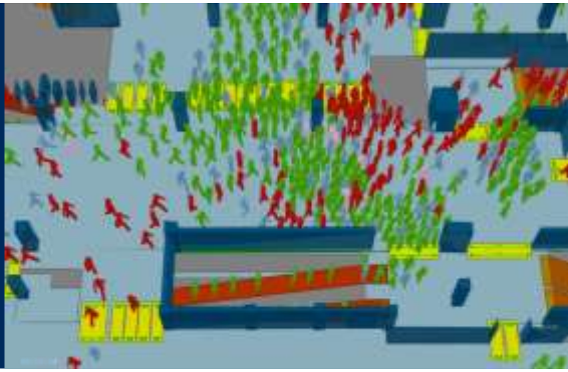
A Comparative Analysis of Agent-Based Pedestrian Modelling Approaches

Gregory Hoy – MASc Student

Supervisor: Professor Amer Shalaby

September 15th, 2017



UNIVERSITY OF TORONTO
FACULTY OF APPLIED SCIENCE & ENGINEERING
Transportation Research Institute

Agenda

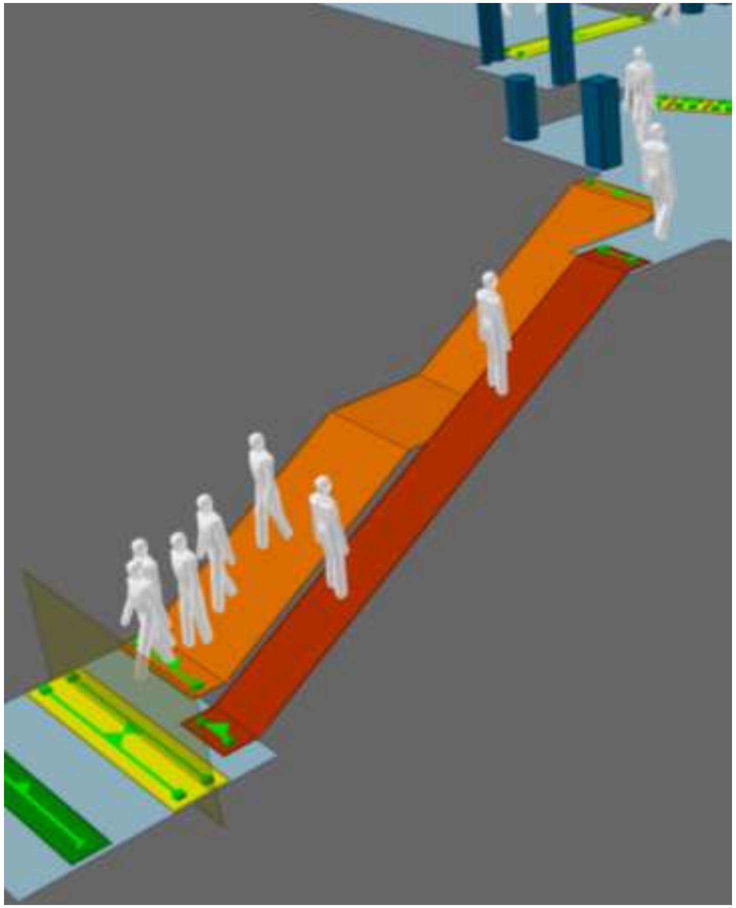
- Introduction
- Context and Motivation
- Project Methodology
- Data Collection
- Model Development
- Calibration
- Validation and Comparison
- Conclusions and Future Work

Introduction

- Pedestrian simulation is a powerful tool for evaluating major pedestrian facilities
- Many models have been developed in this field, from simple tools to commercial software
- In practice, these models have been used to improve the design, operation, and safety of transit stations, event venues, and religious sites

Introduction

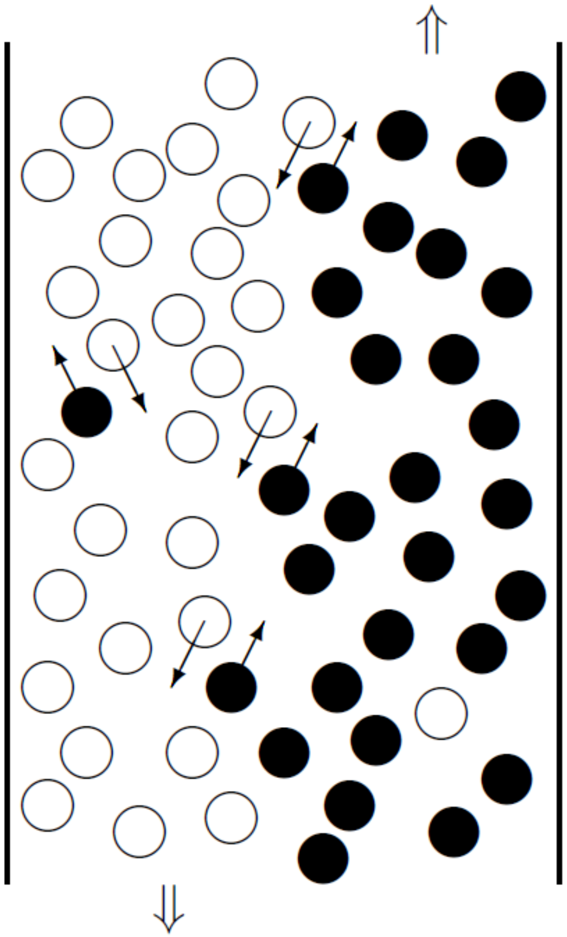
- Recent advances in pedestrian modelling have created more complex models, both for research and commercial use
- While powerful, these models can be slow and computationally demanding



Introduction

- Older and less complex models are still thought to be useful

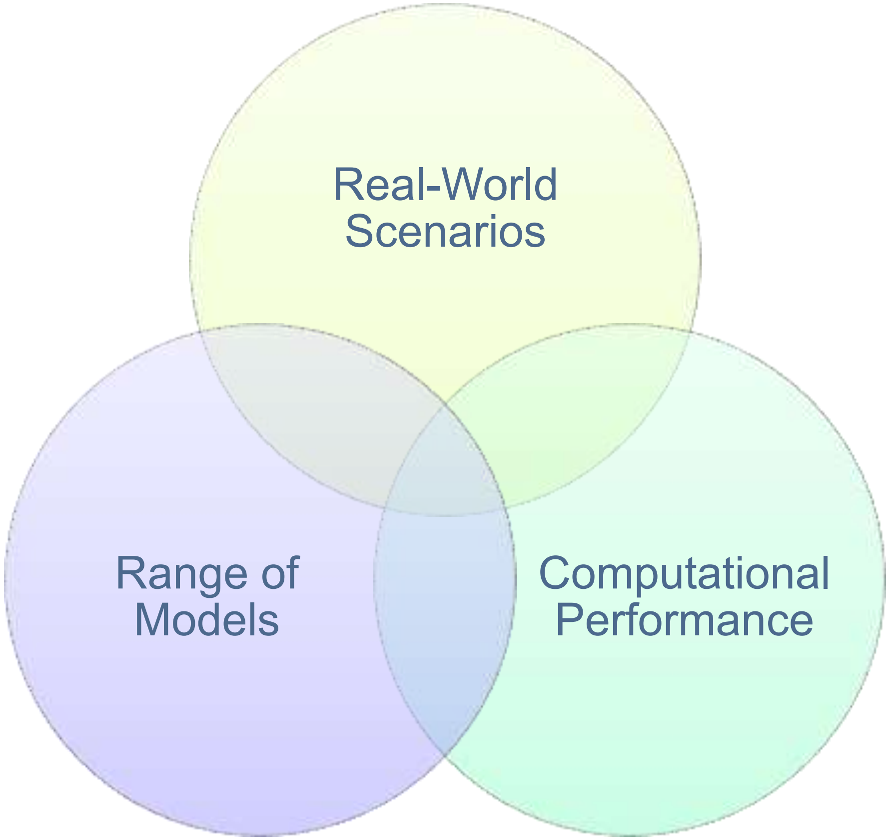
- Little has been done to compare their accuracy and performance against modern competitors



Introduction

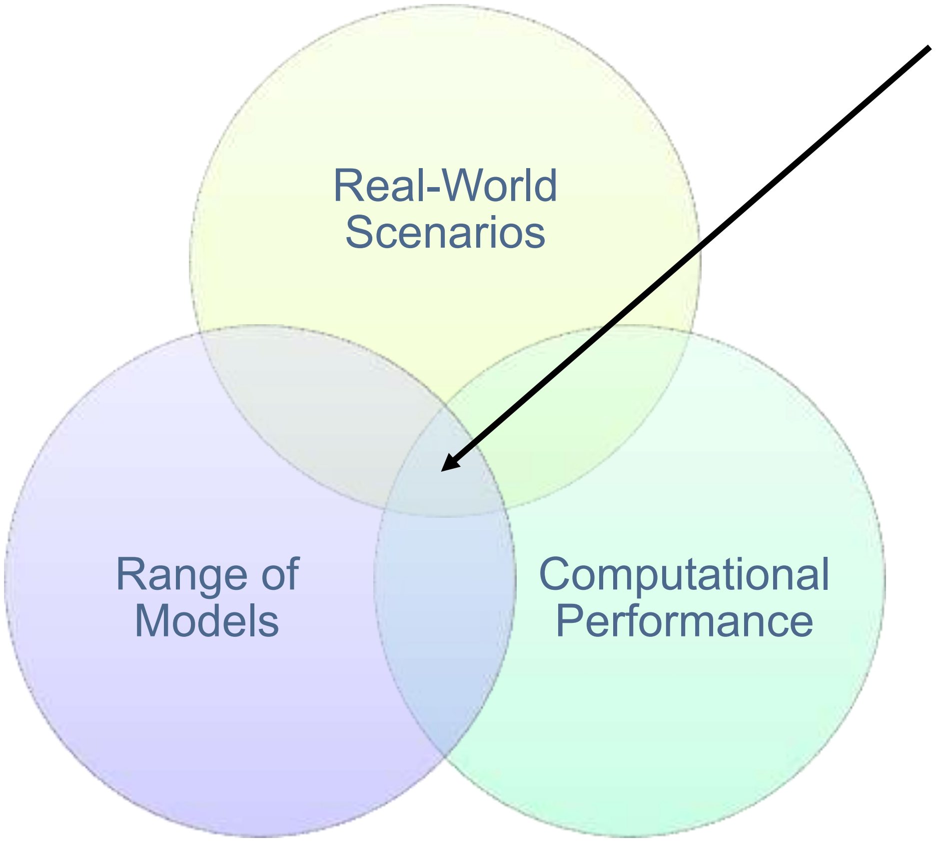
- A wide variety of models exist, ranging from macroscopic to microscopic approaches
- Some ‘traditional’ methods include fluid flow approximation, graph/ network models, Cellular Automata, and Social Forces
- Contemporary methods include hybrids (e.g. Optimal Steps) and new developments (e.g. Gradient Navigation, group dynamics)

Introduction



- Existing research has touched on these areas separately when comparing models
- Often, models are all from the same family (microscopic, CA, etc.)
- Computational performance is often ignored

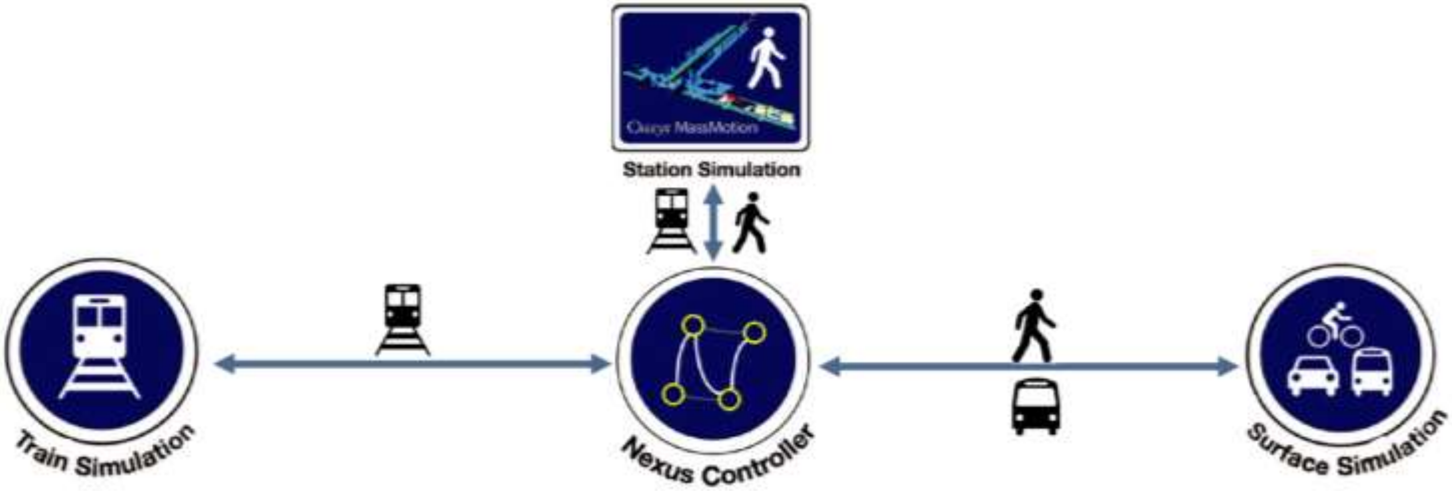
Introduction



- This research combines all three elements to fill this knowledge gap

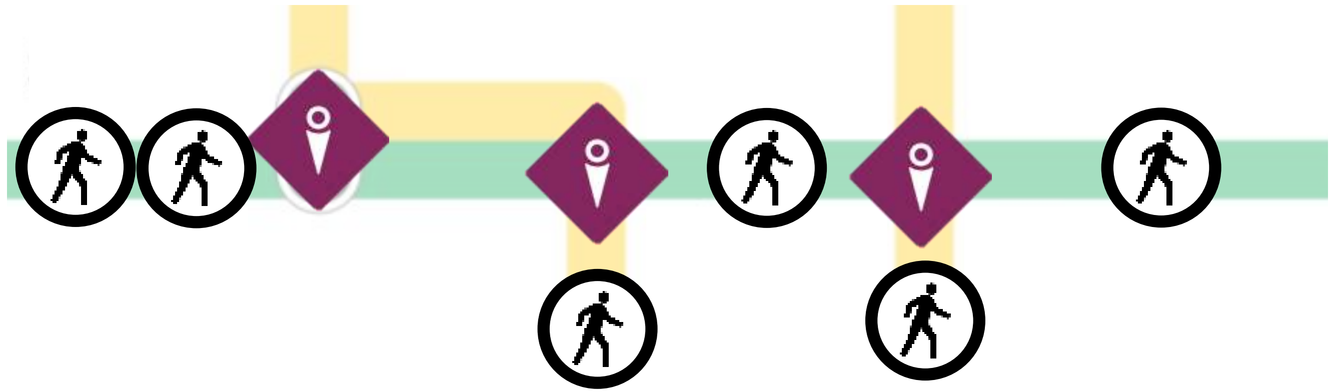
Context and Motivation

- The Nexus platform connects rail, surface, and pedestrian simulation to model transit networks
- Pedestrian simulation is an important part of the platform, but it is often the slowest component



Context and Motivation

- Currently, MassMotion is the simulator of choice for all station models, but it is demanding in terms of data, computer resources, and time
- The ideal solution is to use a simplified model for smaller and less complex facilities



Context and Motivation

- Without existing comparisons of simple and more advanced models, how can we pick?
- How do agent behaviours differ between models, and how does this affect results?
- More importantly, how do we know whether a new model will improve computational performance?

