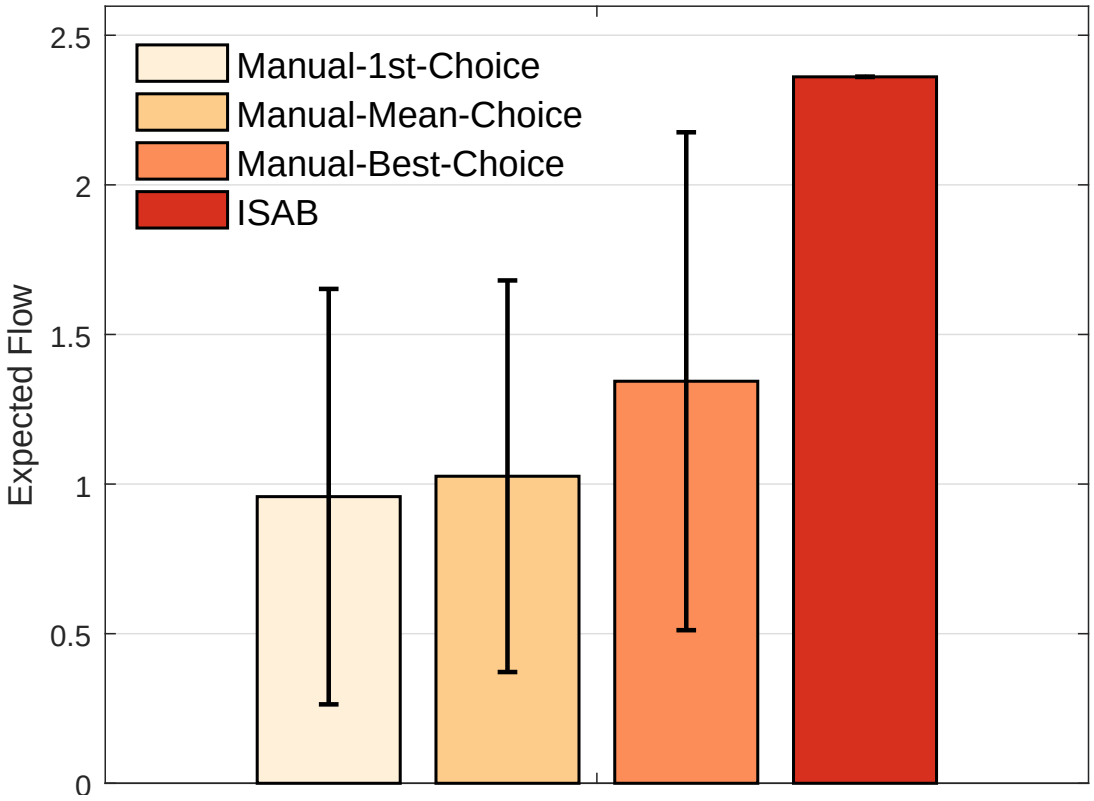


Figure 10: Average flow of Manual and ISAB assisted two pillar placement in uni-directional scenario, with the 95% confidence intervals.

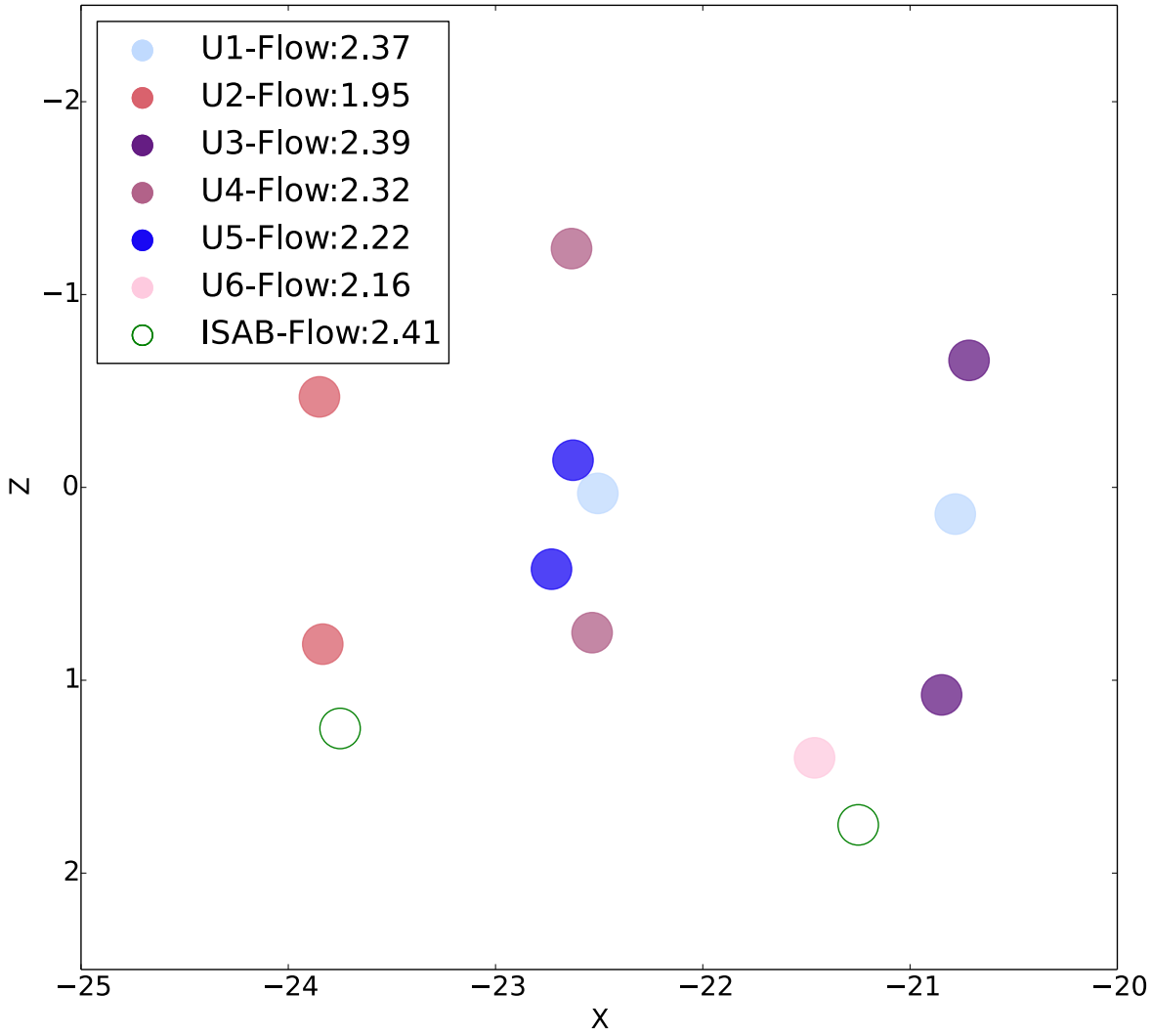


Figure 11: Manual and and `ISAB` placement of 2 pillars for **SF** in the uni-directional hallway environment. Crowd is moving from right-to-left.