Comparative Study Of Pedestrian Simulation Models Using Entropy-Based Evaluation

1 Introduction

In environments where multiple agents operate simultaneously, effective navigation and collision avoidance are essential for smooth and predictable interactions. This research focuses on simulating two autonomous agents that swap their positions while skillfully avoiding collisions using sophisticated computational techniques. The simulation utilizes three key models of agent navigation: the Reciprocal Velocity Obstacles (RVO) library, the Social Force Model (SFM), and the Universal Power Law (UPL) framework. This approach assesses their performance in different scenarios.

The RVO model simulates realistic movements and interactions of agents. It offers a strong framework for studying complex navigation behaviours. The Social Force Model (SFM) uses a physics-based method. It treats interactions between agents as forces that affect their paths. In contrast, the UPL framework [Karamouzas et al. 2014] uses probabilistic reasoning to model crowd dynamics. It offers a stochastic view of agent behaviour and decision-making.

A crucial part of this research is the computation of entropy, which measures uncertainty within the system. Entropy calculations based on the covariance matrices reveal insights into the reliability and predictability of the agents' state estimations. High entropy values indicate more uncertainty, especially during complex actions like collision avoidance and position swapping. Lower values show increased confidence in the agents' predicted states. This dual focus on precise state estimation and uncertainty measurement allows the simulation to effectively model agent interactions and evaluate the strength of the navigational strategies used. Specifically, we apply the entropy calculation method introduced by [Guy et al. 2012] to assess the accuracy of the simulators.

The implementation shows the smooth interaction between the simulation models and the EnKS-driven state estimation. By reading and processing real-world data inputs, initialising ensemble states, and continuously updating agent states through simulation steps, the research creates a solid framework for studying autonomous agent behaviours. Additionally, using Maximum Likelihood Estimation (MLE) techniques ensures the regular refinement of covariance matrices, preserving the integrity and stability of entropy calculations over time.

Overall, this research offers a solid platform for examining the dynamics of autonomous agents in collision-prone situations. By carefully integrating velocity obstacle algorithms, ensemble-based state estimation, and uncertainty measurement, the simulation experiments provide valuable insights into the effectiveness of navigational strategies and the mechanics behind reliable agent interactions in complex environments. By comparing the RVO, SFM, and UPL models, this work enhances our understanding of their respective strengths and weaknesses, paving the way for more effective and robust multi-agent systems in dynamic settings.

2 Literature review

2.1 Entropy Based Evaluation of A Simulator

We have employed the framework developed by [Karamouzas et al. 2018] to evaluate the fidelity of crowd simulation algorithms. Building upon the original entropy formulation proposed by [Guy et al. 2012], this approach quantifies the likelihood that a simulation reproduces real-world pedestrian trajectories. In its original form, the entropy metric computes a single, global accuracy score for each simulation method over the entire dataset. While useful, this approach assumes a fixed number of agents and lacks the granularity to identify localised or transient simulation errors.

The modified metric calculates entropy values for each agent, allowing for dynamic accuracy estimation. Each agent acts as a central observer in localized simulations, enabling the inclusion of agents entering or leaving the scenario over time, which is common in real-world datasets. The entropy is calculated using short, sliding time windows (e.g., 5 seconds) to capture temporary differences between simulated and actual paths. The mathematical formulation of the modified entropy for an agent over a given time window τ is:

$$E(\widetilde{Q}) = \frac{1}{2}\log\left((2\pi e)^d \det(\widetilde{Q})\right), \tag{1}$$

where $\widetilde{Q}$ is the estimated noise covariance matrix reflecting the simulation error, and d is the dimensionality of the agent's state, typically 2 in the case of 2D position data.

The computation of this entropy metric relies on an iterative pipeline involving the Ensemble Kalman Smoother (EnKS) and Maximum Likelihood Estimation (MLE). The EnKS is used to infer the most probable internal state of each agent at every time step using Bayesian filtering. This involves a prediction step, where the agent's state is propagated using the simulation function f, followed by a correction step using observed trajectory data. The ensemble approach runs multiple stochastic simulations (e.g., 2000 ensemble members), each perturbed with Gaussian noise sampled from $\widetilde{Q}$, to build a probabilistic estimate of the agent's state.

Subsequently, the MLE step estimates the optimal value of $\widetilde{Q}$ that best explains the observed data. This is accomplished by computing the residuals between the EnKS-predicted and actual observations, and iteratively updating $\widetilde{Q}$ using an Expectation-Maximization (EM) algorithm. We assume that the relation between the crowd state $\mathbf{a}_k$ and the data $\mathbf{z}_k$ available of the crowd at time k is given by a known function h. The relevant models for state prediction and observation are given by:

$$\mathbf{a}_k = f(\mathbf{a}_{k-1}) + \mathcal{N}(0, \widetilde{Q}), \quad \mathbf{z}_k = \mathbf{h}(\mathbf{a}_k) + \mathcal{N}(0, R), \tag{2}$$

where $\mathbf{a}_k$ represents the agent's internal state, $\mathbf{z}_k$ is the observed state, $\widetilde{Q}$ is the process noise covariance, and R is the fixed observation noise covariance.

2.1.1 Contributions of the Modified Metric. The main difference between [Karamouzas et al. 2018] and [Guy et al. 2012] lies in

dividing the dataset into per-time-step prediction tasks. While [Guy et al. 2012] modelled the simulation state as a joint state across all agents within an entire frame (requiring a fixed number of agents throughout the dataset), [Karamouzas et al. 2018] instead estimates a time-varying value of Q independently for each agent. This modified entropy metric provides a per-agent per-time-based measure of simulation accuracy. It enables the analysis of:

- Accuracy variations across different crowd scenarios.
- The performance of different simulation methods in localised regions of the crowd.

This granular, agent-centric approach enhances the applicability of entropy metrics for evaluating complex and dynamic crowd behaviours.

2.2 Reciprocal Velocity Obstacles (RVO2) and ORCA Framework

The Reciprocal Velocity Obstacles (RVO2) framework, introduced by [Van Den Berg et al. 2011] in their work on *Optimal Reciprocal Collision Avoidance* (ORCA), provides a decentralised solution for multi-agent collision avoidance. This method addresses the challenge of reciprocal n-body collision avoidance, where agents independently compute collision-free velocities without explicit communication.

In RVO2, each agent constructs velocity obstacles to identify velocities that would cause collisions with other agents or static obstacles within a time horizon τ. The velocity obstacle is defined as

$$VO^{\tau}_{A|B} = \{\mathbf{v} \mid \exists t \in [0,\tau],\ t\mathbf{v} \in D(\mathbf{p}_B - \mathbf{p}_A, r_A + r_B)\}, \tag{3}$$

where $D(\mathbf{p}, r)$ denotes a disc centered at position $\mathbf{p}$ with radius r, representing the combined radii of the agents. This formulation captures all velocities for which the relative position between agents A and B would result in an overlap, and hence a collision, within the specified time horizon.

ORCA improves upon prior RVO methods by introducing the notion of reciprocal responsibility, where agents share responsibility for avoiding collisions. This is done by defining half-plane constraints in velocity space. Each agent adjusts its velocity by half the minimum required deviation to ensure safety. The permitted set of velocities for agent A, assuming agent B also adjusts accordingly, is given by

$$ORCA^{\tau}_{A|B} = \left\{\mathbf{v} \,\middle|\, \left(\mathbf{v} - \left(\mathbf{v}_A + \frac{\mathbf{u}}{2}\right)\right) \cdot \mathbf{n} \geq 0\right\}, \tag{4}$$

where $\mathbf{u}$ is the minimum vector needed to move the relative velocity out of the velocity obstacle, and $\mathbf{n}$ is the outward-facing normal to the boundary of the velocity obstacle at the point of minimal adjustment. This constraint defines a linear region in velocity space where agent A can safely operate.

To compute their next motion, agents solve low-dimensional linear programs to select velocities that are closest to their preferred directions while satisfying collision-avoidance constraints. In cases where no feasible solution exists, such as in dense scenarios, a three-dimensional linear program is solved to minimise constraint violations. The RVO2 framework is computationally efficient and scalable, supporting real-time performance for thousands of agents through parallelisation and spatial pruning strategies, such as ignoring distant agents. This framework has been successfully applied in crowd simulations and multi-robot systems, yielding smooth and oscillation-free trajectories.

The Optimal Reciprocal Collision Avoidance (ORCA) framework offers several advantages for multi-agent navigation. It guarantees collision-free motion under sufficient conditions, ensuring safety in densely populated environments without requiring inter-agent communication and by relying solely on observed positions and velocities. Additionally, it handles static obstacles by enforcing non-reciprocal constraints, prioritising obstacle avoidance over inter-agent reciprocity. However, ORCA assumes holonomic agents with perfect sensing—a limitation in real-world pedestrian applications where motion is non-holonomic and sensing is error-prone. This research leverages ORCA's strengths in scalability and computational efficiency while addressing its limitations through comparative analysis with anticipatory (UPL) and force-based (SFM) models, ultimately assessing their fidelity to human-like navigation in controlled scenarios.

2.3 Social Force Model for Pedestrian Dynamics

The Social Force Model (SFM), introduced by [Helbing and Molnar 1995], provides a foundational framework for understanding pedestrian dynamics by modelling human motion as the result of "social forces." These forces are conceptualised as internal motivations that drive individuals to perform specific actions, such as movement or avoidance, rather than direct physical interactions. The model has significantly influenced research in pedestrian and crowd dynamics, urban planning, and related fields.

At the core of the model lies the principle that each pedestrian adjusts their velocity to align with a preferred speed and direction. This adjustment is described by an acceleration force toward the desired velocity, expressed mathematically as:

$$\vec{F}^0_\alpha = \frac{1}{\tau_\alpha}\left(v^0_\alpha \vec{e}_\alpha - \vec{v}_\alpha\right), \tag{5}$$

where v^0_α is the desired speed, $\vec{e}_\alpha$ is the unit vector indicating the desired direction, $\vec{v}_\alpha$ is the current velocity of pedestrian α, and τ_α is the characteristic relaxation time determining how quickly the pedestrian adapts to the desired velocity.

In addition to this driving force, pedestrians interact with their environment through repulsive and, optionally, attractive forces. Repulsive interactions are used to maintain personal space and avoid collisions. These include interactions with other pedestrians, modelled via exponentially decaying functions based on distance, and with environmental boundaries such as walls and obstacles. Though attractive forces were not emphasised in the initial formulation, they can be incorporated to simulate behaviour such as movement toward landmarks, points of interest, or social groups.

Furthermore, the model incorporates perceptual limitations by considering that a pedestrian primarily reacts to stimuli within a limited field of view, typically around 200°. This reflects the empirical observation that individuals are less responsive to entities or obstacles behind them.

The Social Force Model has been extensively validated through computer simulations, which have consistently demonstrated its capacity to replicate key phenomena observed in pedestrian dynamics.

One such phenomenon is lane formation, where pedestrians in bidirectional flows spontaneously self-organise into lanes of uniform walking direction. This behaviour closely mirrors real-world crowd movement patterns. Additionally, the model effectively reproduces bottleneck oscillations, wherein pedestrian groups alternately pass through narrow passages in opposing directions—an oscillatory pattern frequently observed in empirical studies. Furthermore, the model captures the adaptive nature of collision avoidance, as individuals dynamically adjust their trajectories to maintain safe interpersonal distances.