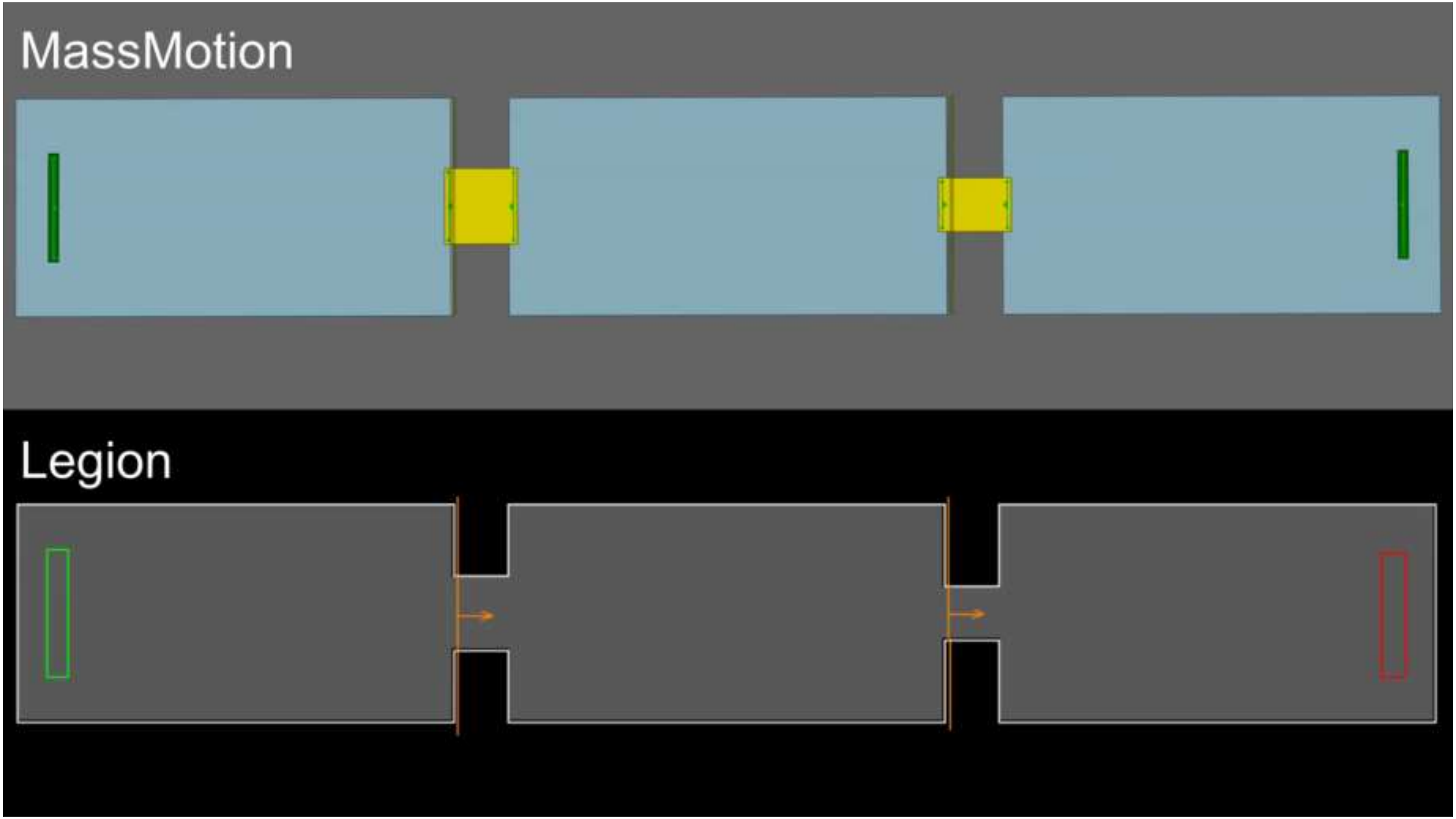
Context and Motivation

- Performing simple tests with MassMotion and Legion illustrates how two similar tools can produce surprisingly different behaviours
- If two advanced models cannot agree on results, what can we expect from a range of models?

MassMotion

LEGION
SCIENCE IN MOTION

Context and Motivation



Project Methodology

We want to compare a set of pedestrian models by:

- Selecting models from the literature
- Collecting real-world data from transit stations
- Coding the models to interface with MassMotion
- Calibrating the models
- Testing the models using a number of scenarios

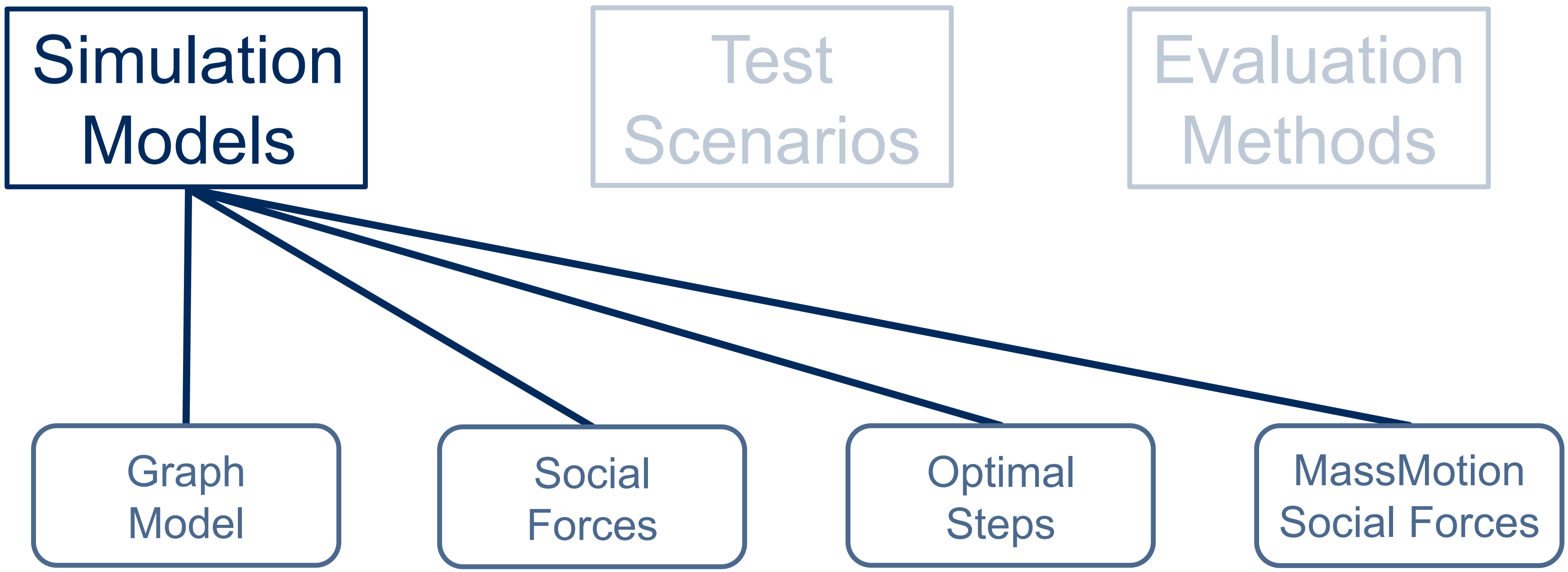
Project Methodology

Simulation Models

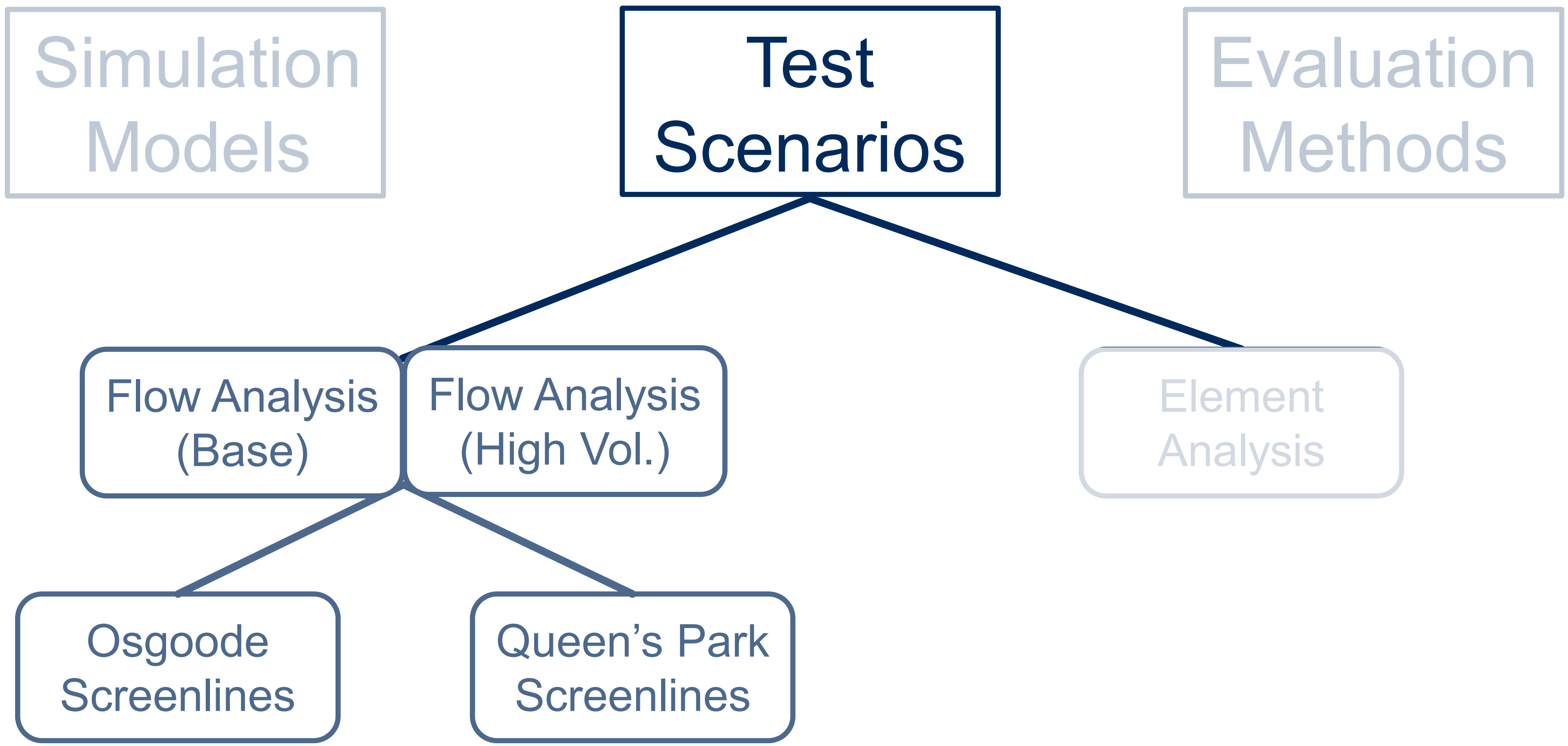
Test Scenarios

Evaluation Methods

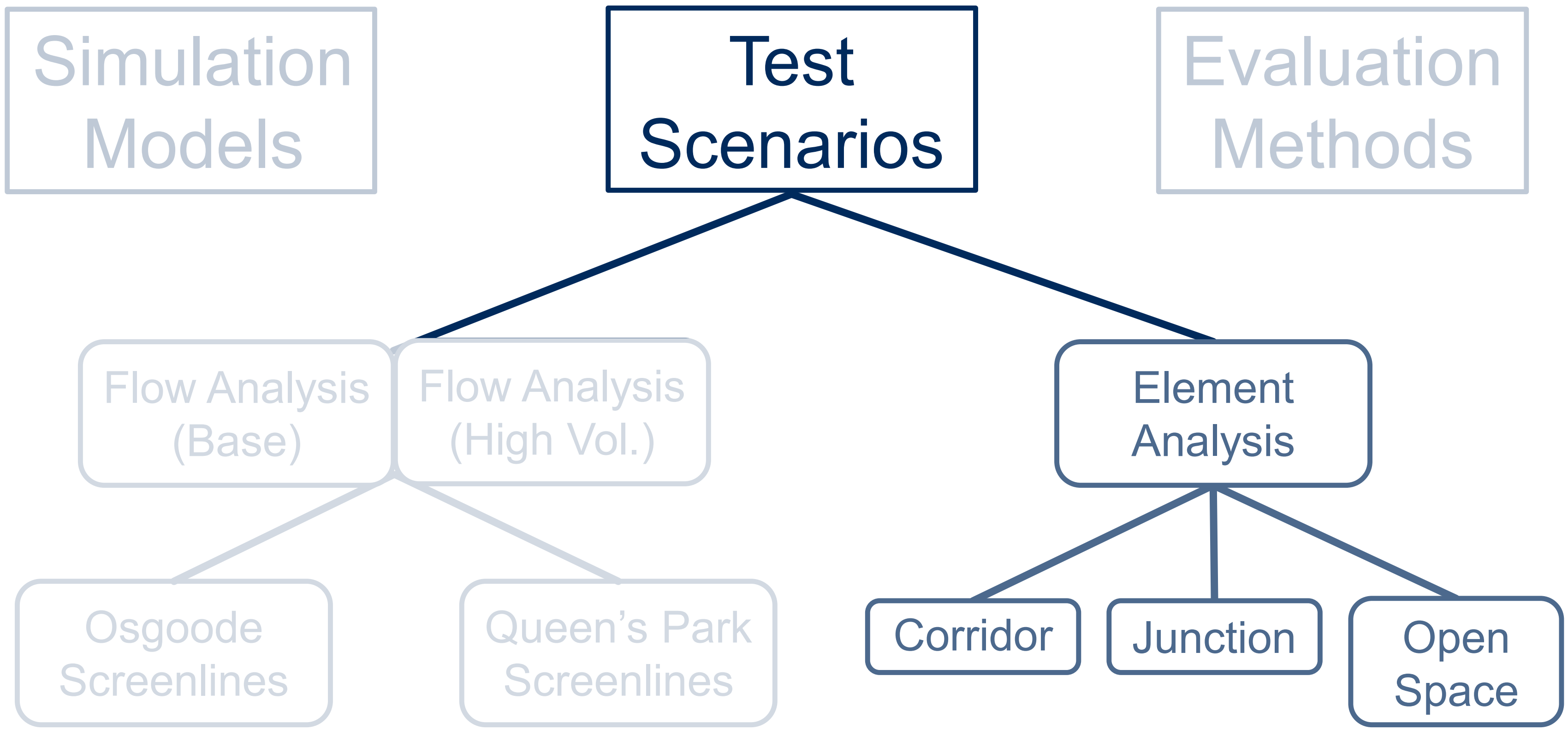
Project Methodology



Project Methodology



Project Methodology



Project Methodology

Simulation Models

Test Scenarios

Evaluation Methods

- Screenline Flows
- Agent Flow Paths
- Fruin Level of Service
- Simulation Run Times

Project Methodology

Simulation Models	Test Scenarios	Evaluation Methods

- By running each model for each scenario, we can evaluate performance in a variety of ways
- These evaluation methods can be used to identify limits to each model's accurate operation

Data Collection

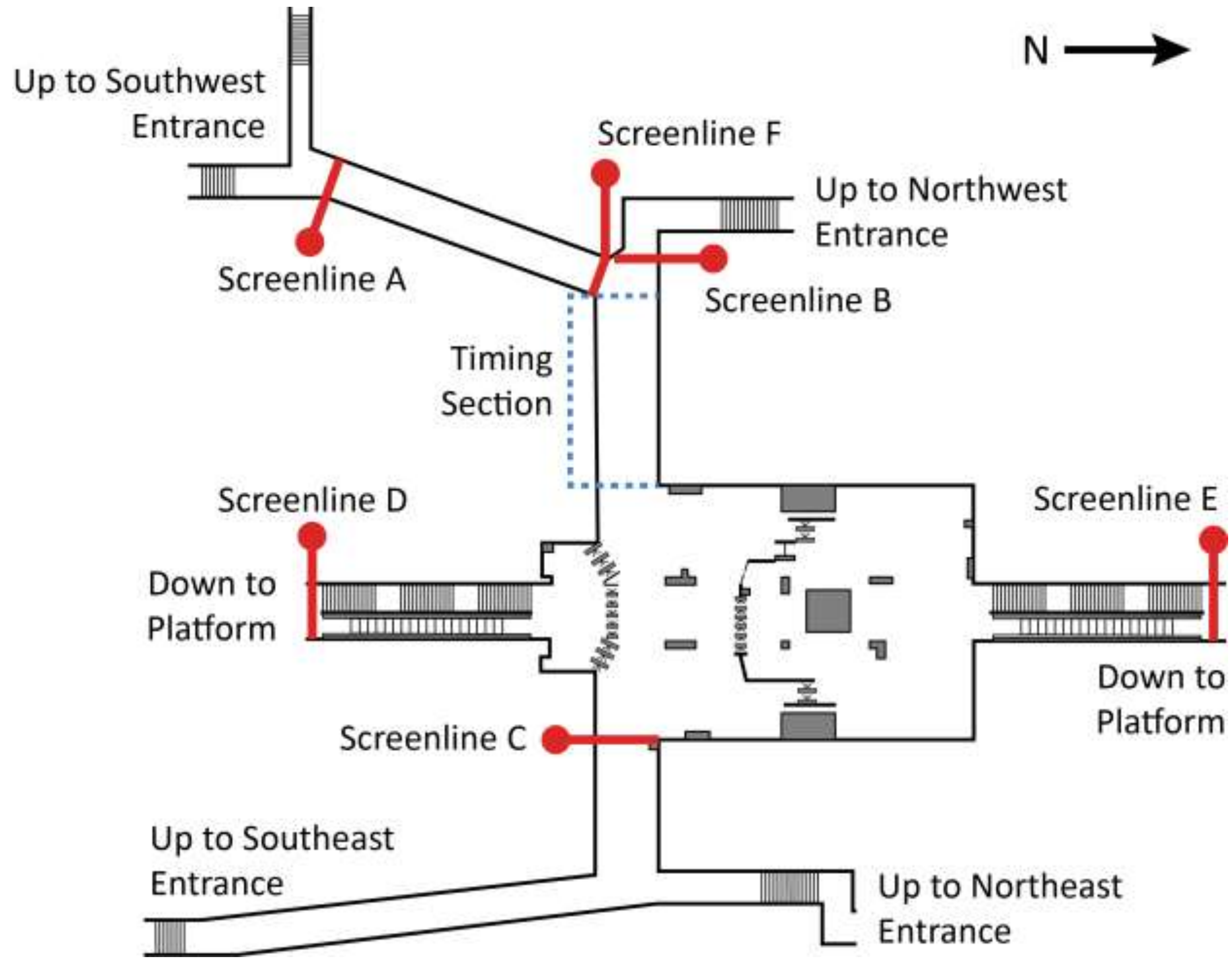
- To simulate each scenario, we need complete station geometry as well as data detailing how pedestrians move through the space