Social Force Model (SFM) demonstrated its strength in linking psychological motivations (e.g., goal-directed movement, collision avoidance) to physical motion through a quantifiable framework, enabling empirical validation against real-world pedestrian dynamics and simulating emergent collective behaviours like lane formation. However, its assumption of uniform agent behaviour overlooks the heterogeneity observed in real-world interactions, potentially contributing to overly stable entropy profiles that underrepresent uncertainty during complex maneuvers[Zanlungo et al. 2011]. Additionally, its reliance on exponentially decaying repulsive forces simplifies anticipatory adjustments seen in human-like navigation, a gap highlighted when contrasted with models incorporating predictive metrics. The Social Force Model (SFM) aligns with this work's integration of Ensemble Kalman Smoother (EnKS) and Maximum Likelihood Estimation (MLE) by providing a quantitative, parameter-driven framework to bridge agent motivations (e.g., goal-seeking) with physical motion, enabling precise state estimation and covariance refinement via EnKS-MLE.

2.4 Universal Power Law Governing Pedestrian Interactions

Understanding pedestrian interactions is a critical aspect of modelling crowd dynamics and has been extensively studied using various theoretical and empirical approaches. Human motion in crowds often mirrors interacting particle systems, with individuals adjusting their movements to avoid collisions. Traditional models of pedestrian dynamics rely heavily on deterministic or empirical methods, but the exact nature of interaction forces has remained a challenging problem.

Early works, such as Helbing's Social Force Model (SFM) [Helbing and Molnar 1995], introduced a physics-inspired framework where pedestrians interact via repulsive forces based on their spatial separations. As mentioned earlier, these models successfully simulated basic crowd phenomena such as lane formation, bottleneck clogging, and group behavior in dense crowds. However, SFM and similar distance-based models struggled to capture the anticipatory nature of human interactions, which are influenced by future states rather than just present positions.

Cellular automata [Lubaś et al. 2016] and agent-based [Ben et al. 2013] approaches were also proposed, modeling pedestrians as discrete agents governed by local rules. While computationally efficient, these methods lacked generalizability to diverse scenarios and densities.

A significant advancement in pedestrian modelling was the recognition of *time-to-collision* (τ) as a key metric in the work of [Karamouzas et al. 2014]. Unlike distance-based interactions, τ captures the temporal aspect of a potential collision, quantifying how imminent it is based on the relative positions and velocities of two pedestrians. This anticipatory metric aligns well with human behaviour, as pedestrians adjust their trajectories to avoid future collisions rather than reacting solely to current proximity. [Karamouzas et al. 2014] introduced a novel statistical mechanics-based framework to analyze pedestrian interactions. Leveraging large datasets of human trajectories from sparse outdoor environments and dense bottleneck scenarios, the authors derive an empirical interaction law based on two key metrics:

Pair Distribution Function ($g(\tau)$). The pair distribution function quantifies the likelihood of two pedestrians having a time-to-collision τ, normalised by the expected likelihood of pedestrian separations if they moved independently, ignoring mutual interactions:

$$g(\tau) = \frac{P_{\text{obs}}(\tau)}{P_{\text{exp}}(\tau)}, \tag{6}$$

where $P_{\text{obs}}(\tau)$ is the observed probability density at time-to-collision τ, and $P_{\text{exp}}(\tau)$ is the expected probability density for non-interacting pedestrians at τ.

Interpretation:

- $g(\tau) \gg 1$: Indicates configurations with minimal interaction or cooperative behaviour.
- $g(\tau) \ll 1$: Reflects strong avoidance behaviour, as pedestrians adjust paths to prevent imminent collisions.

[Karamouzas et al. 2014] demonstrates that $g(\tau)$ collapses into a universal curve across diverse scenarios, underscoring the fundamental importance of τ as the key descriptor of pedestrian interactions.

Interaction Energy ($E(\tau)$). Using a Boltzmann-like relation, the interaction energy is derived as:

$$E(\tau) \propto \ln\left(\frac{1}{g(\tau)}\right). \tag{7}$$

This energy represents the cost of avoiding a collision and is central to the power-law model.

A UPL governing pedestrian interactions can be given by:

$$E(\tau) = \frac{k}{\tau^2} e^{-\tau/\tau_0}, \tag{8}$$

where k is a scaling constant, and τ_0 is the maximum interaction range. The power law exhibits:

- Quadratic Falloff: $E(\tau) \propto \frac{1}{\tau^2}$ for intermediate τ.
- Exponential Truncation: Interaction energy diminishes for large τ, reflecting finite human anticipatory behavior.

This interaction energy $E(\tau)$ enables the calculation of anticipatory forces acting on pedestrians. The force $\mathbf{F}$ is derived as the negative spatial gradient of $E(\tau)$:

$$\mathbf{F} = -\nabla_{\mathbf{r}} \left(\frac{k}{\tau^2} e^{-\tau/\tau_0}\right), \tag{9}$$

where:

- $\nabla_{\mathbf{r}}$: Spatial gradient with respect to the relative position $\mathbf{r}$.
- $\tau = \frac{\mathbf{r}\cdot\mathbf{v}}{|\mathbf{v}|^2}$: Time-to-collision, based on relative position $\mathbf{r}$ and velocity $\mathbf{v}$.

This force provides a predictive mechanism for collision avoidance, enabling pedestrians to adjust their trajectories in advance based on anticipated interactions.

The UPL framework complements this work's integration of the EnKS and MLE by formalising time-to-collision (τ) as a universal stochastic metric for anticipatory behaviour, directly informing entropy-based uncertainty quantification during collision avoidance. The empirically validated power-law interaction model provides a physics-grounded foundation for state estimation via EnKS, enabling robust covariance updates through MLE's iterative refinement. However, the model has limitations, particularly in high-density environments where close-range interactions and human reaction times become critical [Cowen 2014], and its assumption of near-equilibrium conditions may not fully capture highly dynamic or chaotic situations.

2.5 Empirical Analysis of the Lane Formation Process in Bidirectional Pedestrian Flow

[Feliciani and Nishinari 2016] investigates the phenomenon of self-organised lane formation in bidirectional pedestrian flows, with a specific focus on how varying flow ratios influence stability, efficiency, and collision avoidance. The experiments were conducted in a controlled environment, where a mock corridor of dimensions 10 m by 3.5 m was constructed with standardised buffer zones and waiting areas to regulate entry and exit. Male university students were assigned to walk in opposite directions, designating one side as the major flow (left-to-right) and the other as the minor flow (right-to-left). Four flow-ratio configurations were studied: a unidirectional setup (6/0) and three bidirectional arrangements—5/1, 4/2, and 3/3—corresponding to flow-ratios of 0.167, 0.333, and 0.500, respectively. The study identifies five distinct phases: unidirectional free-flow, lane formation, full bidirectional flow, lane dissolution, and final free-flow. Key findings include that balanced bidirectional flow (3/3) achieves the most stable lane structures but requires significant lateral movement during formation, making it prone to deadlocks at high densities.

This study on bidirectional flow highlights how crucial it is to conduct controlled experiments when trying to isolate specific factors, like flow ratio, and to understand complex behaviours, such as how lanes form in crowds. By applying robust experimental methods, our research makes significant strides in improving collision avoidance models in simulations. Specifically, we demonstrate how controlled environments can connect theoretical modeling approaches—such as UPL's power-law based anticipatory interactions—with empirical observations, using entropy as a proxy for pedestrian uncertainty. Collectively, these efforts highlight the synergy between physical experimentation and virtual simulation in furthering our understanding of pedestrian crowd dynamics.

3 Materials and Methods

The section outlines the simulation design, tools, procedures, and data analysis techniques used in the study. It details the implementation of three simulation models—RVO2, SFM, and UPL, alongside methodologies such as the EnKS and MLE for uncertainty quantification.

3.1 Entropy Computation

The accuracy of crowd simulation methods is evaluated using a modified entropy metric that quantifies how closely simulated trajectories match real human behaviour. The framework as per [Karamouzas et al. 2018] involves the following steps:

3.1.1 Data Decomposition. Instead of treating the entire frame as a single state, the method decomposes real-world crowd trajectories into per-agent, per-timestep prediction tasks. Each agent is treated as a *central agent* in a localised simulation that considers interactions with its nearby neighbours.

3.1.2 Ensemble Kalman Smoothing (EnKS). The *Ensemble Kalman Smoother* (EnKS) technique is used to estimate the state of each central agent over a short time window (e.g., 5 seconds). This involves:

- Prediction Step: Predicting the agent's next state using the chosen simulation method (the prediction function) while introducing noise to model simulation uncertainty. The noise is sampled from a zero-mean Gaussian distribution with covariance Q.
- Correction Step: Correcting the predicted state based on observed trajectory data. Multiple simulations (an ensemble) are run with different noise samples, resulting in an ensemble of predicted states.

3.1.3 Maximum Likelihood Estimation (MLE). After estimating agent states using EnKS, Maximum Likelihood Estimation (MLE) is employed to refine the noise covariance Q. The goal is to maximise the likelihood of observing the real trajectory data, given the simulation method:

$$Q = \text{argmax} \prod_k P(z_k | f(a_k, Q)), \tag{10}$$

where z_k represents the observed trajectory, $f(a_k, Q)$ is the simulation prediction model, and Q is iteratively refined.

The process iterates between EnKS and MLE until the covariance Q converges. To validate the assumption that the error between predicted and observed states is Gaussian, the Kolmogorov-Smirnov test is used.

3.1.4 Entropy Calculation. The converged covariance is run independently for each agent and generates a value of Q per central agent, per time step. To produce a final accuracy estimate, we accumulate values of Q over a time window and refer to the average of these Q for each agent as $\widetilde{Q}$. The per-agent, per time step accuracy score is then the entropy of this distribution:

$$E(\widetilde{Q}) = \frac{1}{2} \log \left((2\pi e)^d \det(\widetilde{Q}) \right), \tag{11}$$

where d is the dimensionality of $\widetilde{Q}$ (e.g., 2 for positional errors in x and y).

This score reflects the magnitude of noise required to align the simulation with real data. A smaller $\det(\widetilde{Q})$ indicates higher simulation accuracy, leading to a lower entropy score.

For the purpose of this study, we calculate this score for both the agents at each time step and average them to obtain a single entropy value per time step:

$$E_{\text{avg}}(\widetilde{Q}(t)) = \frac{1}{2} \left(E_1(\widetilde{Q}(t)) + E_2(\widetilde{Q}(t)) \right), \tag{12}$$

where $E_1(\widetilde{Q}(t))$ and $E_2(\widetilde{Q}(t))$ are the entropy scores for Agent 1 and Agent 2 at time step t, respectively.

3.2 Reciprocal Collision Avoidance (RVO2)

The Optimal Reciprocal Collision Avoidance (ORCA) framework computes collision-free velocities for multiple agents independently. We have explained the mathematical formulation and workflow of RVO2 below [Van Den Berg et al. 2011]:

3.2.1 Velocity Obstacle (VO) Definition. For two agents A and B, the velocity obstacle $VO^{\tau}_{A|B}$ represents the set of relative velocities that would cause a collision within time horizon τ:

$$VO^{\tau}_{A|B} = \{\mathbf{v} \mid \exists t \in [0,\tau] : t\mathbf{v} \in D(\mathbf{p}_B - \mathbf{p}_A, r_A + r_B)\} \tag{13}$$

- $\mathbf{p}_A, \mathbf{p}_B \in \mathbb{R}^2$: Current positions of agents A and B
- $r_A, r_B \in \mathbb{R}^+$: Radii of agents A and B
- $D(\mathbf{p}, r) = \{\mathbf{q} \mid \|\mathbf{q} - \mathbf{p}\| < r\}$: Open disc of radius r centered at $\mathbf{p}$
- $\tau \in \mathbb{R}^+$: Time window for collision avoidance

Geometrically, $VO^{\tau}_{A|B}$ is a truncated cone in velocity space.