Data Collection

- Stations with moderate peak volumes and geometry of interest were selected

Data Collection

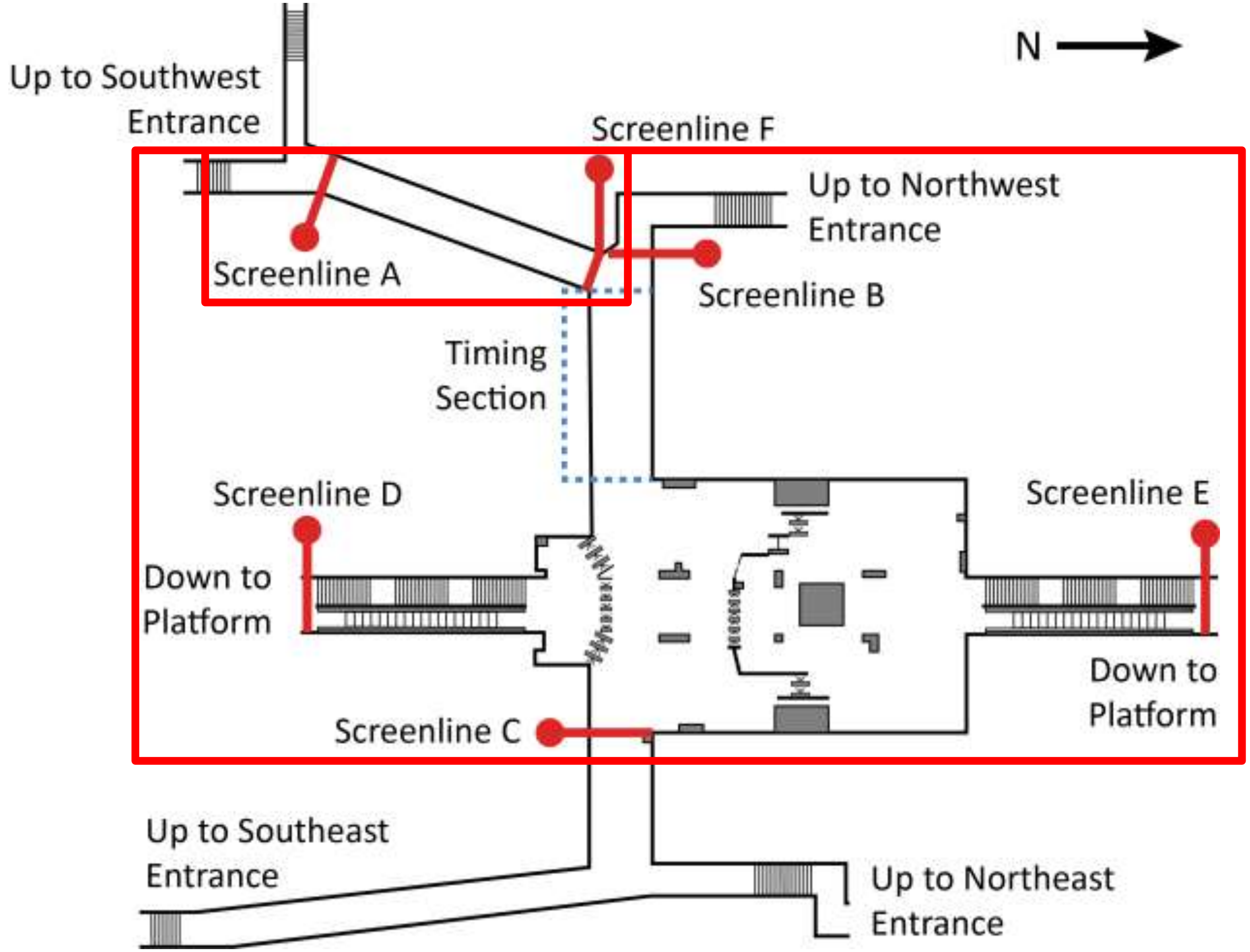
Osgoode



Data Collection

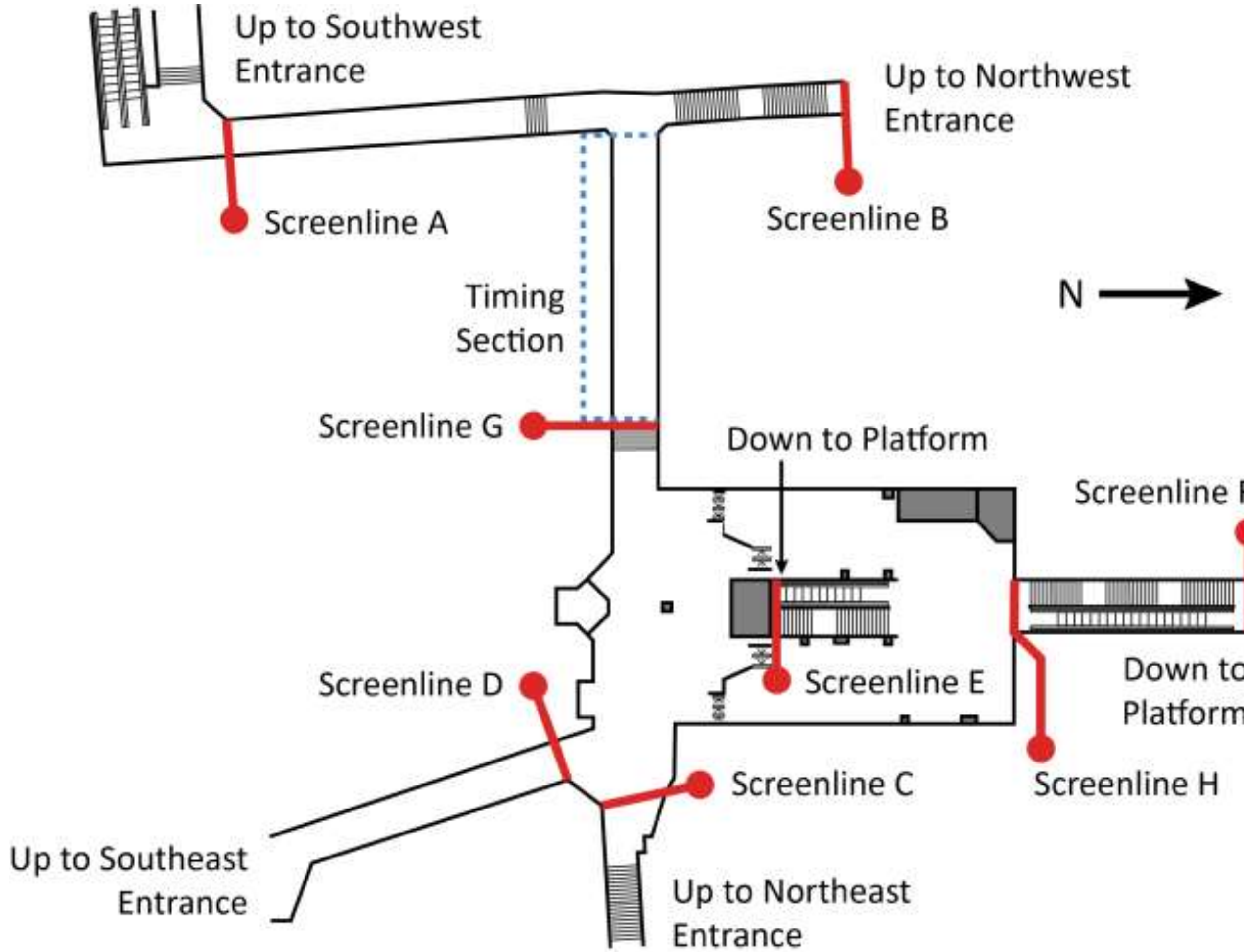
Osgoode

- Straight Corridor
- Complete Concourse



Data Collection

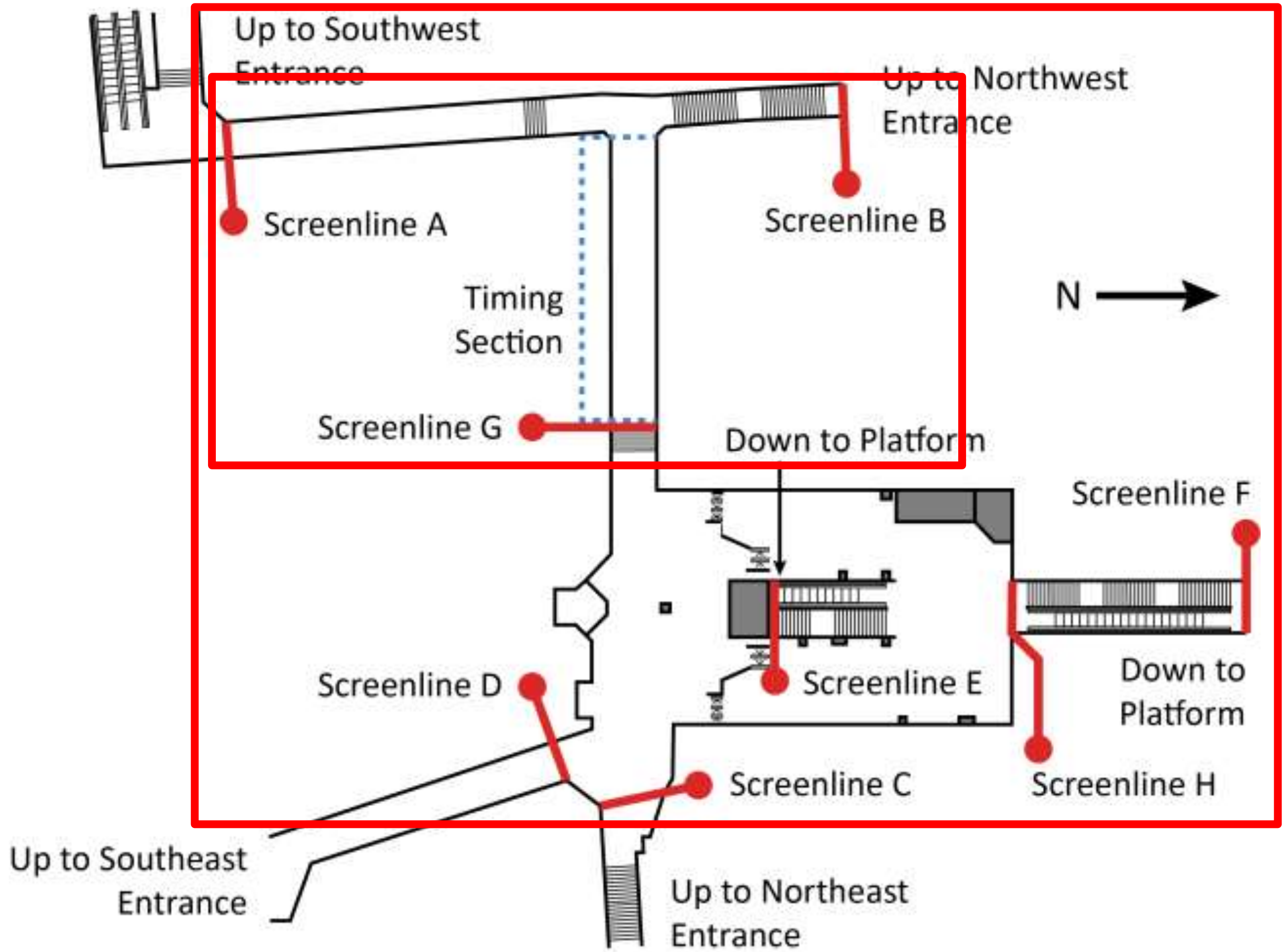
Queen's Park



Data Collection

Queen's Park

- T-Junction
- Complete Concourse

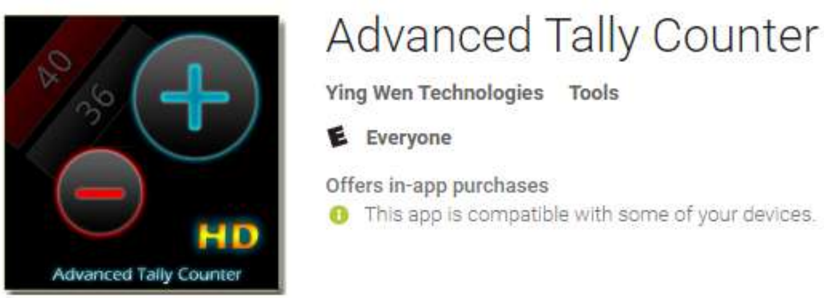


Data Collection

- A team of 10 students and researchers collected data at the stations
- Each collection lasted approximately 1.5 hours:
  - 60 minutes of continuous counting,
  - 20 minutes of simultaneous timings, and
  - 20 minutes of measurements
- Collected data was used to make origin-destination matrices for each scenario

Data Collection

- Timestamped pedestrian counts were taken at all screenline locations shown on maps
- Bidirectional flows were recorded using smartphone applications (Android/iOS)

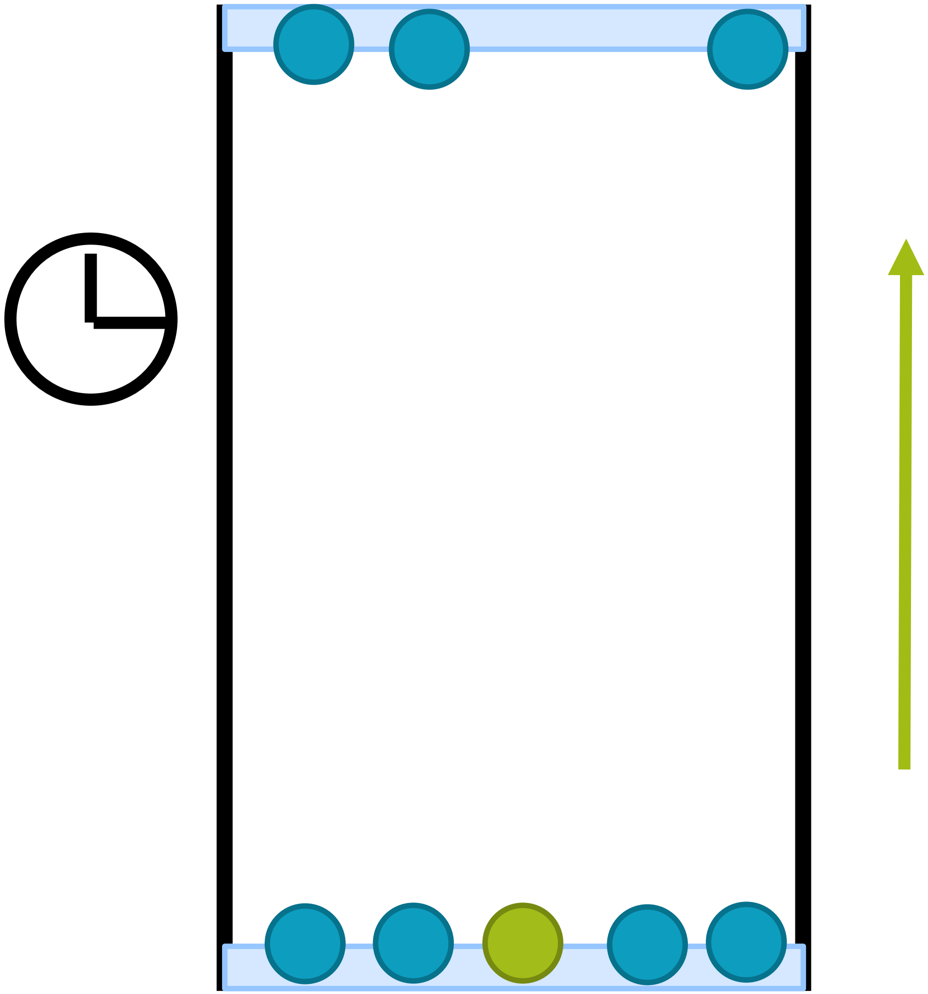


Model Development

- To test and compare model types, a set of representative methods were selected
- Each specific method was coded in C# to control agent movement via the MassMotion SDK
- This allows each method to be compared against others with identical external parameters:
  - Cost mapping,
  - Model geometry,
  - Agent attributes, etc.

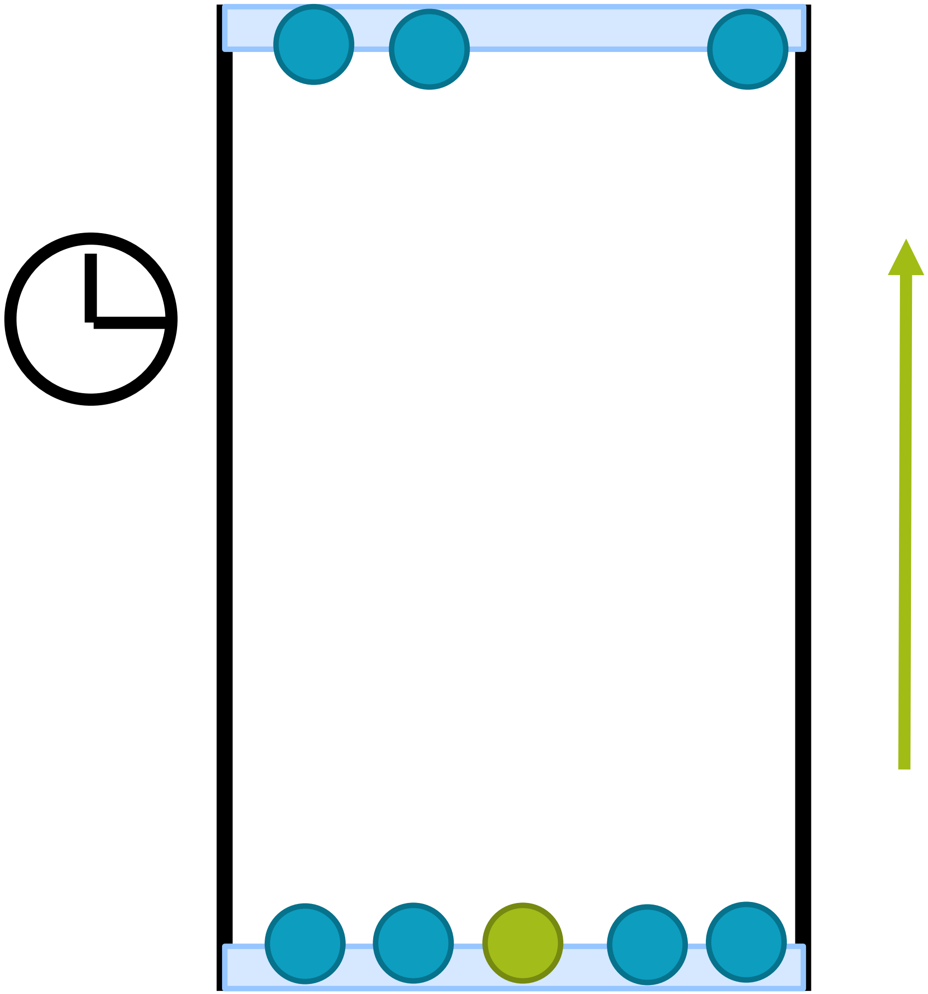
Model Development – Graph Model

- A mesoscopic approach representing each model as a set of links and nodes



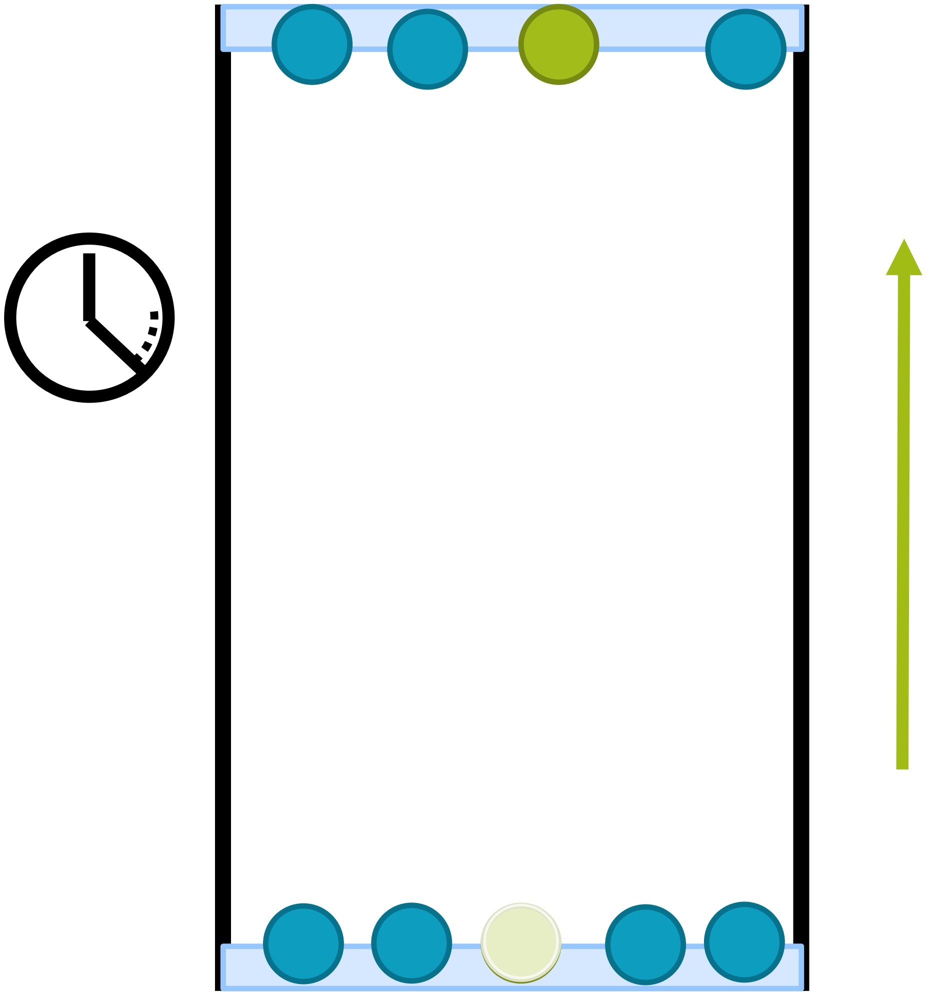
Model Development – Graph Model

- A mesoscopic approach representing each model as a set of links and nodes
- Agents jump between waypoints based on the distance and their speed



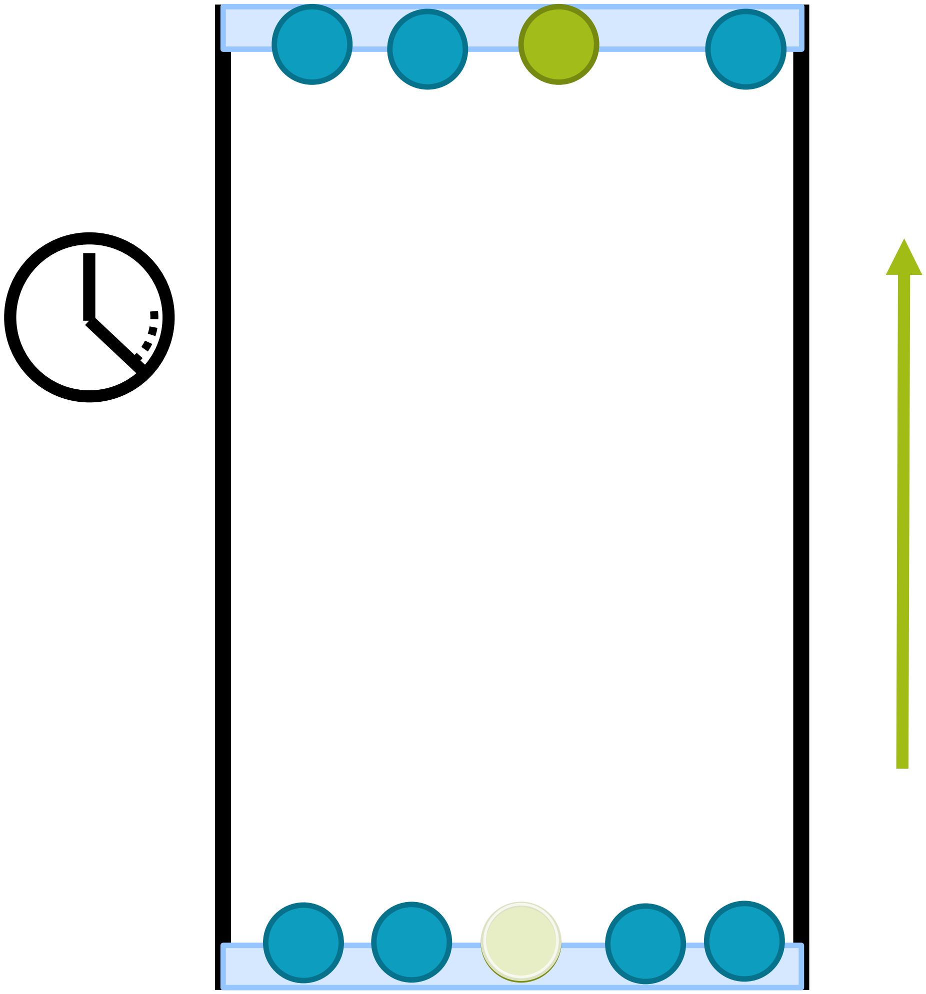
Model Development – Graph Model

- A mesoscopic approach representing each model as a set of links and nodes
- Agents jump between waypoints based on the distance and their speed



Model Development – Graph Model

- A mesoscopic approach representing each model as a set of links and nodes
- Agents jump between waypoints based on the distance and their speed
- Has limited analytical value, but runs quickly



Model Development – Social Forces

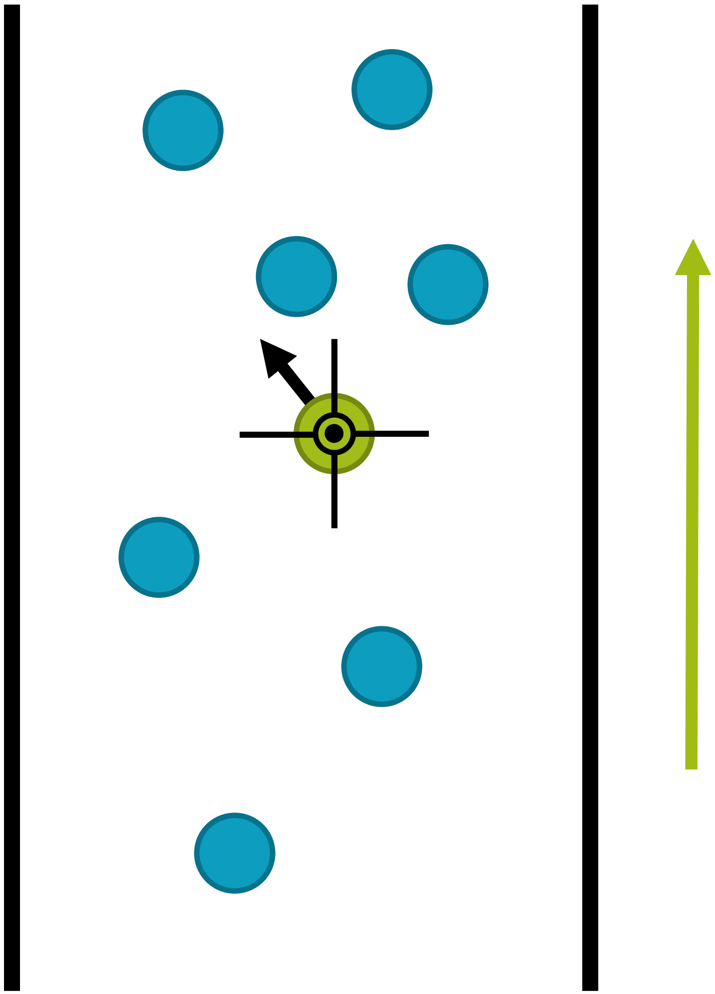
- A simple Social Forces method based on work by Helbing and Molnár (1995), with equations provided by Helbing and Johansson (2009)

$$\boldsymbol{f}_\alpha(t) = \frac{1}{\tau_\alpha}(v_\alpha^0 \boldsymbol{e}_\alpha^0 - \boldsymbol{v}_\alpha) + \sum_{\beta(\neq\alpha)} \boldsymbol{f}_{\alpha\beta}(t) + \sum_i \boldsymbol{f}_{\alpha i}(t)\,,$$

$$\boldsymbol{f}_{\alpha\beta}(\boldsymbol{d}_{\alpha\beta}) = A\mathrm{e}^{-b_{\alpha\beta}/B} \cdot \frac{\|\boldsymbol{d}_{\alpha\beta}\| + \|\boldsymbol{d}_{\alpha\beta} - \boldsymbol{y}_{\alpha\beta}\|}{2b_{\alpha\beta}} \cdot \frac{1}{2}\left(\frac{\boldsymbol{d}_{\alpha\beta}}{\|\boldsymbol{d}_{\alpha\beta}\|} + \frac{\boldsymbol{d}_{\alpha\beta} - \boldsymbol{y}_{\alpha\beta}}{\|\boldsymbol{d}_{\alpha\beta} - \boldsymbol{y}_{\alpha\beta}\|}\right) \tag{9}$$

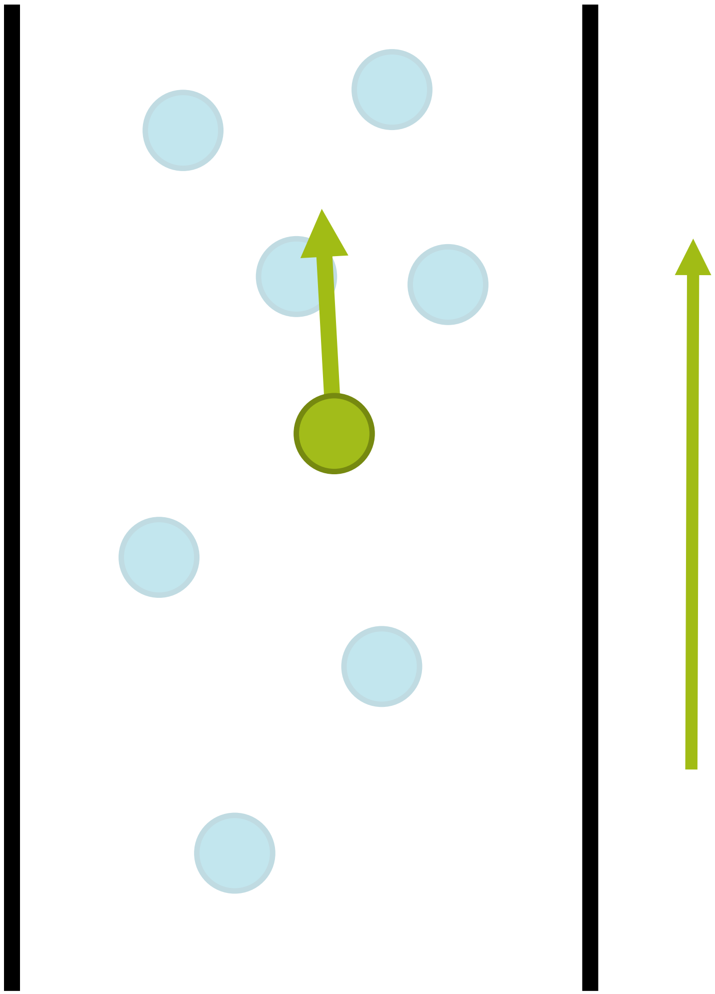
Model Development – Social Forces

- The agent's current location and velocity is queried



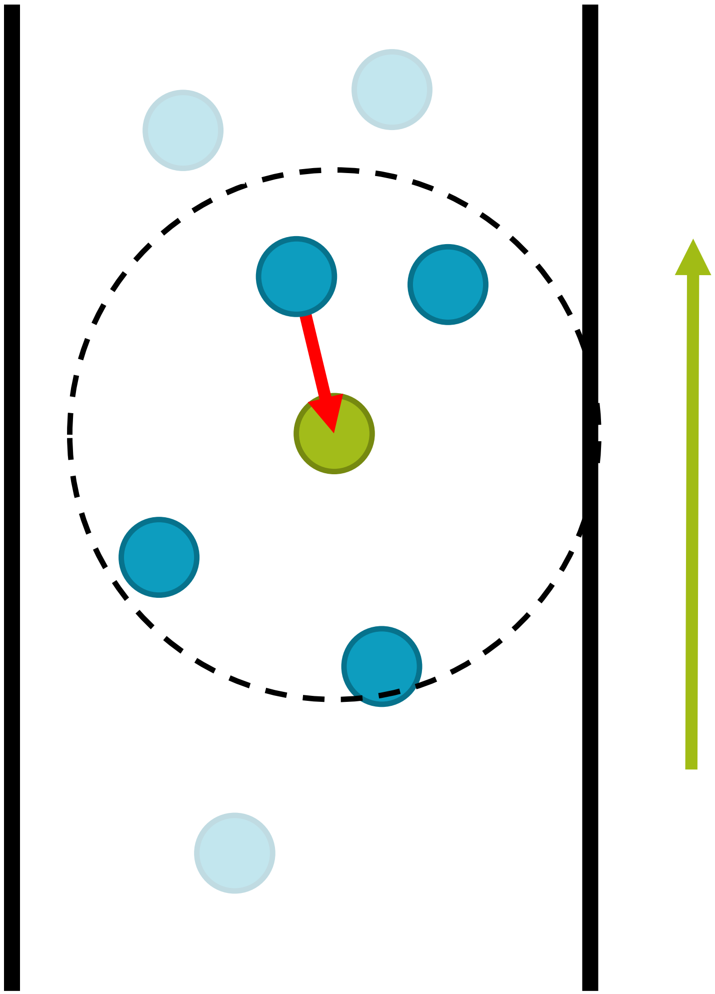
Model Development – Social Forces

- The agent's current location and velocity is queried
- The agent's attractive "goal force" is calculated based on direction to goal and desired velocity



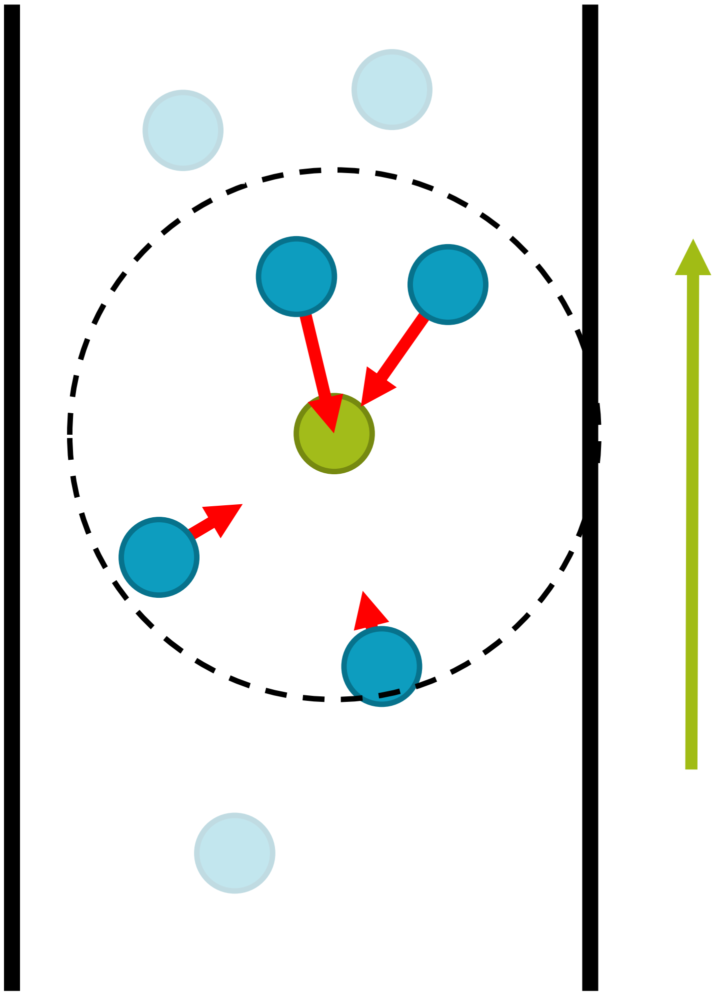
Model Development – Social Forces

- The agent's current location and velocity is queried
- The agent's attractive "goal force" is calculated based on direction to goal and desired velocity
- A repulsive "neighbour force" is calculated for every nearby agent based on distance and direction



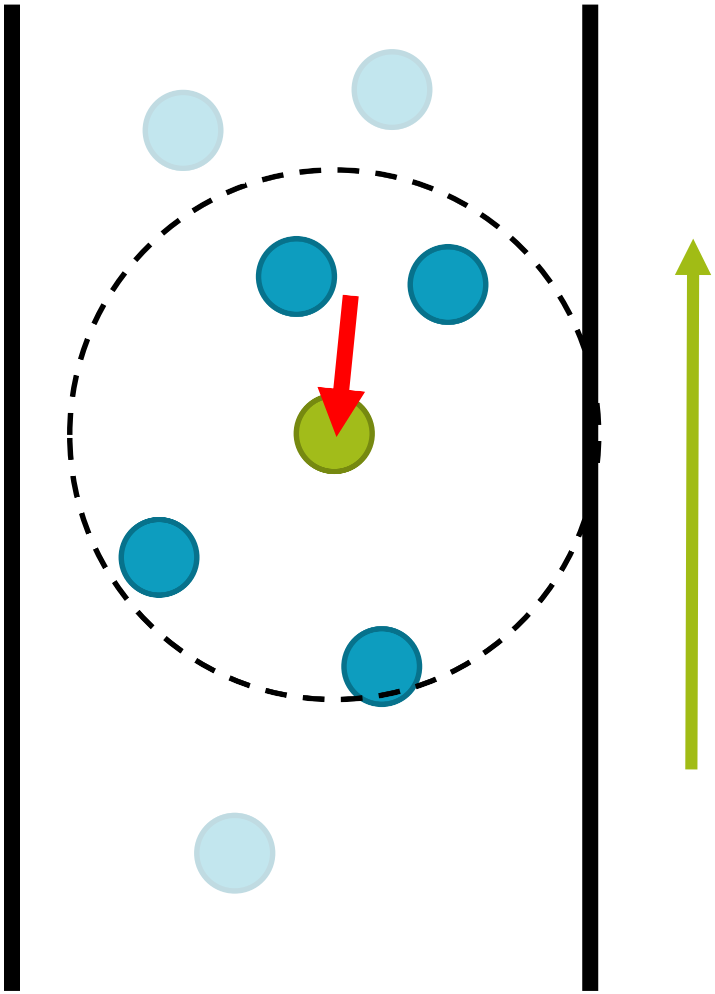
Model Development – Social Forces

- The agent’s current location and velocity is queried
- The agent’s attractive “goal force” is calculated based on direction to goal and desired velocity
- A repulsive “neighbour force” is calculated for every nearby agent based on distance and direction