3.2.2 Optimal Reciprocal Collision Avoidance (ORCA). For reciprocal avoidance, agents share responsibility by adjusting their velocities. The permitted velocities for A relative to B form a half-plane:

$$ORCA^{\tau}_{A|B} = \left\{\mathbf{v} \,\middle|\, \left(\mathbf{v} - \left(\mathbf{v}_A^{\text{opt}} + \frac{1}{2}\mathbf{u}\right)\right) \cdot \mathbf{n} \geq 0\right\} \tag{14}$$

- $\mathbf{v}_A^{\text{opt}}, \mathbf{v}_B^{\text{opt}} \in \mathbb{R}^2$: Optimization velocities (typically current velocities)
- $\mathbf{u} \in \mathbb{R}^2$: Minimum relative velocity adjustment:

$$\mathbf{u} = \left(\arg\min_{\mathbf{v} \in \partial VO^{\tau}_{A|B}} \|\mathbf{v} - (\mathbf{v}_A^{\text{opt}} - \mathbf{v}_B^{\text{opt}})\|\right) - (\mathbf{v}_A^{\text{opt}} - \mathbf{v}_B^{\text{opt}}) \tag{15}$$

- $\mathbf{n} \in \mathbb{R}^2$: Outward normal to $\partial VO^{\tau}_{A|B}$ at $(\mathbf{v}_A^{\text{opt}} - \mathbf{v}_B^{\text{opt}}) + \mathbf{u}$

3.2.3 Multi-Agent Collision Avoidance. For n agents, the permitted velocities for agent A are the intersection of all ORCA half-planes and the maximum speed constraint:

$$ORCA^{\tau}_{A} = D(\mathbf{0}, v_A^{\max}) \cap \bigcap_{B \neq A} ORCA^{\tau}_{A|B} \tag{16}$$

- $v_A^{\max} \in \mathbb{R}^+$: Maximum speed of agent A
- $D(\mathbf{0}, v_A^{\max})$: Velocity space disc ensuring $\|\mathbf{v}\| \leq v_A^{\max}$

3.2.4 Velocity Selection. Agent A selects the new velocity $\mathbf{v}_A^{\text{new}}$ closest to its preferred velocity $\mathbf{v}_A^{\text{pref}}$:

$$\mathbf{v}_A^{\text{new}} = \arg\min_{\mathbf{v} \in ORCA^{\tau}_{A}} \|\mathbf{v} - \mathbf{v}_A^{\text{pref}}\| \tag{17}$$

- $\mathbf{v}_A^{\text{pref}} \in \mathbb{R}^2$: Preferred velocity (e.g., goal-directed)

This is a convex optimization problem solved via linear programming (LP).

3.2.5 Handling Infeasibility in Dense Scenarios. If $ORCA^{\tau}_{A} = \emptyset$, a 3D LP minimizes the worst-case constraint violation:

$$\mathbf{v}_A^{\text{new}} = \arg\min_{\mathbf{v} \in D(\mathbf{0}, v_A^{\max})} \max_{B \neq A} d_{A|B}(\mathbf{v}) \tag{18}$$

- $d_{A|B}(\mathbf{v})$: Signed distance of $\mathbf{v}$ to $ORCA^{\tau}_{A|B}$; negative if $\mathbf{v} \in ORCA^{\tau}_{A|B}$

3.2.6 Static Obstacles. For a static obstacle O, agent A takes full responsibility. The permitted velocities form a half-plane tangent to $VO^{\tau}_{A|O}$:

$$ORCA^{\tau}_{A|O} = \left\{\mathbf{v} \,\middle|\, \left(\mathbf{v} - \mathbf{v}_A^{\text{opt}}\right) \cdot \mathbf{n}_O \geq 0\right\} \tag{19}$$

- $\mathbf{v}_A^{\text{opt}} = \mathbf{0}$: Ensures obstacle constraints are always feasible

3.3 Social Force Model for Pedestrian Dynamics

The Social Force Model (SFM) describes pedestrian motion as a superposition of forces reflecting individual motivations.

3.3.1 Forces Governing Pedestrian Behaviour. The motion of a pedestrian is governed by three primary components:

I. Desired Force. The desired force propels agents toward their intended targets. It is calculated based on the difference between the agent's current velocity $\mathbf{v}_{\text{current}}$ and the desired velocity $\mathbf{v}_{\text{desired}}$, scaled by the relaxation time τ:

$$\begin{aligned} \mathbf{F}_{\text{desired}} &= \frac{\mathbf{v}_{\text{desired}} - \mathbf{v}_{\text{current}}}{\tau}, \\ \mathbf{v}_{\text{desired}} &= V_0 \cdot \frac{\mathbf{d}}{||\mathbf{d}||} \end{aligned} \tag{20}$$

where $\mathbf{d}$ is the displacement vector from the agent's current position to the target.

II. Agent-Agent Repulsive Force. To prevent collisions, agents exert repulsive forces on each other. The magnitude of this force decreases exponentially with the distance d_{ij} between agents i and j, controlled by constants A and B. If agents are within twice the collision radius ($2R$), the force is adjusted to prevent overlap:

$$\mathbf{F}_{\text{repulsion},ij} = \begin{cases} A \cdot e^{-d_{ij}/B} \cdot \frac{\mathbf{d}_{ij}}{\|\mathbf{d}_{ij}\|} \cdot \left(1 - \frac{2R}{\|\mathbf{d}_{ij}\|}\right), & \text{if } d_{ij} < 2R \\ A \cdot e^{-d_{ij}/B} \cdot \frac{\mathbf{d}_{ij}}{\|\mathbf{d}_{ij}\|}, & \text{otherwise} \end{cases} \tag{21}$$

where $\mathbf{d}_{ij}$ is the displacement vector from agent j to agent i.

III. Boundary Repulsive Force. Agents near the environment boundaries experience repulsive forces to remain within bounds. These forces also decay exponentially with the distance d_{boundary} from the boundary, governed by constants C and D:

$$\mathbf{F}_{\text{boundary}} = C \cdot e^{-d_{\text{boundary}}/D} \cdot \mathbf{n} \tag{22}$$

where $\mathbf{n}$ is the normal vector pointing into the environment.

Total Force and Acceleration. The total force $\mathbf{F}_{\text{total},i}$ acting on agent i is the sum of the desired force, repulsive forces from other agents, and boundary forces:

$$\mathbf{F}_{\text{total},i} = \mathbf{F}_{\text{desired},i} + \sum_{j \neq i} \mathbf{F}_{\text{repulsion},ij} + \mathbf{F}_{\text{boundary},i} \tag{23}$$

To ensure numerical stability, the total force is capped at $F_{\max}$:

$$\mathbf{F}_{\text{total},i} = \begin{cases} \mathbf{F}_{\text{total},i} \cdot \frac{F_{\max}}{||\mathbf{F}_{\text{total},i}||}, & \text{if } ||\mathbf{F}_{\text{total},i}|| > F_{\max} \\ \mathbf{F}_{\text{total},i}, & \text{otherwise} \end{cases} \tag{24}$$

Using Newton's second law, the acceleration $\mathbf{a}_i$ is computed:

$$\mathbf{a}_i = \frac{\mathbf{F}_{\text{total},i}}{m} \tag{25}$$

3.3.2 Velocity and Position Updates. Agents' velocities and positions are updated using the computed acceleration and the time step Δt:

$$\begin{aligned} \mathbf{v}_i^{\text{new}} &= \mathbf{v}_i^{\text{current}} + \mathbf{a}_i \cdot \Delta t, \\ \mathbf{p}_i^{\text{new}} &= \mathbf{p}_i^{\text{current}} + \mathbf{v}_i^{\text{new}} \cdot \Delta t. \end{aligned} \tag{26}$$

If the updated velocity exceeds $V_{\max}$, it is scaled down to maintain:

$$||\mathbf{v}_i^{\text{new}}|| \leq V_{\max} \tag{27}$$

3.4 Universal Power Law Governing Pedestrian Interactions

The Universal Power Law (UPL) model is a force-based framework for pedestrian dynamics that quantifies interactions based on anticipatory collision avoidance[Karamouzas et al. 2014]. Below we have provided a detailed methodology:

3.4.1 Force Computation and State Update. The UPL framework models agent interactions through two primary forces: interaction forces and goal-directed forces.

3.4.2 Interaction Forces ($\mathbf{F}_{ij}$). To prevent collisions and simulate realistic agent dynamics, repulsive interaction forces are computed between a pair of agent (i) and (j). Each pedestrian is modeled as a circle with radius R, representing their personal space.The force $\mathbf{F}_{ij}$ exerted on agent i by agent j is defined as:

$$\mathbf{F}_{ij} = \begin{cases} \alpha \frac{\mathbf{r}_{ij}}{\tau^3} e^{-\tau/\tau_0}(2R - d), & \text{if } d < 2R, \\ \alpha \frac{\mathbf{r}_{ij}}{\tau^3} e^{-\tau/\tau_0}, & \text{otherwise.} \end{cases} \tag{28}$$

where:

$$\begin{aligned} \mathbf{r}_{ij} &= \mathbf{x}_i - \mathbf{x}_j, \quad d = |\mathbf{r}_{ij}|, \\ \tau &= \frac{\mathbf{r}_{ij} \cdot \mathbf{v}_{ij}}{|\mathbf{v}_{ij}|^2}, \quad \mathbf{v}_{ij} = \mathbf{v}_i - \mathbf{v}_j. \end{aligned}$$

The following are the key considerations in this computation:

- If $\tau \leq 0$, it is set to infinity to negate the force.
- A stronger repulsive force is applied when agents are within the collision radius ($d < 2R$).

3.4.3 Goal-Directed Forces ($\mathbf{F}_{goal}$). Each agent is directed towards a predefined target position through a goal-directed force:

$$\mathbf{F}_{\text{goal}} = K_{\text{goal}} \frac{\mathbf{d}}{|\mathbf{d}|}, \tag{29}$$

where $\mathbf{d} = \mathbf{x}_{\text{target}} - \mathbf{x}$ is the direction vector from the agent's current position to its target.

3.4.4 Total Force and State Update. The total force $\mathbf{F}_{\text{total},i}$ acting on agent i is the sum of all interaction forces and the goal-directed force:

$$\mathbf{F}_{\text{total},i} = \sum_{\substack{j=1 \\ j \neq i}}^{N} \mathbf{F}_{ij} + \mathbf{F}_{\text{goal},i}. \tag{30}$$

The agents' velocities and positions are updated using Newtonian mechanics:

$$\mathbf{v}_i^{\text{new}} = \mathbf{v}_i + \frac{\mathbf{F}_{\text{total},i}}{m} \Delta t, \tag{31}$$

$$\mathbf{x}_i^{\text{new}} = \mathbf{x}_i + \mathbf{v}_i^{\text{new}} \Delta t. \tag{32}$$

To prevent overlapping, positional adjustments are made if two agents come within twice the collision radius ($d < 2R$) that is as pedestrians approach each other and τ decreases, the increasing repulsive force naturally causes them to alter their trajectories, effectively maintaining personal space and preventing overlaps.

4 Experimental Data

This comparative simulation study compares pedestrian dynamics through a controlled experiment. The data comes from [Jain et al. 2025], which is a 10-second video of pedestrians moving in a street environment (10m X 3.5m) captured by an overhead camera. A frame from this video is shown in Figure 1, provides a clear view of pedestrian movements, allowing for easy trajectory extraction and analysis.