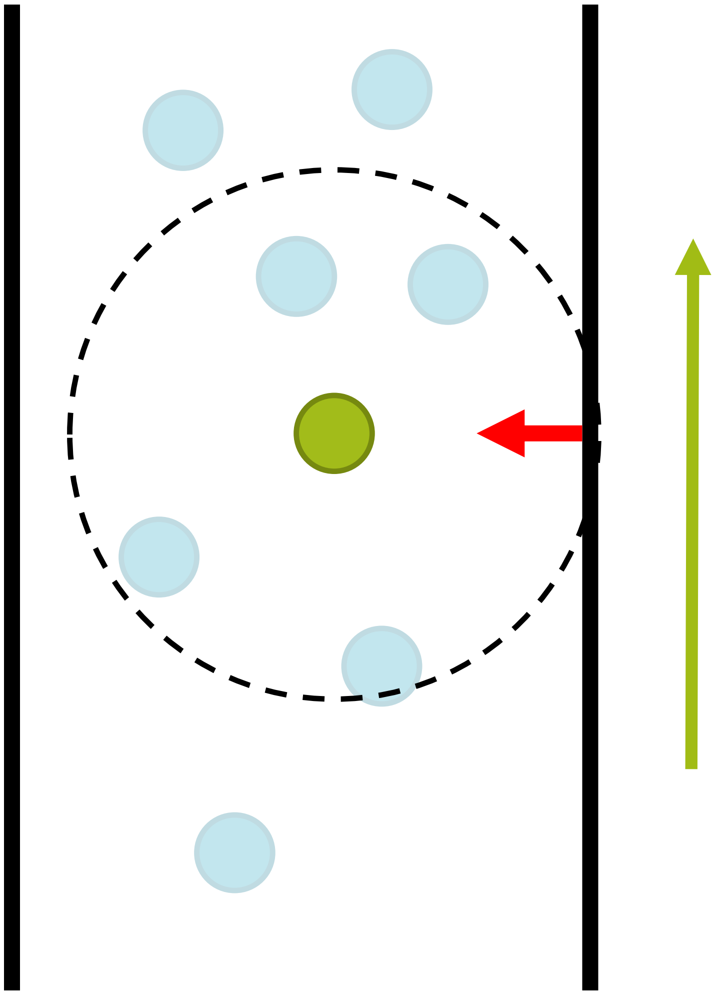
Model Development – Social Forces

- The neighbour forces are summed into a net neighbour force



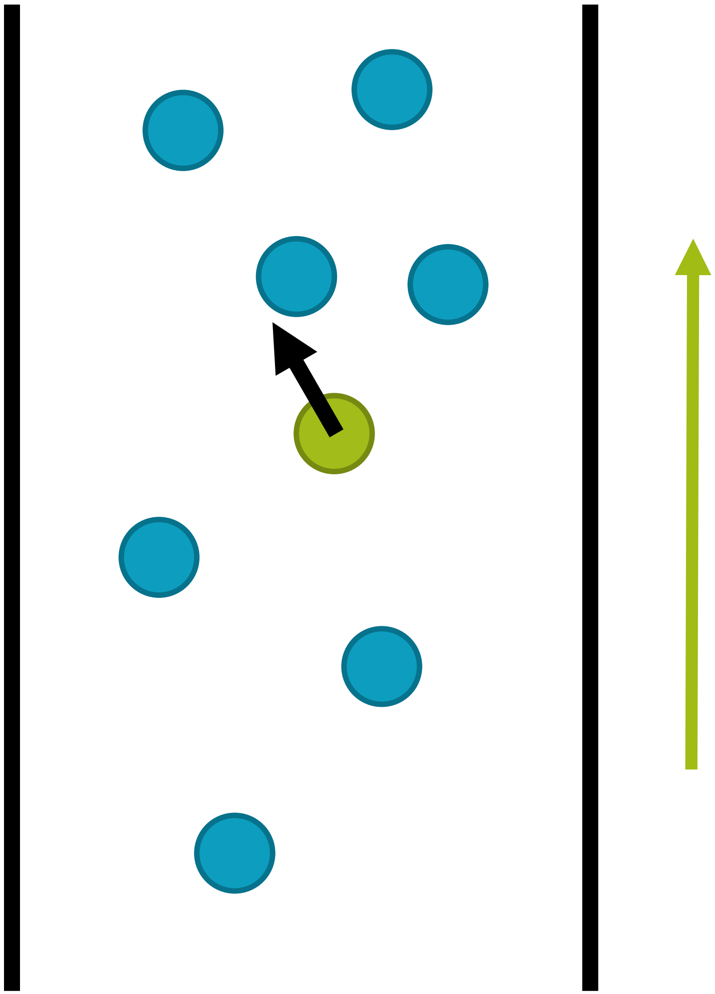
Model Development – Social Forces

- The neighbour forces are summed into a net neighbour force
- A repulsive "obstacle force" is calculated for the nearest obstacle based on distance and direction



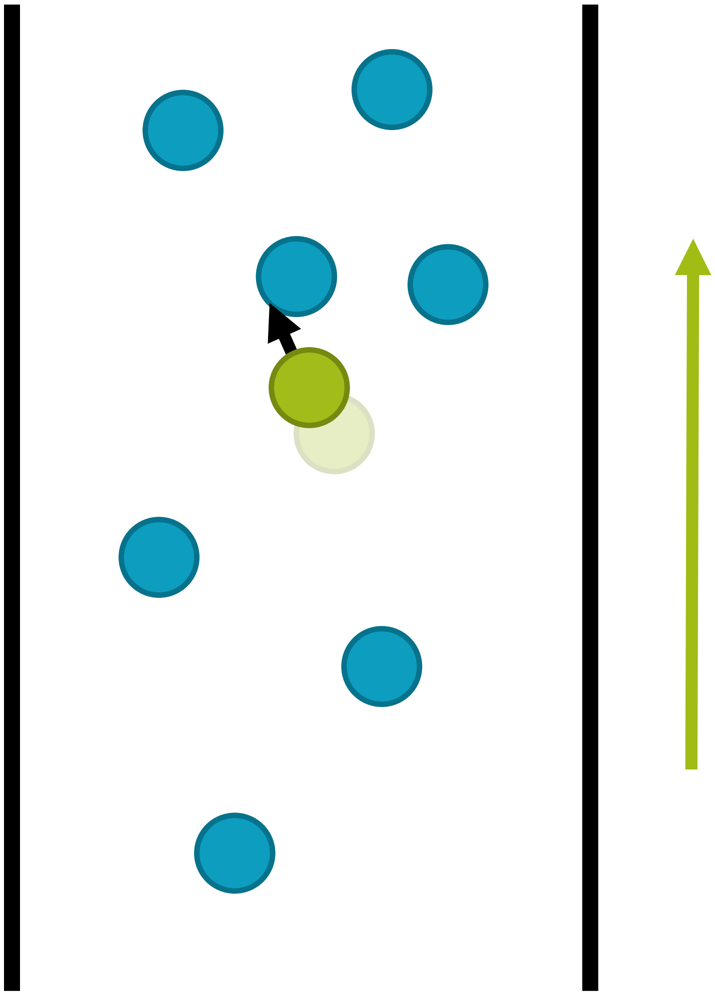
Model Development – Social Forces

- The neighbour forces are summed into a net neighbour force
- A repulsive "obstacle force" is calculated for the nearest obstacle based on distance and direction
- All forces are summed, resulting in a net force to be applied to the agent



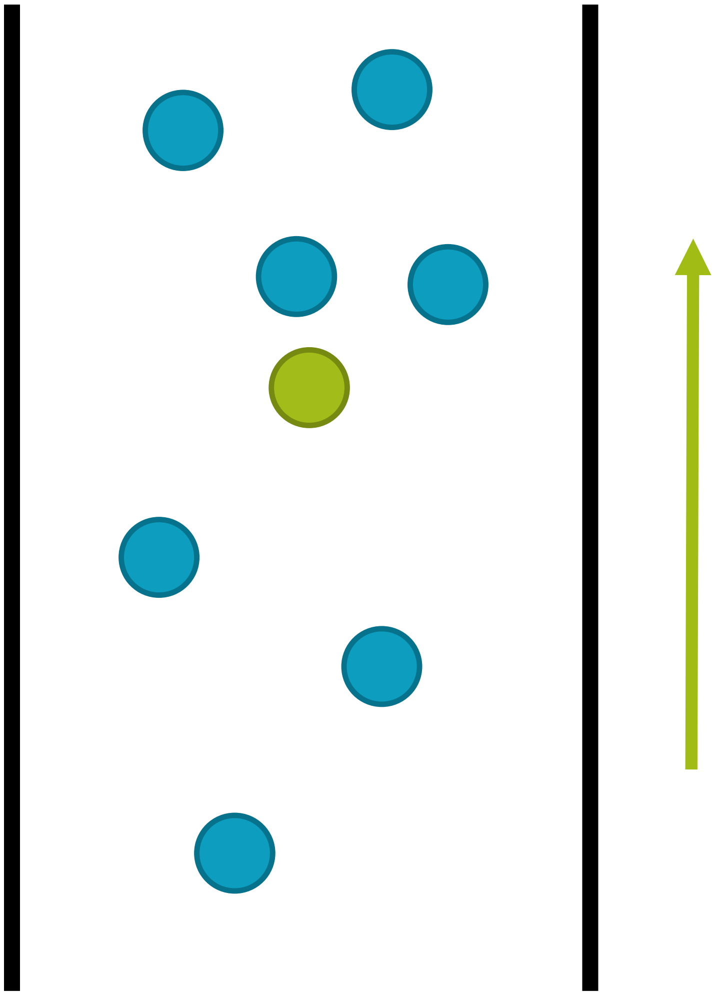
Model Development – Social Forces

- The agent's next velocity and position are calculated



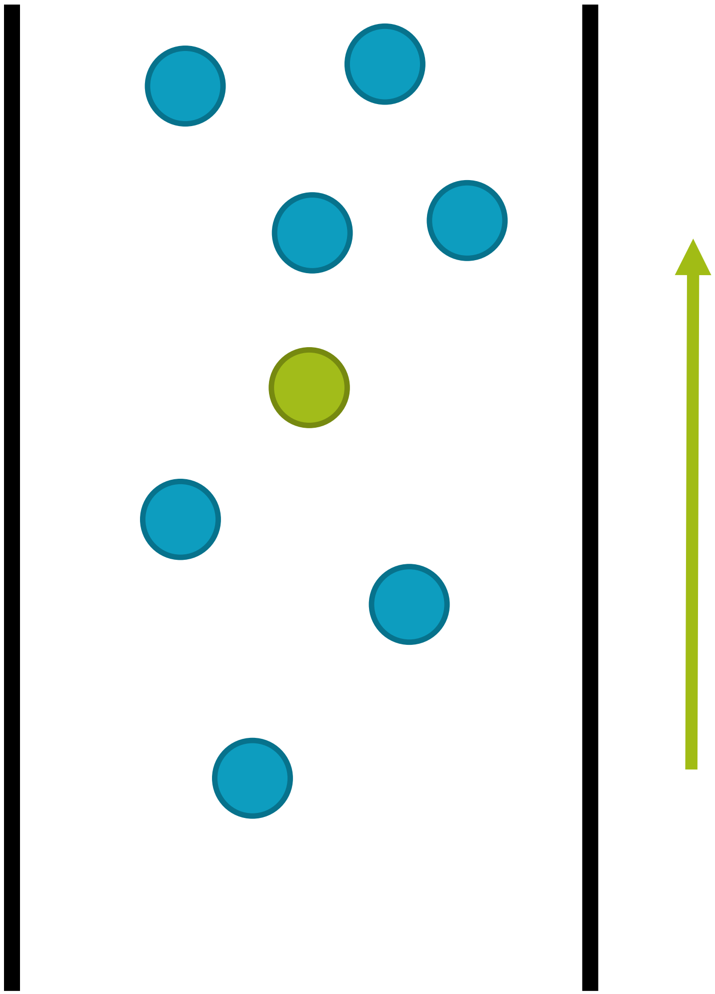
Model Development – Social Forces

- The agent's next velocity and position are calculated
- The agent is moved – if the move is invalid (e.g. off the floor), the agent is repositioned back onto the floor



Model Development – Social Forces

- The agent's next velocity and position are calculated
- The agent is moved – if the move is invalid (e.g. off the floor), the agent is repositioned back onto the floor
- The process is repeated with all other agents and the simulation is advanced by one step

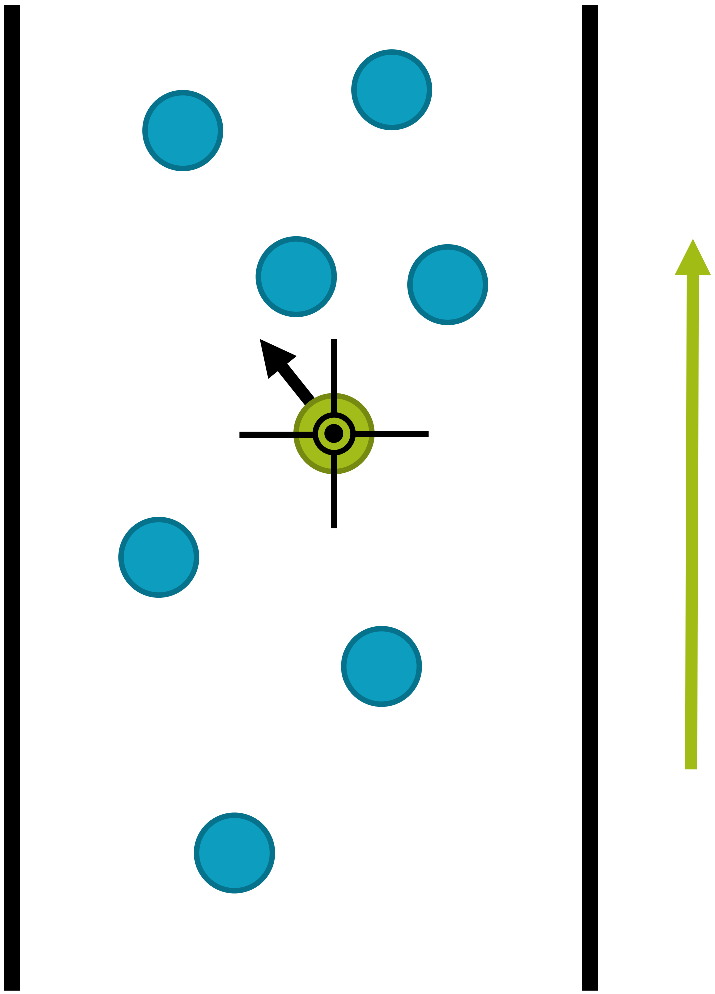


Model Development – Optimal Steps

- Optimal Steps is a hybrid method, combining the continuous space of Social Forces with discrete movements of Cellular Automata
- This specific implementation is based on the work of Seitz and Köster (2012)
- Modifications have been made to handle MassMotion's floor and link elements, which discretize walkable space

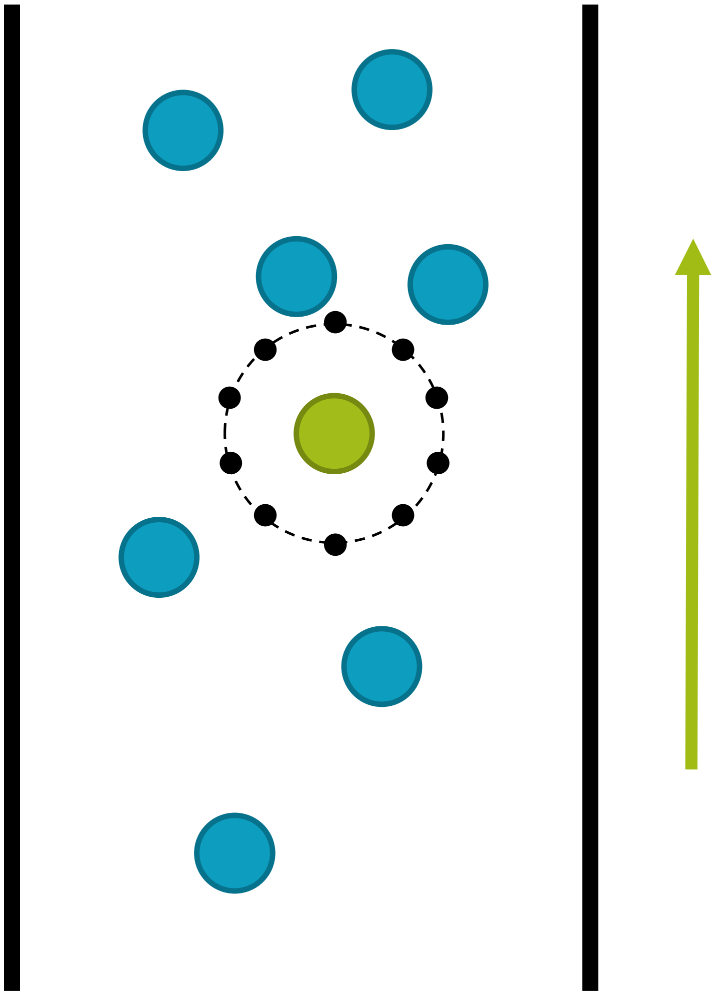
Model Development – Optimal Steps

- The agent's current location and velocity are queried



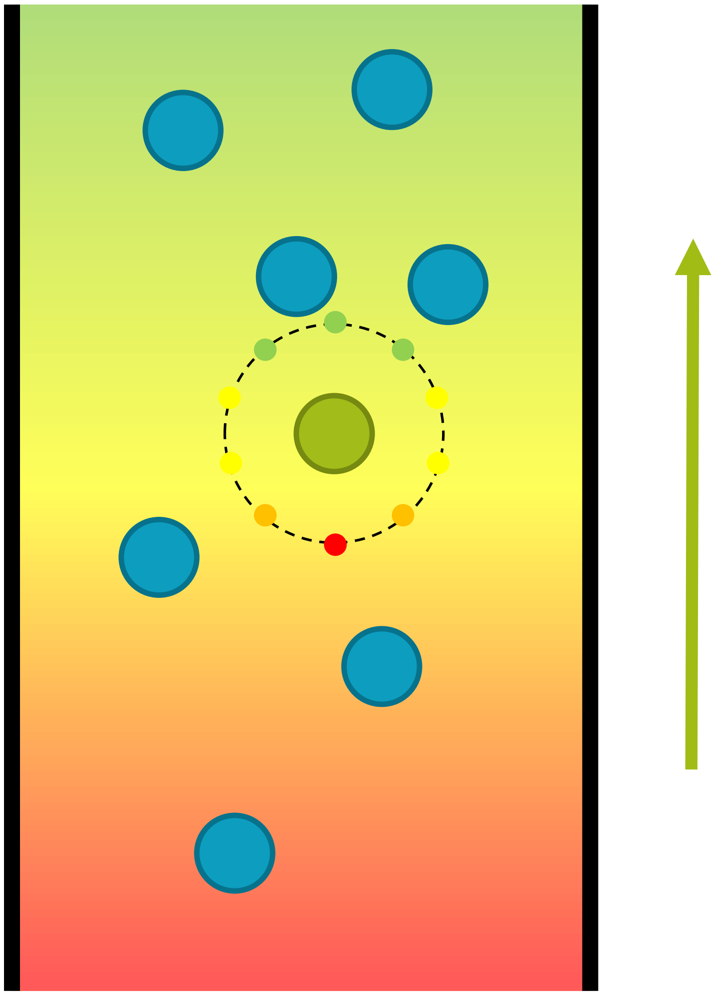
Model Development – Optimal Steps

- The agent’s current location and velocity are queried
- A corresponding step length is calculated, and potential positions are selected



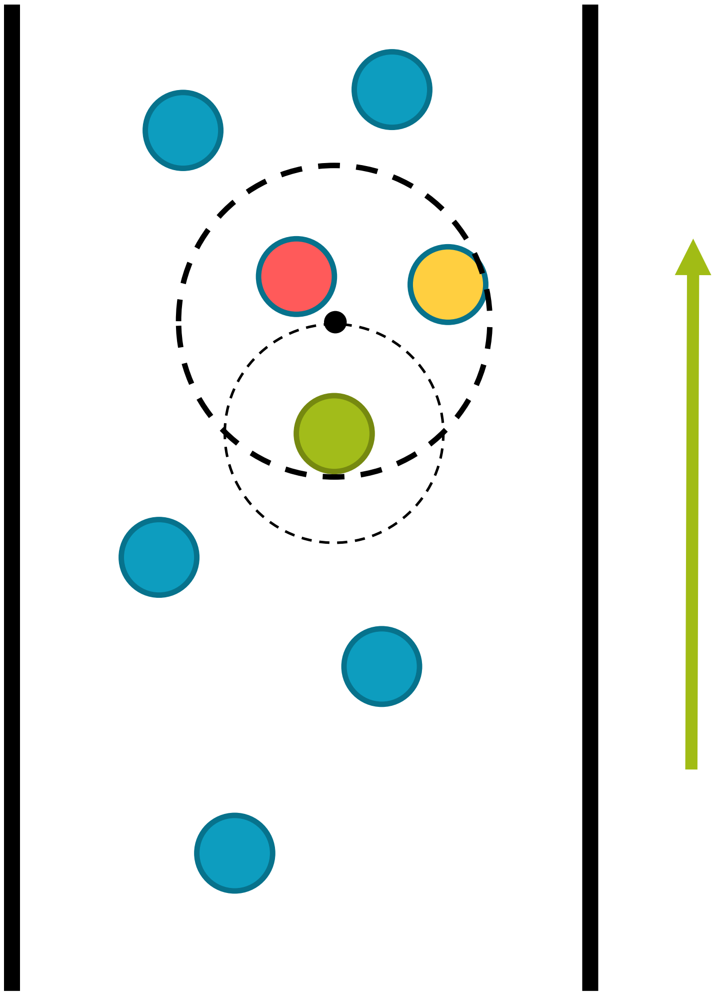
Model Development – Optimal Steps

- The agent's current location and velocity are queried
- A corresponding step length is calculated, and potential positions are selected
- Potential values are assigned to each position based distance to the goal



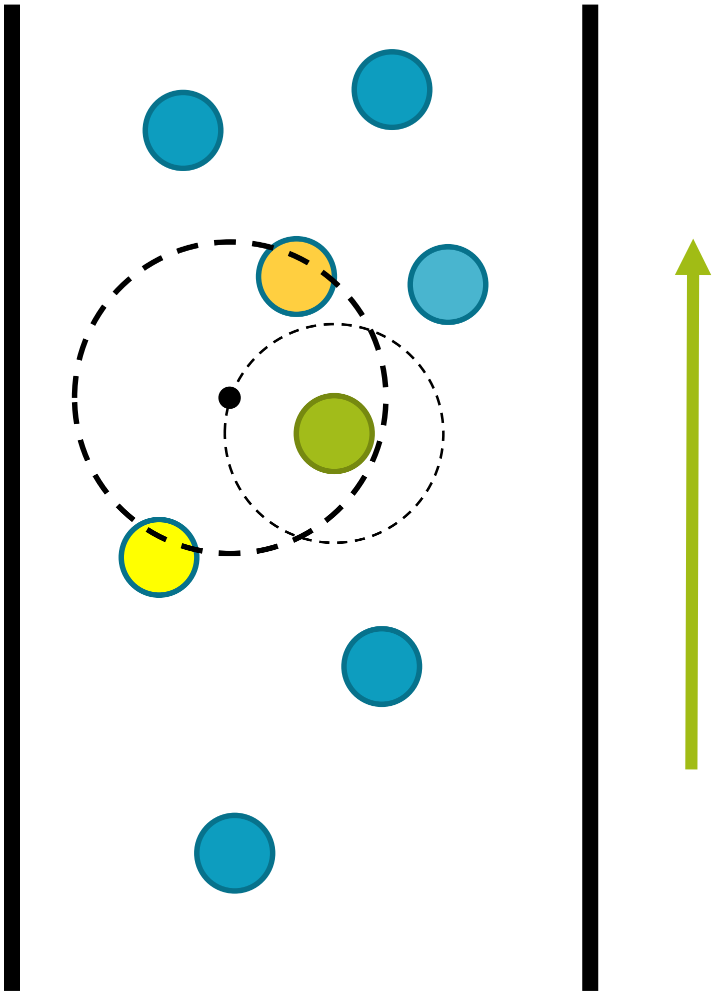
Model Development – Optimal Steps

- Neighbours near each position are checked – close neighbours make the position less desirable



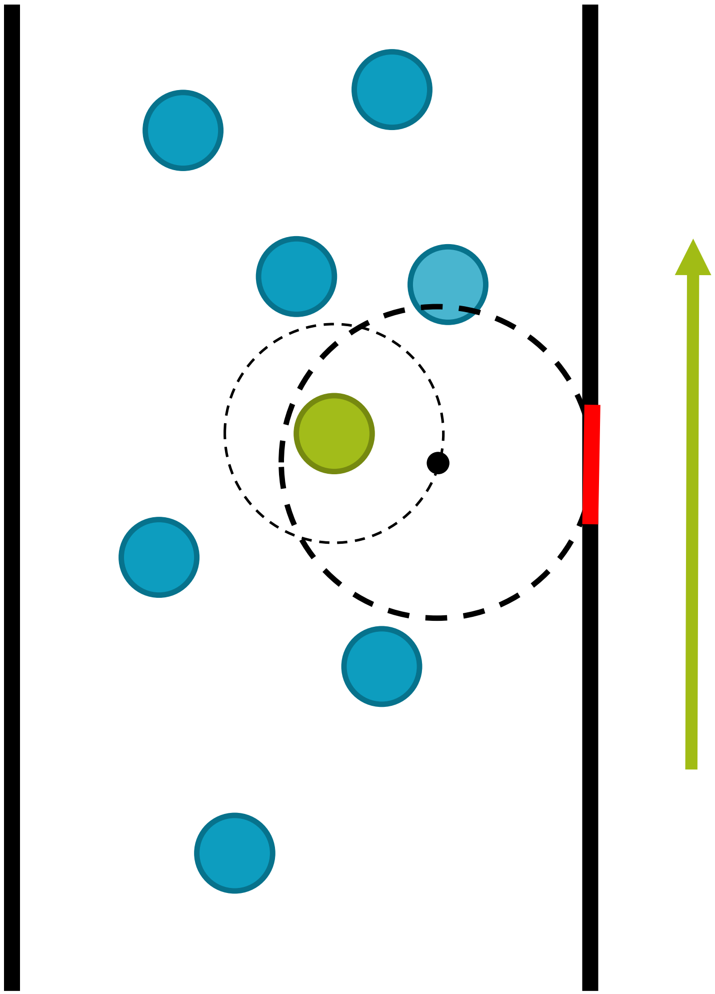
Model Development – Optimal Steps

- Neighbours near each position are checked – close neighbours make the position less desirable



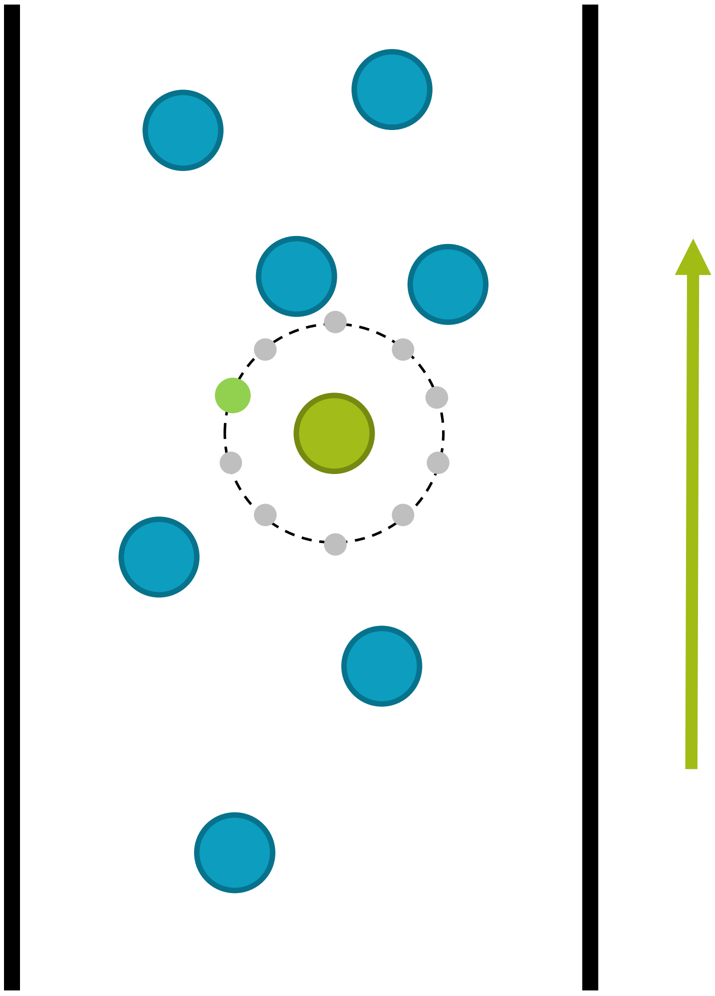
Model Development – Optimal Steps

- Neighbours near each position are checked – close neighbours make the position less desirable
- Obstacles near each position are similarly checked