To process the visual data, the trajectories of individual pedestrians (agents) were extracted using colour-based segmentation in the HSV colour space, a computer vision method. The agents, identified by their red caps, are tracked over time. The method calculates centres using image moments and employs Euclidean distance for tracking, ensuring accurate detection of the agents' positions.

The Figure 1 displays two agents wearing red caps swapping places. The dataset contains 32 frames where both agents are detected at the same time, providing enough instances to simulate and analyse their interactions. These frames act as discrete time steps in the simulation, allowing the modelling of scenarios where agents try to swap positions while avoiding collisions.

A key part of this research is the agent-focused entropy calculation. This method measures the uncertainty in each agent's state estimates, offering insights into the reliability and predictability of their movements in the simulated environment. By evaluating entropy for each agent, the study improves the understanding of dynamic interactions and the success of collision avoidance strategies.

Figure 1: Two agents exchanging their positions

4.1 Extraction of 2D-Pixel Coordinates of Agents

The video capturing the agents was processed frame by frame at a frame rate of 30 frames per second (fps = 30). To reduce computational complexity, frames were processed at intervals of 10 frames, resulting in a time interval between processed frames of $\Delta t = \frac{\text{Frame Interval}}{\text{fps}} = \frac{10}{30} = 0.3333$ seconds. Each frame was preprocessed to enable effective detection of red-colored regions.

For segmentation, the frames were converted from the RGB colour space to the HSV (Hue, Saturation, Value) colour space, which enhances the differentiation of colours. Two HSV ranges were defined to detect both lighter and darker shades of red:

$$\begin{aligned} &\text{Lower Range:} \quad [0, 100, 100] \leq \text{HSV} \leq [10, 255, 255], \\ &\text{Upper Range:} \quad [160, 100, 100] \leq \text{HSV} \leq [180, 255, 255]. \end{aligned} \tag{33}$$

Binary masks corresponding to these HSV ranges were created and combined, highlighting the red-colored regions in the frames.

To identify individual red-colored objects, contours of the detected regions were computed. The two largest contours, corresponding to the red caps worn by the agents, were selected based on their areas. The centroids of these contours were calculated to determine the pixel coordinates of the agents. The centroid coordinates (C_x, C_y) were computed using spatial moments as:

$$C_x = \frac{M_{10}}{M_{00}}, \quad C_y = \frac{M_{01}}{M_{00}}, \tag{34}$$

where M_{10}, M_{01}, and M_{00} are the spatial moments derived from the contours.

To ensure the consistent tracking of an agent across frames, a proximity-based matching algorithm was implemented. Initially, the centroids of the two largest red regions were assigned to the two agents. In subsequent frames, the detected centroids were matched to the agents by minimizing the distances between their current and previous positions. The Euclidean distance between two points was calculated as:

$$\text{Distance} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}. \tag{35}$$

The pixel coordinates (x, y) of the agents were recorded for each processed frame, along with their corresponding timestamps. The timestamps were computed as:

$$\text{Timestamp} = \text{Frame Index} \times \Delta t, \tag{36}$$

where $\Delta t = 0.3333$ seconds.

This methodology enabled accurate detection and tracking of the two agents wearing red caps, ensuring reliable extraction of their 2D-pixel coordinates over time.

4.2 Pixel Coordinate to Real World Coordinate

The coordinates extracted from the video are in pixels and need to be converted to SI units (metres). Using four known real-world coordinates and their corresponding pixel coordinates, we establish the following relationship for conversion:

Real-World Coordinates (in metres)	Pixel Coordinates (in pixels)
$(0, 0)$	$(0, 0)$
$(10, 0)$	$(480, 0)$
$(10, 3.50)$	$(480, 272)$
$(0, 3.50)$	$(0, 272)$

For a general point (x, y) in pixels, the corresponding coordinates in metres are calculated as:

$$(x, y) = \left(\frac{x}{480} \times 10, \frac{y}{272} \times 3.50\right) \tag{37}$$

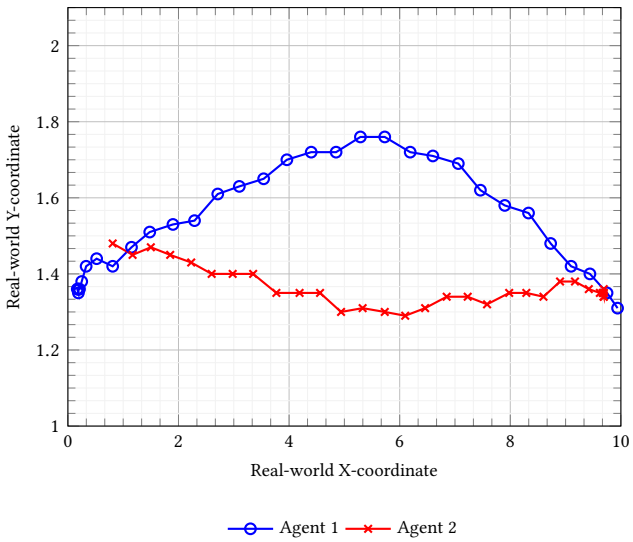

Figure 2: Experimental trajectory of both agents

Figure 2 illustrates the trajectories of two agents navigating a shared environment to exchange positions while avoiding a head-on collision. Agent 1 (blue) begins at the left boundary ($x \approx 0.19$, $y \approx 1.36$) and moves toward the right, while Agent 2 (red) starts at the right boundary ($x \approx 9.69$, $y \approx 1.36$) and travels leftward.

Both agents exhibit anticipatory collision avoidance by adjusting their vertical (y-axis) positions during their encounter near the environment's midpoint ($x \approx 5$). Agent 1 ascends to $y \approx 1.76$ while Agent 2 descends to $y \approx 1.29$, creating a vertical separation that ensures collision-free passage. Post-interaction, both agents smoothly return to their original y-axis baseline, completing the position swap. This behaviour aligns with the study's focus on quantifying uncertainty and dynamic adaptability in multi-agent systems, as reflected in subsequent entropy analyses.

4.3 Velocity Calculation Using the Central Difference Formula

To calculate the velocities of agents in the horizontal (v_x) and vertical (v_y) directions using the real-world coordinate, we use the Central Difference Formula with a five-point stencil [Sauer 2012] for the first derivative of a function, as shown in Algorithm 1. This algorithm determines the velocity components for each timestamp i by applying the five-point central difference formula to the positional data. The calculated velocities are then saved along with the corresponding timestamps and positions for further analysis. This method provides a reliable numerical approximation of the derivative.

In one dimension, if the spacing between points in the grid is h, the five-point stencil for a point x in the grid is:

$$\{x - 2h, x - h, x, x + h, x + 2h\} \tag{38}$$

The first derivative of a function $f(x)$ at x is then approximated as:

$$f'(x) = \frac{-f(x + 2h) + 8f(x + h) - 8f(x - h) + f(x - 2h)}{12h} \tag{39}$$

For this simulation, the x and y positions of an agent are represented as functions $f(t)$ and $g(t)$, respectively, where t is time:

$$f(t) = x, \quad g(t) = y \tag{40}$$

Here, h is the time difference between successive frames.

Initial Values: Since the formula requires $t > 2h$, the initial two velocity values for both agents are set to zero:

$$v_x[0] = v_x[1] = 0, \quad v_y[0] = v_y[1] = 0 \tag{41}$$

Algorithm 1 Central Difference Formula

1: **Input:** x_1 (x-coordinates of Agent 1), y_1 (y-coordinates of Agent 1), t (number of timestamps)
2: **Output:** CSV files `t`,`x_1`, `y_1`,`vx_1`, `vy_1`
3: **for** $i = 2$ to $t - 2$ **do**
4: $\quad h = \text{time}[i + 2] - \text{time}[i]$
5: $\quad v_x[i] = \frac{-x_1[i+2]+8x_1[i+1]-8x_1[i-1]+x_1[i-2]}{12h}$
6: $\quad v_y[i] = \frac{-y_1[i+2]+8y_1[i+1]-8y_1[i-1]+y_1[i-2]}{12h}$
7: **end for**

4.4 Velocity Unit Representation

Since the calculated velocities are small, they are shown in metres per second (m/s) for clarity and precision, as seen in fig 3. Agent 1 (blue), moving from left to right, shows positive v_x values that slowly increase from 0.0 m/s to a peak of 0.68 m/s near the midpoint ($x \approx 5$ m). Agent 2 (red), traveling from right to left, shows negative v_x values (indicating leftward motion), with its velocity magnitude reaching -0.64 m/s near $x \approx 7$ m, followed by a gradual slowdown.

The velocity profiles show that both agents have near-constant velocity phases after their initial acceleration, which helps in navigating efficiently while avoiding collisions. Agent 1 (blue) reaches a steady speed of about 0.6 m/s after reaching $x \approx 2$ m and maintains this speed until $x \approx 8$ m. Agent 2 (red) stabilises at approximately -0.6 m/s after moving leftward. The difference in peak velocities, with Agent 2 showing a higher magnitude, suggests that they make adjustments to avoid collisions, which aligns with the vertical movement seen in trajectory plots. Notably, the constant velocity briefly changes near the interaction midpoint ($x \approx 5$ m), where a slight slowdown occurs as the agents pass each other, likely due to collision avoidance.

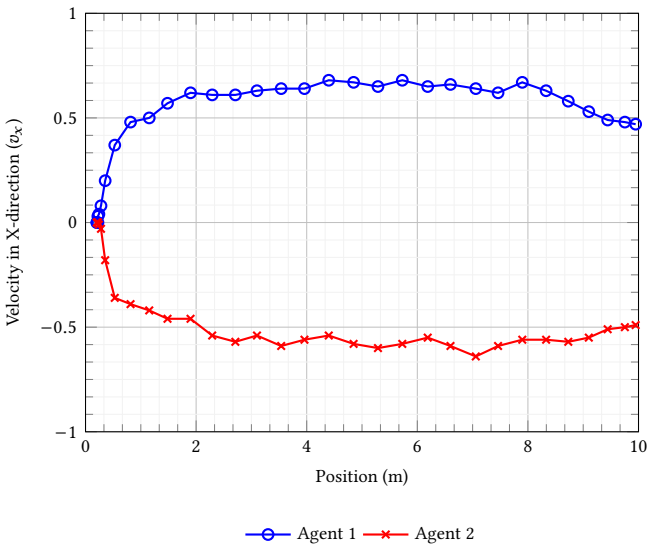

Figure 3: Velocities of both agents in the X-direction over time

5 Implementation and Evaluation

The objective of this study is to compare the results of three simulation methods (RVO2, UPL and SFM) on the basis of entropy and velocity profile by accurately tracking and analysing the movement of two agents within a simulated environment. An entropy-based metric is employed to quantify the uncertainty in each agent's state over time for different collision radii. This methodology integrates simulation techniques with statistical filtering to monitor and evaluate agent dynamics effectively.

5.1 Simulation Model ($\hat{f}$)

In this study, we employ three different simulation models: RVO2, UPL, and SFM. Each of these models offers a unique approach to modelling agent interactions and movement in dynamic environments.

RVO2: It is a velocity-based collision avoidance framework that enables agents to navigate in dynamic environments while avoiding collisions with other moving agents. Each agent selects a velocity that minimises deviation from its preferred velocity while ensuring that it does not enter the velocity obstacle (VO) induced by neighbouring agents. The collision avoidance strategy in RVO2 is governed by the following optimisation formulation:

$$\mathbf{v}_i^* = \arg\min_{\mathbf{v}_i} \|\mathbf{v}_i - \mathbf{v}_i^{\text{pref}}\|^2, \quad \mathbf{v}_i \notin \bigcup_{j \in \mathcal{N}(i)} \text{VO}_{ij} \tag{42}$$

where $\mathbf{v}_i^{\text{pref}}$ is the preferred velocity of agent i, and VO_{ij} represents the velocity obstacle imposed by agent j. RVO2 is widely used in crowd simulation scenarios where real-time collision avoidance and smooth navigation are critical. It is particularly effective in handling dense pedestrian traffic and multi-agent interactions by allowing for reciprocal adjustments between agents [Van den Berg et al. 2008].

UPL: It is a fundamental interaction model that captures the universal principles governing pedestrian dynamics. It describes how pedestrians modulate their velocity and trajectory based on the presence of other individuals, balancing efficiency and collision avoidance through a power-law-based formulation. The interaction force as discussed in Section 3.4.2 between two pedestrians at a given distance r follows the universal power law:

$$F_{ij} = A\left(\frac{r_{ij}}{r_0}\right)^{-\alpha} \tag{43}$$

where A is the interaction strength, r_0 is a reference distance, and α is an exponent that controls the decay of interaction strength with distance. This model effectively captures pedestrian interactions across different density conditions and has been validated against real-world pedestrian trajectories. UPL is particularly useful for modelling large-scale crowd dynamics where interactions occur across varying distances and densities [Karamouzas et al. 2014].

SFM: This model describes pedestrian movement as the result of attractive and repulsive forces acting on each agent. These forces include self-driven motion towards a goal, repulsion from obstacles, and social interactions with other agents. The acceleration of an agent is determined by the combined effect of these forces:

$$m_i \frac{d\mathbf{v}_i}{dt} = \mathbf{F}_i^{\text{goal}} + \sum_j \mathbf{F}_{ij}^{\text{social}} + \sum_O \mathbf{F}_{iO}^{\text{obstacle}} \tag{44}$$

where $\mathbf{F}_i^{\text{goal}}$ is the driving force towards the agent's goal, $\mathbf{F}_{ij}^{\text{social}}$ represents the interaction forces with other agents, and $\mathbf{F}_{iO}^{\text{obstacle}}$ accounts for obstacle avoidance. SFM is particularly effective in modelling human-like navigation behaviour, capturing emergent patterns such as lane formation and collective motion in dense pedestrian scenarios [Helbing and Molnar 1995].

Each of these simulation models provides a different perspective on agent motion. RVO2 focuses on velocity-based collision avoidance with reciprocal adjustments, UPL generalises pedestrian interactions through power-law dynamics, and SFM incorporates social interaction forces to model movement behaviour. The choice of model depends on the specific application, with RVO2 excelling in real-time simulations [Van den Berg et al. 2008], UPL offering a universal perspective on pedestrian behaviour [Jain et al. 2025], and SFM effectively capturing emergent collective phenomena [Huang et al. 2018].

5.2 Data Acquisition

The initial step involves collecting real-world data representing the positions and velocities of the two agents. This data is stored in a Comma-Separated Values (CSV) file, which is subsequently loaded into the simulation framework. Each agent's state is characterised by four parameters given below and are given in Table A1 & A2 in Appendix :

- Position Coordinates: x_{pos} and y_{pos}
- Velocity Components: v_x and v_y

These parameters collectively define the state vector for each agent:

$$\mathbf{s} = [x_{\text{pos}}, y_{\text{pos}}, v_x, v_y]$$

5.3 Simulation Setup

The simulation environment is established for various simulation model, which facilitates realistic agent movement while avoiding collisions. Key constants and parameters used in the simulation are given below:

- **Number of Agents (N)**: The simulation models $N = 2$ agents interacting within the environment.
- **Ensemble Size (E)**: An ensemble of $E = 2000$ state estimates is maintained to capture the distribution of possible states for each agent.
- **State Dimensions (D)**: Each agent's state is characterized by $D = 4$ dimensions: position coordinates (x, y) and velocity components (v_x, v_y).
- **Maximum Iterations (T)**: The simulation progresses through a maximum of $T = 32$ timesteps unless termination conditions are met earlier.
- **Noise Standard Deviation (σ)**: Gaussian noise with a standard deviation of $\sigma = 0.05$ is introduced to model process uncertainties in state predictions.
- **Window Size (w)**: A window size of $W = 5$ sec is used to quantify the uncertainty in a simulation method's predictions, by aggregating covariance Q to compute the agent-centric entropy metric.

Each agent is initialised with its respective state from the loaded data, and target positions are assigned based on the agents' final recorded positions.

5.4 Ensemble-Based Kalman Filter

To manage and update the uncertainty in each agent's state, an Ensemble Kalman Filter (EnKF) [Mandel 2009] approach is implemented. This technique leverages multiple ensemble members to represent the distribution of possible states, allowing for robust state estimation and uncertainty quantification.

5.4.1 Ensemble Generation. At each simulation timestep, an ensemble of E members is generated by introducing Gaussian noise to the current state of each agent. Noise added to velocity is half of that being added in the state of agents, as velocity data is less noisy (Fig 3) than positional data (Fig 2). By assigning velocity a smaller noise variance, the model prioritises stable velocity estimates, reducing erratic jumps in trajectory predictions. So, the simulator function $\hat{f}$ as discussed in section 4.3, takes in a state $x_{\text{pos},k}$ of the crowd at time k and produces an estimate of the state $x_{\text{pos},k+1}$ of the crowd at one unit of time later (henceforth referred to as a timestep). The noise ensures variability within the ensemble, enabling the filter to capture the spread and uncertainty in the agents' states. The noise is applied as follows:

$$
\begin{aligned}
x_{\text{pos,k+1}}^{(e)} &= \hat{f}(x_{\text{pos,k}}) + \mathcal{N}(0, \sigma^2), \\
y_{\text{pos,k+1}}^{(e)} &= \hat{f}(y_{\text{pos,k}}) + \mathcal{N}(0, \sigma^2), \\
v_{\text{x,k+1}}^{(e)} &= \hat{f}(v_{\text{x,k}}) + \mathcal{N}\left(0, \left(\frac{\sigma}{2}\right)^2\right), \\
v_{\text{y,k+1}}^{(e)} &= \hat{f}(v_{\text{y,k}}) + \mathcal{N}\left(0, \left(\frac{\sigma}{2}\right)^2\right).
\end{aligned} \tag{45}
$$

where $\mathcal{N}(0, \sigma^2)$ denotes a Gaussian distribution with mean 0 and variance σ^2.

5.4.2 Kalman Gain Calculation and Correction. The Kalman Gain ($\mathbf{K}$) is pivotal in updating the state estimates by balancing the uncertainty from the prediction and the measurements. For each agent, the Kalman Gain is computed using the state covariance ($\mathbf{P}$) and the measurement noise covariance (R):

$$
K = \frac{P}{P + R} \tag{46}
$$

Given that both P and R are diagonal matrices, the calculation simplifies to element-wise operations:

$$
K_i = \frac{P_{ii}}{P_{ii} + R_{ii}}, \quad \forall i \in \{1, 2, 3, 4\} \tag{47}
$$

where i corresponds to each state dimension.

Subsequently, the ensemble states are corrected using the Kalman Gain:

$$
\mathbf{s}_{\text{new}}^{(e)} = \hat{f}(\mathbf{s}^{(e)}) + \mathbf{K} \odot (\mathbf{z} - \hat{f}(\mathbf{s}^{(e)})) \tag{48}
$$

where:

- $\mathbf{s}^{(e)}$ is the ensemble member's state.
- $\mathbf{z}$ is the measurement state.
- $\odot$ denotes element-wise multiplication.

5.4.3 Covariance Update using MLE and Smoothing. To refine the process noise covariance ($\mathbf{Q}$), MLE is employed. This involves calculating the sum of squared differences between the measurements and the ensemble predictions:

$$
Q_i = \frac{1}{E} \sum_{e=1}^{E} \left(z_i - \hat{f}(s_i^{(e)})\right) \left(z_i - \hat{f}(s_i^{(e)})\right)^\top + \epsilon \tag{49}
$$

where E=2000 ensemble members contribute to the average and ϵ is a small regularisation term (1×10^{-6}) to prevent numerical instability as discussed in the AWS documentation [AWS 2025].

A rolling window approach is utilised to smooth the covariance estimates. For a window size of $w = 5\,\text{sec}$:

$$
\tilde{Q}_i = \frac{1}{5} \sum_{k=t-w}^{t+w} Q_i^{(k)} \tag{50}
$$

This averaging mitigates fluctuations and ensures stable covariance estimates.

5.5 Entropy Calculation

Entropy serves as a quantitative measure of the uncertainty associated with each agent's state covariance. For each agent, entropy (S) is calculated using the determinant of the covariance matrix ($\mathbf{Q}$) [Jwo et al. 2023]:

$$
S(\mathbf{Q}) = \frac{1}{2} \left(\ln\left((2\pi e)^D \cdot \det(\mathbf{Q})\right)\right) \tag{51}
$$

Given that $\mathbf{Q}$ is a diagonal matrix, its determinant simplifies to the product of its diagonal elements:

$$
\det(\mathbf{Q}) = \prod_{i=1}^{D} Q_{ii} \tag{52}
$$

Thus, the entropy expression becomes:

$$
S(\mathbf{Q}) = \frac{1}{2} \left(D \ln(2\pi e) + \sum_{i=1}^{D} \ln(Q_{ii})\right) \tag{53}
$$

where $D = 4$ represents the state dimensions.

To ensure numerical stability, any non-positive or undefined elements within $\mathbf{Q}$ are adjusted by assigning a minimal positive value (1×10^{-6}) before computing the determinant.

The computed entropy reflects the degree of uncertainty in the agents' positions and velocities. Lower entropy values indicate higher confidence in the state estimates, whereas higher values signify greater uncertainty.

5.6 Data-driven evaluation and analysis

5.6.1 RVO2. Several critical variables and parameters govern the behaviour and performance of the RVO2 simulation:

Table 1: RVO2 Parameter Settings for Agent Simulation

Parameter	Description
neighborDist	Maximum neighbour distance: 5.0 m
maxNeighbors	Maximum number of neighbours: 2
timeHorizon	Agent-agent collision time horizon: 1.5 s
timeHorizonObst	Agent-obstacle collision time horizon: 5.0 s
radius	Collision radius (r): 0.2 m
maxSpeed	Maximum agent speed: 0.68 m/s

The trajectories generated by the RVO simulation as shown in fig 4 exhibit distinct behavioral characteristics compared to the real-world data as shown in Fig 2. While the simulated agents for $r = 0.2$ follow smooth, near-linear paths with minimal lateral deviation (<0.1m in the Y-axis), the real-world trajectories (Fig 2 display pronounced non-linear adjustments, particularly in the Y-direction (up to 0.45m variability). This discrepancy arises from the RVO algorithm's deterministic collision-avoidance logic, which prioritises direct target-seeking behaviour and suppresses natural path oscillations observed in real-world interactions. This discrepancy gets more pronounced for larger collision radius ($r = 1.6$) as shown in Fig 4.

For $r = 0.2$ both agents exhibit smooth, predictable trajectories with minimal vertical displacement. Agent 1 (blue) moves from left to right, starting at approximately (0.31, 1.36) and ending at (9.17,

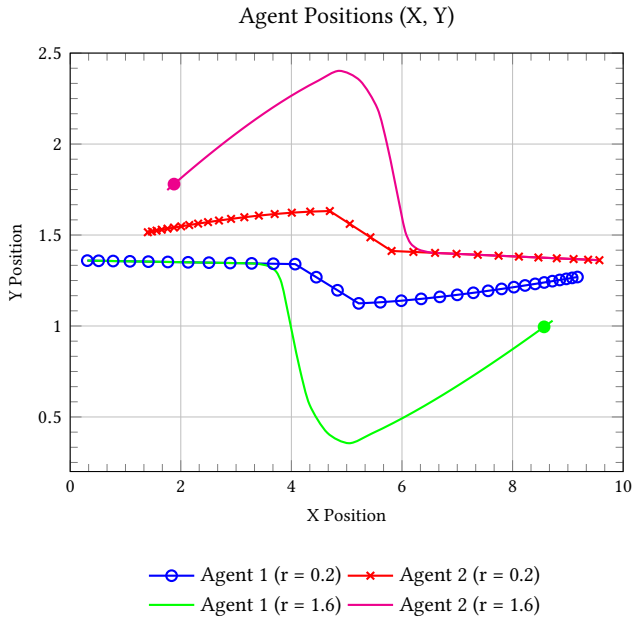

Figure 4: Agent Positions (RVO2)

1.27), maintaining a relatively flat path until around $x = 4$, where it dips slightly before gradually rising again. Agent 2 (red) travels in the opposite direction, from (9.57, 1.36) to (1.40, 1.52), with a modest rise in y-position around the middle of its journey.

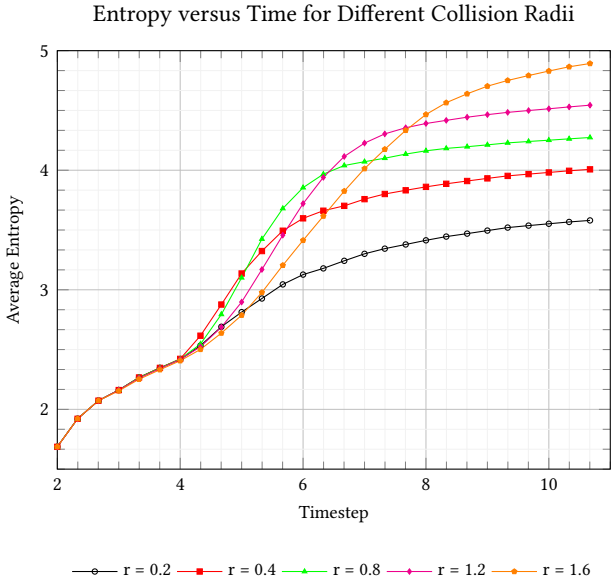

Figure 5: Entropy versus time for different Collision radii (RVO2)

The entropy behaviour in the two-agent RVO simulation reveals important patterns across different collision radii. The temporal

Table 2: Final entropy values at different Collision radii (RVO2)

Collision Radius (m)	Average Entropy
0.2	3.58
0.4	4.00
0.8	4.25
1.2	4.51
1.6	4.82

evolution of average entropy (Eqn 12) reveals a general monotonic increase across all tested radii, as shown in Fig 5. This consistent upward trend reflects the cumulative nature of simulation errors. Even small discrepancies in early time steps tend to propagate forward, gradually amplifying deviations as agents continue to interact. For instance, between $t = 2$ s and $t = 4$ s, the entropy remains approximately constant (~ 0.01) across all radii, indicating similar behaviour during initialisation. However, by $t = 10.67$ s, the entropy increases significantly, reaching around 3.58 for $r = 0.2$ and 4.89 for $r = 1.6$. This indicates that minor inaccuracies grow more pronounced over time, especially in scenarios involving complex agent interactions. This is also evident in Fig. 4, where agents with $r = 1.6$ show trajectories that deviate more significantly from the experimental path and thus have higher entropy compared to those of agents with smaller radii. For the purpose of readability, we have only shown trajectories for two radii—one corresponding to the maximum tested value ($r = 1.6$) and the other to the optimal value based on entropy minimisation.

Therefore, collision radius r, which defines the effective personal space maintained by each agent during navigation, has a direct influence on entropy values. Smaller radii, such as $r = 0.2$, allow agents to come into closer proximity before initiating avoidance behaviour. This leads to more realistic and human-like interactions, particularly in sparse environments. Entropy under this configuration remains relatively low in early timesteps and increases gradually, reaching approximately 3.58 by $t = 10.67$ s, indicating only moderate accumulation of error. In contrast, larger radii such as $r = 1.6$ impose a broader zone of personal space, prompting agents to detour prematurely and generate wide, unrealistic paths. This conservative behaviour leads to rapid entropy growth, with values spiking to around 4.89 by $t = 10$ s, especially during periods of dense interaction or trajectory overlap. The entropy trends in RVO2 simulations for a fixed window size reveal a pronounced trade-off between predictability (low entropy) and agent adaptability (higher entropy) as collision radii vary, larger safety margins provide better collision avoidance but at the cost of reduced movement freedom.

5.6.2 UPL: . Several critical variables and parameters that govern the behaviour and performance of the UPL simulation are adapted from the work of [Karamouzas et al. 2014] to mirror empirical observations of pedestrian behaviour, ensuring that the UPL simulation accurately reproduces real-world crowd dynamics:

The trajectories generated by the UPL simulation are shown in fig 6 for different radii. For agents with $r = 0.2$, the model output exhibits identical overall trajectory patterns with real-world data, capturing the fundamental movement behaviour where Agent 1

Table 3: UPL Parameter Settings for Agent Simulation

Parameter	Description
K	Scaling constant for interaction energy: 1.0
τ_0	Intrinsic interaction range: 1.4 seconds
α	Scaling factor for repulsive force: 2.0
K_{goal}	Scaling factor for goal-directed force: 1.0

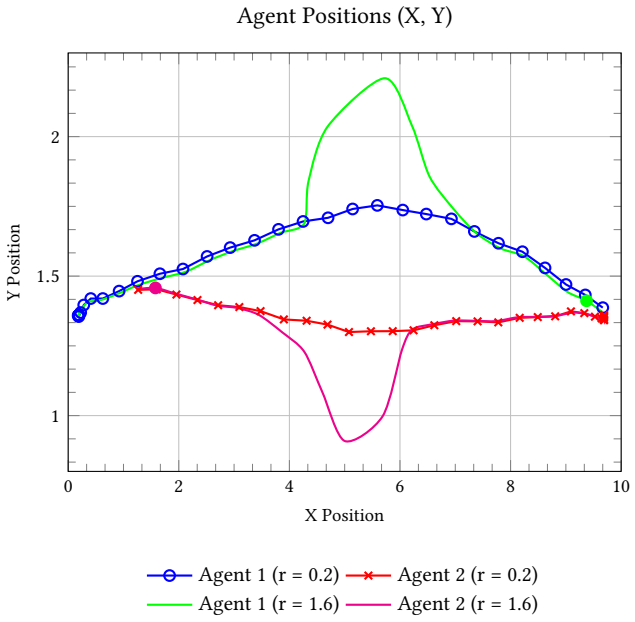

Figure 6: Agent Positions (UPL)

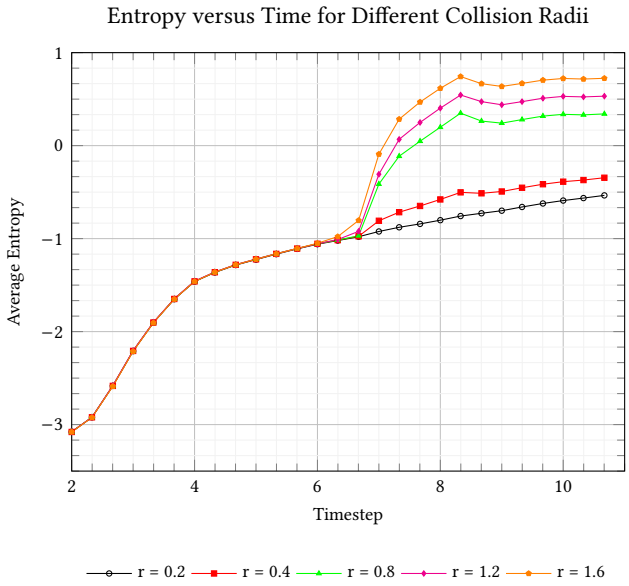

Figure 7: Entropy versus time for different Collision radii (UPL)

Table 4: Final entropy values at different Collision radii (UPL)

Collision Radius (m)	Average Entropy
0.2	-0.53
0.4	-0.33
0.8	0.34
1.2	0.53
1.6	0.72

(blue) moves from left to right and Agent 2 (red) moves from right to left. Both datasets maintain nearly identical starting and ending positions, demonstrating the UPL model's ability to accurately reproduce basic navigation behaviour. The precision of starting and ending coordinates further validates the model's accuracy, with Agent 1 beginning at approximately (0.19, 1.36) and ending near (9.7, 1.35), while Agent 2 starts at (9.69, 1.36) and concludes around (0.8, 1.48). This high level of accuracy indicates excellent calibration of the simulation.

The collision avoidance strategy remains consistent, as agents adjust their vertical positions when passing each other to maintain separation. This vertical displacement strategy is a key similarity between the simulated and real-world movements. Additionally, the magnitude of vertical displacement is highly comparable, with Agent 1 reaching a maximum y-position of approximately 1.75 in the UPL simulation and 1.76 in the real-world data, showing a minimal difference of only 0.01 units. The UPL simulation successfully replicates essential characteristics of real human movement patterns, particularly the strategic vertical displacement utilised for collision avoidance.

The data, as shown in Table 4, reveal critical insights into how physical parameters influence simulation accuracy within the UPL framework. The analysis shows that the choice of collision radius significantly impacts how well the simulated trajectories align with real-world pedestrian behaviour. Among all the tested values, the radius $r = 0.2$ consistently yields the lowest average entropy across all time steps, beginning at approximately -3.07 and stabilizing around -0.53 at $t = 10.67$ s. This indicates a high level of agreement between the simulated and observed trajectories. In contrast, for $r = 0.4$, while the entropy also starts near -3.49, it increases more rapidly and reaches -0.35 by the end, suggesting growing deviations as agent interactions become more pronounced. For even larger radii, specifically $r \geq 0.8$, entropy rises sharply and even reaches positive values (e.g., $r = 1.6$ peaks at +0.72), reflecting a marked divergence from realistic pedestrian motion. Such a large value of entropy is also evident from the trajectory the agents took while avoiding collision, as shown in Fig 6 for $r = 1.6$.

A notable point of divergence occurs around $t \approx 6$ s in Fig 7, where entropy spikes for $r \geq 0.8$ coincide with agents attempting to resolve collisions. The spike corresponds to the point where both agents initially approach each other, with the distance decreasing from 3.32 units at 5.33s A1 to a minimum of 0.45 units at 6.67s (closest proximity). After this point, the distance increases steadily to 2.44 units by 7.67s, indicating the agents begin to move apart

after their closest encounter. Larger collision radii appear to induce overly conservative avoidance behaviour and thus a bigger detour (Fig 6), which contradicts the anticipatory nature of the UPL model. The UPL defines repulsion through $E(\tau) \propto \tau^{-2}$, relying on time-to-collision τ rather than static proximity. A smaller radius like $r = 0.2$ permits agents to delay avoidance until τ is sufficiently small, better replicating natural human behaviour. Larger radii, in contrast, impose artificial spatial constraints that lead to abrupt and premature detours, deviating from the more fluid interactions seen in reality.

Therefore, the consistent entropy minimum at $r = 0.2$ highlights its suitability as the optimal parameter for UPL-based simulations and for comparative study with results of RVO2 and SFM. It preserves the anticipatory human behaviour by avoiding distortions caused by excessive geometric constraints.

5.6.3 SFM: . Several critical variables and parameters govern the behaviour and performance of the SFM simulation, are given below [Helbing and Molnar 1995]:

Table 5: SFM Model Parameter Settings for Agent Simulation

Parameter	Value and Description
A	Interaction strength between agents: 2.1 m^2/s^2
B	Distance decay for repulsion: 0.3 m
C	Interaction strength with walls: 10.0 m^2/s^2
D	Distance decay for boundary repulsion: 0.2 m
V_0	Desired speed: 1.5 m/s
τ	Relaxation time: 0.5 s
F_{max}	Maximum allowable force: 1000.0 N

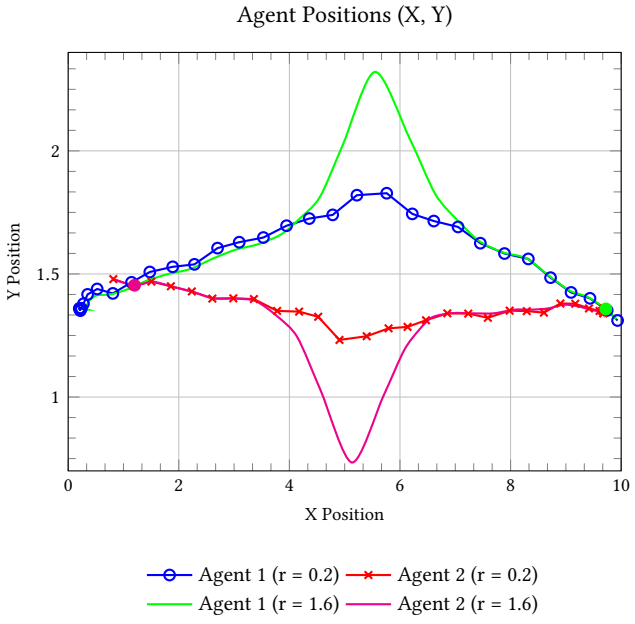

Figure 8: Position Coordinates of Agents 1 and 2 (SFM)

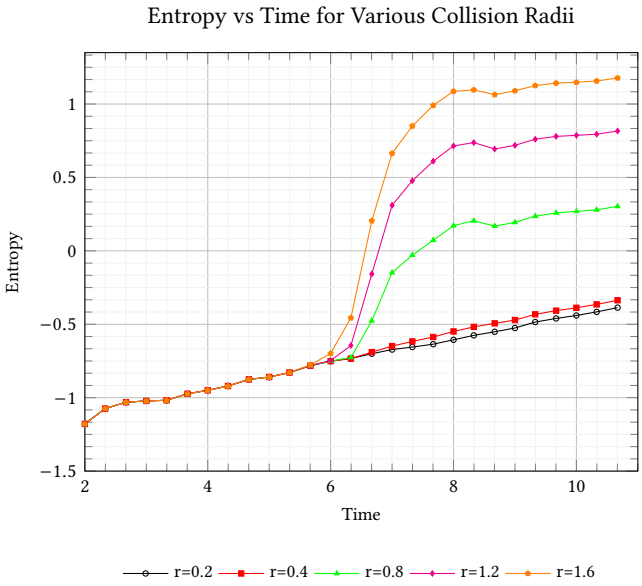

Figure 9: Entropy versus time for different Collision radii (SFM)

The trajectories generated by the SFM simulation are shown in fig 8 for different collision radii. For $r = 0.2$, the comparison of positional coordinates reveals distinct deviations between Fig 2 and Fig 8, particularly in mid-trajectory segments. For Agent 1, while both paths start and end nearly identically (real: 0.19, 1.36 $\rightarrow$ 9.94, 1.31; simulated: 0.206, 1.360 $\rightarrow$ 9.935, 1.311), discrepancies emerge during intermediate phases. At $t = 6$ s, the real position is (4.4, 1.72), while the simulation places the agent at (4.359, 1.725), reflecting a horizontal drift of 0.041 m. This divergence grows slightly at $t = 7.67$ s, where the real coordinate (6.6, 1.71) contrasts with the simulated (6.61, 1.71), demonstrating a positional offset in the X-direction.

For Agent 2, while starting and ending positions align closely (real: 9.69, 1.36 $\rightarrow$ 0.81, 1.48; simulated: 9.68, 1.36 $\rightarrow$ 0.81, 1.47), mid-trajectory deviations are significant. At $t = 5.67$ s, the real position (6.46, 1.31) diverges from the simulated (6.470, 1.312) by 0.01 m in X and 0.002 m in Y. However, the largest discrepancy occurs at $t = 7.33$ s, where the real coordinate (4.56, 1.35) contrasts sharply with the simulated (4.52, 1.33), yielding a horizontal error of 0.039 m and a vertical gap of 0.024 m.

In summary, while the SFM achieves strong global path alignment, mid-trajectory positional deviations (up to 0.05 m) reflect inherent limitations in modelling fine-grained interaction dynamics. These differences underscore the need for parameter tuning in order to better match real-world pedestrian navigation patterns.

The entropy-time profiles for different collision radii in the SFM reveal critical insights into how agent size influences trajectory predictability. For smaller radii ($r = 0.2$–0.4), the trajectory mimics experimental data, the entropy remains relatively stable over time, increasing from approximately -1.17 to around -0.38 by $t = 10.67$. These curves as shown in Fig 9 exhibit minimal divergence, indicating that smaller personal spaces do not significantly amplify

Table 6: Final entropy values at different Collision radii (SFM)

Collision Radius (m)	Average Entropy
0.2	-0.38
0.4	-0.33
0.8	0.30
1.2	0.81
1.6	1.17

uncertainty. Agents with compact radii interact less frequently and maintain smoother trajectories, allowing the SFM to predict paths with high confidence. The data from the Table 6 reveals a clear and consistent inverse relationship between collision radius and entropy. As the collision radius increases from 0.2m to 1.6m, the average entropy increases from -0.38 to 1.17 (Table 6) as the trajectory deviates from a real-world situation, as shown in Fig 9 for agents with $r = 1.6$.

For moderate radii ($r = 0.8$), entropy rises more steeply, reaching 0.30 by $t = 10.67s$. This increased uncertainty stems from broader collision-avoidance maneuvers. Agents initiate repulsion earlier due to their larger radii, leading to more frequent path adjustments. While the model retains reasonable predictive power, the compounding effects of these interactions introduce measurable unpredictability over time.

We observe a stark divergence occurs for larger radii ($r = 1.2$–1.6). These curves exhibit sharp entropy spikes after $t = 6$, peaking at approximately 0.81 ($r = 1.2$) and 1.17 ($r = 1.6$) as larger personal space requirements force agents to initiate avoidance maneuvers earlier and more aggressively, amplifying trajectory deviations from real-world data.

These results underscore the importance of radius parameterisation in SFM. From a practical perspective, this analysis indicates that in SFM simulations, smaller collision radii (0.2-0.8m) allow for more natural and flexible agent movement, while larger radii (1.2-1.6m) impose severe constraints that may result in overly rigid and predictable trajectories.

6 Result and Analysis

Based on the entropy criteria, a collision radius of 0.2 meters was selected for the final comparison between the three collision avoidance models. This value was chosen as it provides an optimal balance between safety and movement flexibility, with entropy values falling within the desirable range for natural pedestrian movement patterns.

As per the chosen radius of 0.2, Table 7 shows the comparison of performance metrics across all three models. In addition to entropy measurements across different collision avoidance frameworks, showing how predictable and constrained agent movement is during the simulation, we also represent prediction error. Average displacement error is the mean square error (MSE) across all predicted points and the ground truth [Pellegrini et al. 2009], and Final displacement error (FDE) is the difference between the final predicted point and the final ground truth of a trajectory.

The entropy versus time plot for different models reveals clear differences in the predictive accuracy of RVO2, UPL, and SFM frameworks (Figures 5 7 9). These differences highlight the various strengths of each model in capturing realistic pedestrian dynamics. The UPL model consistently shows negative entropy, starting at -3.49 and stabilizing around -0.53. This suggests strong alignment with real-world pedestrian paths, which is further supported by low prediction error. This performance stems from UPL's reliance on the Universal Power Law, where interactions follow a τ^{-2} relationship (τ = time to collision). By focusing on anticipatory adjustments rather than reactive geometric constraints, UPL effectively mimics human-like collision avoidance behaviour. During early interactions ($t < 3$ s), the nearly minimal entropy reflects accurate modelling of initial stationary or slow-motion phases. In mid to late interactions ($t > 5$ s), the gradual rise in entropy remains limited, resembling the smooth changes in velocity seen in real pedestrian behaviour, such as gradual acceleration to navigate around potential collisions. UPL's success lies in its τ-driven energy decay ($E \propto \tau^{-2}$), allowing agents to postpone evasive actions until absolutely necessary, which mimics human judgment and avoids unrealistic constraints.

Table 7: Comparison of Simulation Models. We represent the performance metrics as follows: The first two rows indicate the Final Displacement Error (FDE) for Agent 1 and Agent 2, respectively. The third row presents the overall Average Displacement Error across the trajectory for both agents, and the fourth row shows the final Average Entropy at the end of the prediction time stamp.

Metric	RVO2	UPL	SFM
Agent 1 FDE	0.77	0.13	0.28
Agent 2 FDE	0.60	0.14	0.46
Avg. Disp. Error	2.73	0.13	0.21
Avg. Entropy	3.58	-0.53	-0.38

In contrast, RVO2 shows positive and rising entropy and high prediction error, while SFM displays negative entropy for small radii but high positive entropy for larger radii and relatively lower prediction error, highlighting systematic differences from real behaviour. For RVO2, entropy increases steadily from 1.68 to 3.58, reflecting cumulative errors from rigid geometric constraints. A sharp spike in entropy after collisions (beyond $t = 7$ s) further exposes RVO2's lack of anticipatory modelling, leading to unstable velocity profiles. Meanwhile, SFM shows a more gradual entropy increase from -1.17 to -0.38, indicating better behaviour compared to RVO2. Although it produces results (Fig 8) close to experimental data (Fig 2), its use of distance-based repulsive forces does not capture human sensitivity to time-to-collision, often resulting in overly cautious avoidance strategies. The Social Force Model (SFM) shows unnaturally large deviations in the Y-direction compared to real-world paths, especially during collision avoidance. For example, at 5.76 seconds, SFM places Agent 1 at $(5.76, 1.83)$ (Fig 8), while the real trajectory has it at $(5.73, 1.76)$ (Fig 2), leading to a Y-axis deviation of $+0.07$ which is much larger than the deviation of UPL's position for Agent 1 at $(5.76, 1.75)$, with a deviation of -0.01. This shows that SFM agents take wider, circular paths to avoid collisions, unlike real humans who tend to maintain straighter paths

with minimal lateral adjustments. While SFM's performance (e.g., collision avoidance success) may seem similar to models like UPL, its exaggerated lateral movements reveal a significant deviation from natural pedestrian behaviour at a detailed level.

In conclusion, the UPL model outperforms the other two across all metrics. The entropy-radius analysis (Table 2 4 6) highlights two key findings. First, entropy generally increases with collision radius, particularly when agents interact with each other or respond to environmental factors. Second, the collision radius has a strong impact on the realism of the simulation. Larger radii lead to early and exaggerated avoidance behaviours, increasing entropy and reducing the model's accuracy compared to real-world movement. Based on the entropy value, a collision radius of $r = 0.2\,\text{m}$ emerges as a sound baseline for comparative simulation studies with the lowest entropy value, indicating it stays true to natural interaction patterns.

Additionally, the entropy trends clearly show that anticipatory, τ-based models like UPL perform better than geometric (RVO2) and force-based (SFM) approaches in simulating human crowd behaviour. These findings highlight the need to include time-to-collision logic in collision avoidance models to achieve realism and predictive accuracy in complex situations.

Next, we will compare the velocity behaviour obtained from all three models. For this analysis, we have taken the average magnitude of velocity for both agents and plotted it against position. Figure 10 illustrates the velocity profiles for the real agent data alongside the three simulation models: RVO, UPL, and SFM.

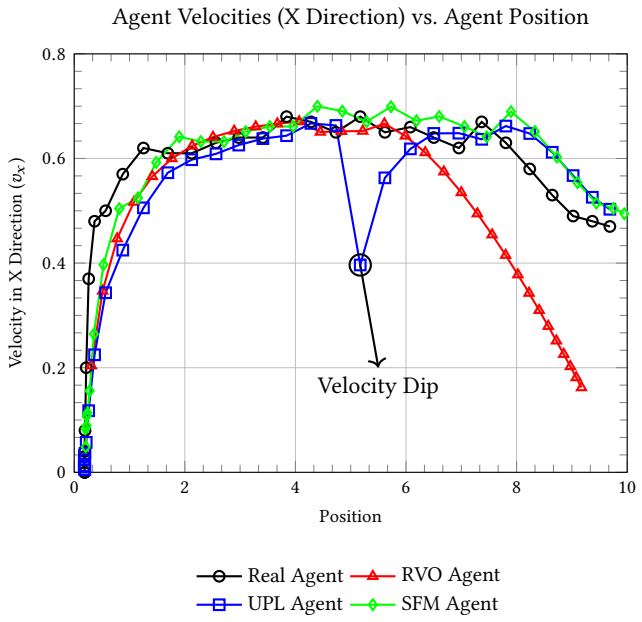

Figure 10: Agent Velocities vs. Agent Position in x-Direction

The RVO model (red line) demonstrates a different velocity pattern. It begins with a higher initial velocity (0.2 m/s) and accelerates more rapidly than the real agent, reaching its peak velocity of about 0.67 m/s earlier in the trajectory (position 4). However, the RVO model shows a continuous deceleration after reaching its peak, unlike the real agent which maintains a plateau of high velocity for a longer period. This early deceleration is consistent with the velocity obstacle approach, where agents preemptively reduce speed to avoid potential collisions.

The UPL model (blue line) most closely resembles the real agent's velocity profile. It starts with a low initial velocity similar to the real agent and follows a comparable acceleration pattern. The UPL model accurately captures the plateau phase of relatively constant velocity in the middle section of the trajectory. However, it shows a notable velocity dip around position 5, corresponding to the point where the agents are very near to each other and are about to pass each other (evident from Table A1 and Fig 2) and collision avoidance is most active and then picks up the velocity profile more closely than SFM to experimental velocity profile (black line). This temporary reduction in velocity during agent interaction is a distinctive feature of the UPL model.

The SFM model (green line) exhibits the most consistent velocity profile among the three simulation approaches. It begins with a moderate initial velocity (0.05 m/s) and accelerates smoothly to reach a peak velocity of approximately 0.7 m/s. The SFM model maintains this high velocity for an extended period, similar to the real agent, before gradually decelerating. The social force approach produces a smoother velocity curve with fewer fluctuations compared to the other models, indicating that forces are more evenly distributed throughout the simulation, but deviate more from the experimental velocity profile compared to UPL.

Overall, while all three models capture the general trend of acceleration followed by deceleration, the UPL model most accurately reproduces the real agent's velocity profile, despite exhibiting a momentary reduction in velocity during agent interactions. The SFM model results in the smoothest velocity transitions, whereas the RVO model exhibits an earlier onset of deceleration that fails to align with the sustained high-velocity phase observed in the real-world data.

Based on the analysis above and the model outputs illustrated in Fig. 4, Fig. 6, and Fig. 8, we observe a relationship between the collision radius and the entropy value. Specifically, agents with larger collision radii exhibit higher entropy values and a larger deviation from the real-world trajectories, while those with smaller collision radii demonstrate much lower entropy values and better trajectory alignment. These findings suggest that entropy serves as a reliable and sufficient metric for evaluating the accuracy of simulation models in replicating real-world agent behaviour.

7 Conclusion

This research has established entropy as a robust, unified metric for evaluating pedestrian simulation models, demonstrating its effectiveness through rigorous comparative analysis of *RVO2*, *UPL*, and *SFM* frameworks by integrating an EnKS and MLE based approach. The *UPL* model emerged as the superior approach, achieving the lowest entropy (-0.53) and minimal trajectory errors (FDE: 0.13–0.14), validating its τ-based anticipatory mechanics as the most faithful replication of human collision avoidance. Further, the entropy-radius correlation revealed $r = 0.2\,\text{m}$ as the optimal

collision radius across all models, balancing safety and natural movement while minimising uncertainty.

Future work should extend this framework to high-density scenarios and real-time parameter adaptation, while exploring hybrid models that combine *UPL*'s anticipation with *RVO2*'s scalability. Such advancements will enable next-generation simulations that accurately capture the complexity of human navigation in dynamic environments, ultimately enhancing safety and efficiency in shared human-robot spaces.

References

AWS. 2025. Training Parameters. https://docs.aws.amazon.com/machine-learning/latest/dg/training-parameters.html. Accessed: April 2025.

Xianye Ben, Xifa Huang, Zhaoyi Zhuang, Rui Yan, and Sen Xu. 2013. Agent-based approach for crowded pedestrian evacuation simulation. *IET Intelligent Transport Systems* 7, 1 (2013), 55–67.

Ron Cowen. 2014. Mathematical time law governs crowd flow. (2014).

Claudio Feliciani and Katsuhiro Nishinari. 2016. Empirical analysis of the lane formation process in bidirectional pedestrian flow. *Physical Review E* 94, 3 (2016), 032304.

Stephen J Guy, Jur Van Den Berg, Wenxi Liu, Rynson Lau, Ming C Lin, and Dinesh Manocha. 2012. A statistical similarity measure for aggregate crowd dynamics. *ACM Transactions on Graphics (TOG)* 31, 6 (2012), 1–11.

Dirk Helbing and Peter Molnar. 1995. Social force model for pedestrian dynamics. *Physical review E* 51, 5 (1995), 4282.

Lin Huang, Jianhua Gong, Wenhang Li, Tao Xu, Shen Shen, Jianming Liang, Quanlong Feng, Dong Zhang, and Jun Sun. 2018. Social Force Model-Based Group Behavior Simulation in Virtual Geographic Environments. *ISPRS International Journal of Geo-Information* 7, 2 (2018). doi:10.3390/ijgi7020079

Kanika Jain, Shankar Prawesh, Indranil Saha Dalal, and Anurag Tripathi. 2025. Benchmarking Pedestrian Dynamics Models for Common Scenarios: An Evaluation of Force-Based Models. *arXiv preprint arXiv:2501.05106* (2025).

Dah-Jing Jwo, Ta-Shun Cho, and Amita Biswal. 2023. Geometric insights into the multivariate gaussian distribution and its entropy and mutual information. *Entropy* 25, 8 (2023), 1177.

Ioannis Karamouzas, Brian Skinner, and Stephen J Guy. 2014. Universal power law governing pedestrian interactions. *Physical review letters* 113, 23 (2014), 238701.

Ioannis Karamouzas, Nick Sohre, Ran Hu, and Stephen J Guy. 2018. Crowd space: a predictive crowd analysis technique. *ACM Transactions on Graphics (TOG)* 37, 6 (2018), 1–14.

Robert Lubaś, Jarosław Wąs, and Jakub Porzycki. 2016. Cellular Automata as the basis of effective and realistic agent-based models of crowd behavior. *The Journal of Supercomputing* 72 (06 2016). doi:10.1007/s11227-016-1718-7

Jan Mandel. 2009. A Brief Tutorial on the Ensemble Kalman Filter. arXiv:0901.3725 [physics.ao-ph] https://arxiv.org/abs/0901.3725

Stefano Pellegrini, Andreas Ess, Konrad Schindler, and Luc Van Gool. 2009. You'll never walk alone: Modeling social behavior for multi-target tracking. In *2009 IEEE 12th international conference on computer vision*. IEEE, 261–268.

Timothy Sauer. 2012. *Numerical Analysis* (2 ed.). Pearson.

Jur Van Den Berg, Stephen J Guy, Ming Lin, and Dinesh Manocha. 2011. Reciprocal n-body collision avoidance. In *Robotics Research: The 14th International Symposium ISRR*. Springer, 3–19.

Jur Van den Berg, Ming Lin, and Dinesh Manocha. 2008. Reciprocal velocity obstacles for real-time multi-agent navigation. In *2008 IEEE international conference on robotics and automation*. Ieee, 1928–1935.

Francesco Zanlungo, Tetsushi Ikeda, and Takayuki Kanda. 2011. Social force model with explicit collision prediction. *Europhysics Letters* 93, 6 (2011), 68005.

Appendix

Table A1: Time and Position of Both Agents

Time	x_1	y_1	x_2	y_2
0.33	0.19	1.36	9.69	1.36
0.67	0.19	1.35	9.69	1.35
1.00	0.17	1.36	9.69	1.35
1.33	0.19	1.36	9.69	1.34
1.67	0.21	1.36	9.69	1.34
2.00	0.25	1.38	9.67	1.35
2.33	0.33	1.42	9.62	1.35
2.67	0.52	1.44	9.42	1.36
3.00	0.81	1.42	9.17	1.38
3.33	1.15	1.47	8.90	1.38
3.67	1.48	1.51	8.60	1.34
4.00	1.90	1.53	8.29	1.35
4.33	2.29	1.54	7.98	1.35
4.67	2.71	1.61	7.58	1.32
5.00	3.10	1.63	7.23	1.34
5.33	3.54	1.65	6.85	1.34
5.67	3.96	1.70	6.46	1.31
6.00	4.40	1.72	6.10	1.29
6.33	4.85	1.72	5.73	1.30
6.67	5.29	1.76	5.33	1.31
7.00	5.73	1.76	4.94	1.30
7.33	6.19	1.72	4.56	1.35
7.67	6.60	1.71	4.19	1.35
8.00	7.06	1.69	3.77	1.35
8.33	7.46	1.62	3.35	1.40
8.67	7.90	1.58	2.98	1.40
9.00	8.33	1.56	2.60	1.40
9.33	8.73	1.48	2.23	1.43
9.67	9.10	1.42	1.85	1.45
10.00	9.44	1.40	1.50	1.47
10.33	9.75	1.35	1.17	1.45
10.67	9.94	1.31	0.81	1.48

Table A2: Position and Velocities of Both Agents in x direction

x_1	v_{x1}	v_{x2}
0.19	0.03	0.00
0.21	0.04	-0.01
0.25	0.08	-0.03
0.33	0.20	-0.18
0.52	0.37	-0.36
0.81	0.48	-0.39
1.15	0.50	-0.42
1.48	0.57	-0.46
1.90	0.62	-0.46
2.29	0.61	-0.54
2.71	0.61	-0.57
3.10	0.63	-0.54
3.54	0.64	-0.59
3.96	0.64	-0.56
4.40	0.68	-0.54
4.85	0.67	-0.58
5.29	0.65	-0.60
5.73	0.68	-0.58
6.19	0.65	-0.55
6.60	0.66	-0.59
7.06	0.64	-0.64
7.46	0.62	-0.59
7.90	0.67	-0.56
8.33	0.63	-0.56
8.73	0.58	-0.57
9.10	0.53	-0.55
9.44	0.49	-0.51
9.75	0.48	-0.50
9.94	0.47	-0.49