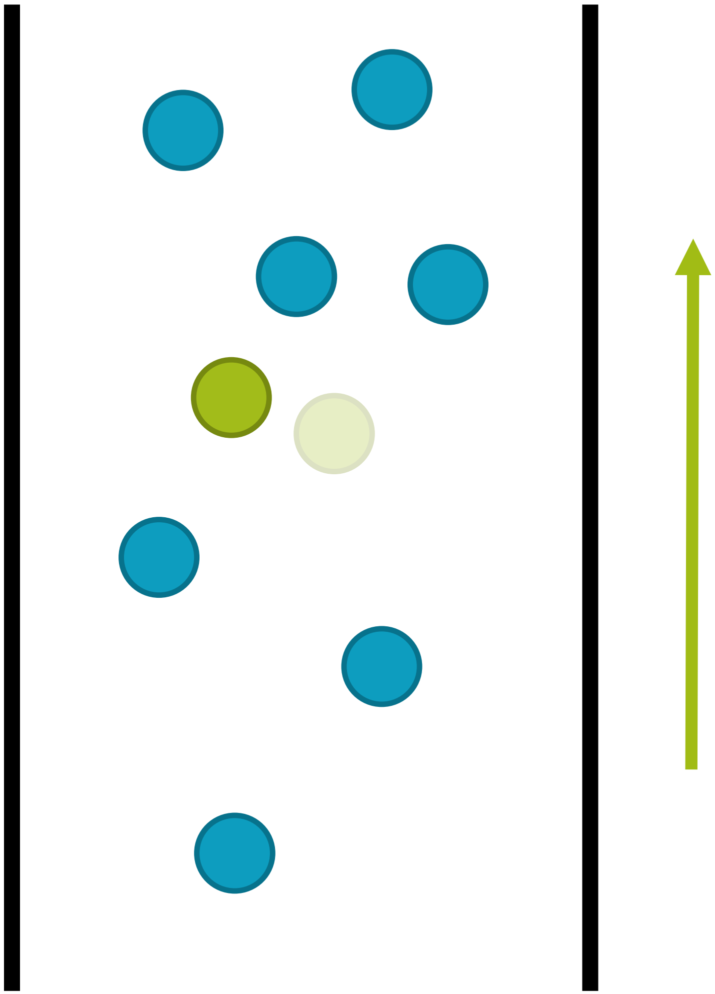
Model Development – Optimal Steps

- Neighbours near each position are checked – close neighbours make the position less desirable
- Obstacles near each position are similarly checked
- Based on all three factors, the most desirable position is selected



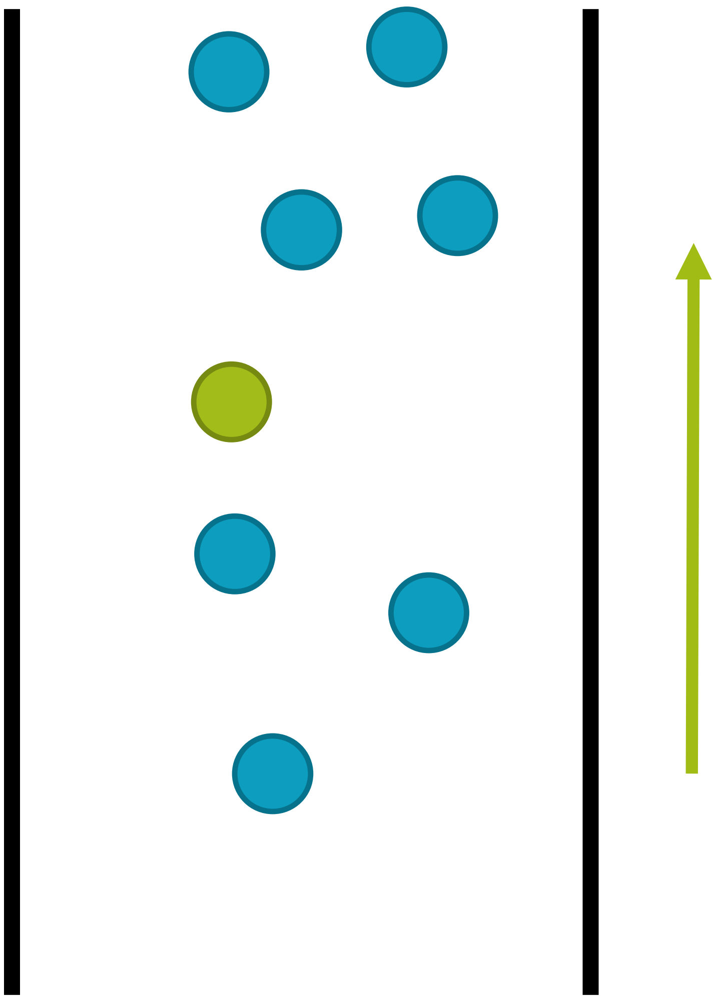
Model Development – Optimal Steps

- The agent is moved to the new position



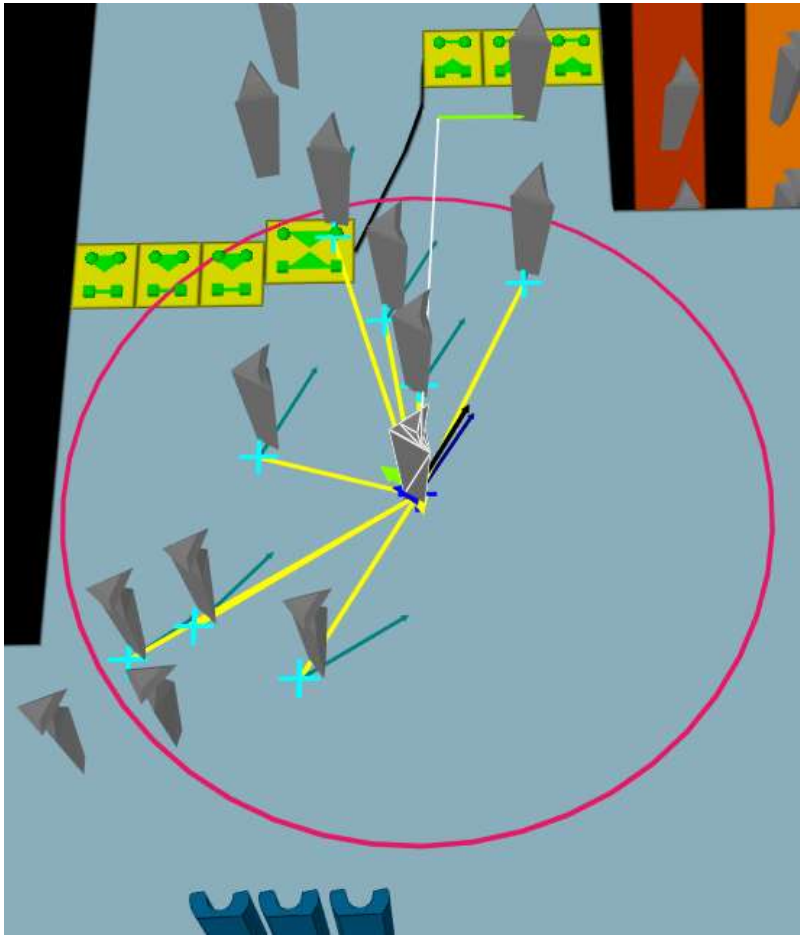
Model Development – Optimal Steps

- The agent is moved to the new position
- Every other agent is sequentially checked and moved based on the same criteria



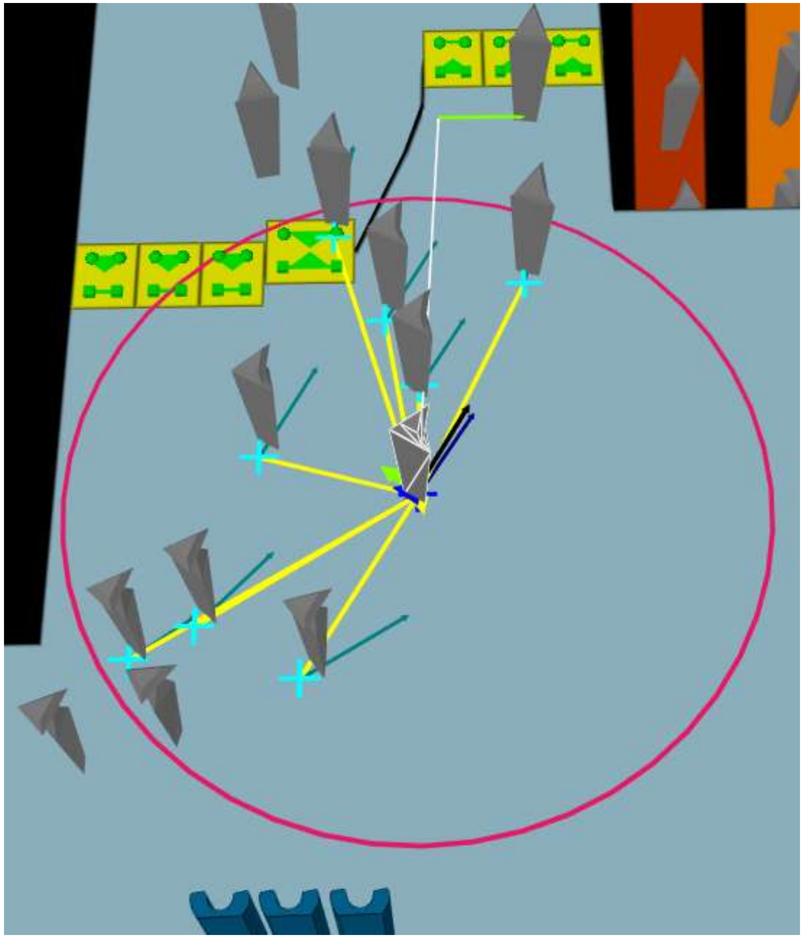
Model Development – MassMotion

- MassMotion is based on the Social Forces concept, but with many more forces



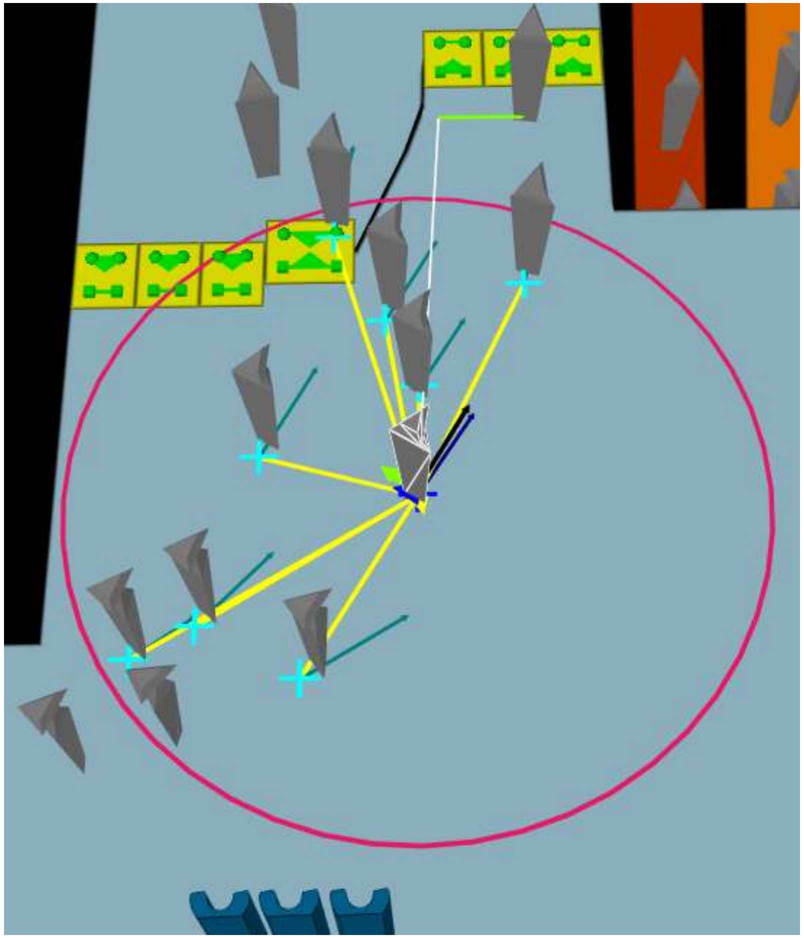
Model Development – MassMotion

- MassMotion is based on the Social Forces concept, but with many more forces
- These include cohesion, queueing, drift, corner, and collision forces, to name a few



Model Development – MassMotion

- MassMotion is based on the Social Forces concept, but with many more forces
- These include cohesion, queueing, drift, corner, and collision forces, to name a few
- No changes were made to this model



Model Calibration

- A Genetic Algorithm approach was chosen for calibration and is supported in the literature
- Attempts were made to calibrate microscopic model parameters (neighbour force, etc.), but this failed due to the lack of microscopic data
- In the end, a speed adjustment parameter was calibrated for all models, compensating for some idiosyncratic behaviours

Model Calibration

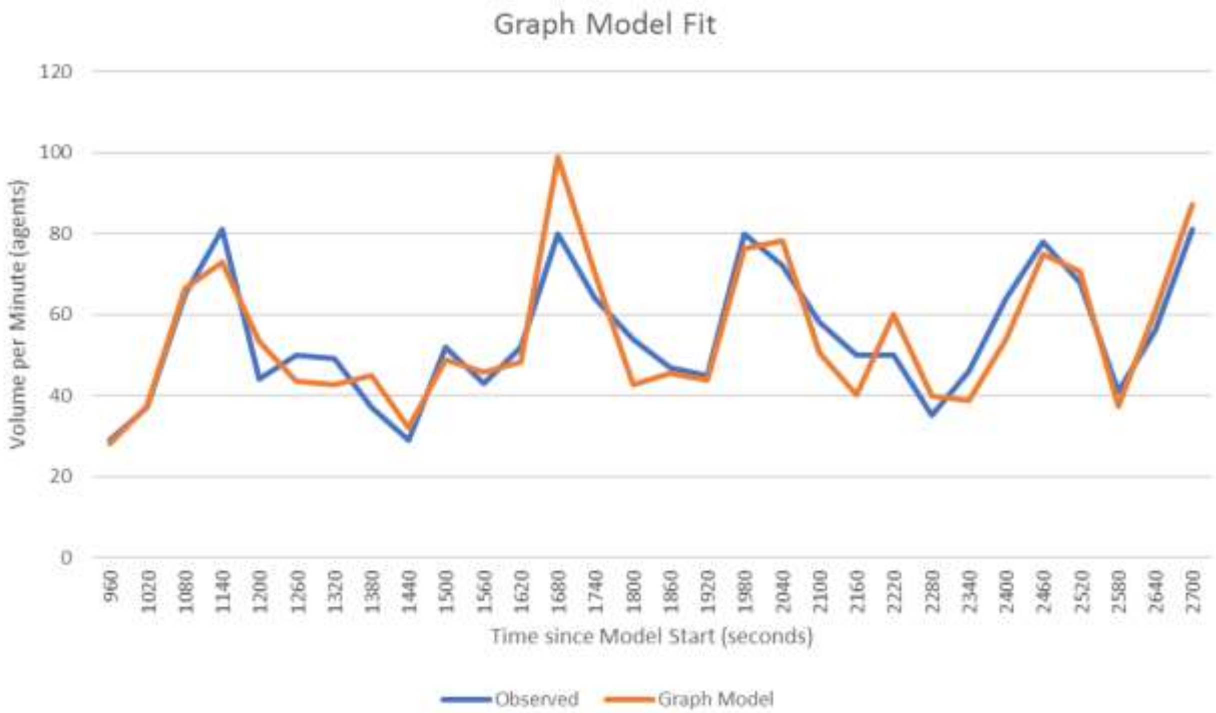
- Model fitness was based on matching 30 minutes of peak concourse exit flow rates
- Global Relative Error (GRE) was used as a fitness function

$$GRE = \frac{\sum_{i=1}^{n} |y_{obs} - y_{sim}|}{\sum_{i=1}^{n} y_{obs}}$$

- 20 different speed adjustment parameters were tested per generation, using 5 runs per factor and 10 generations

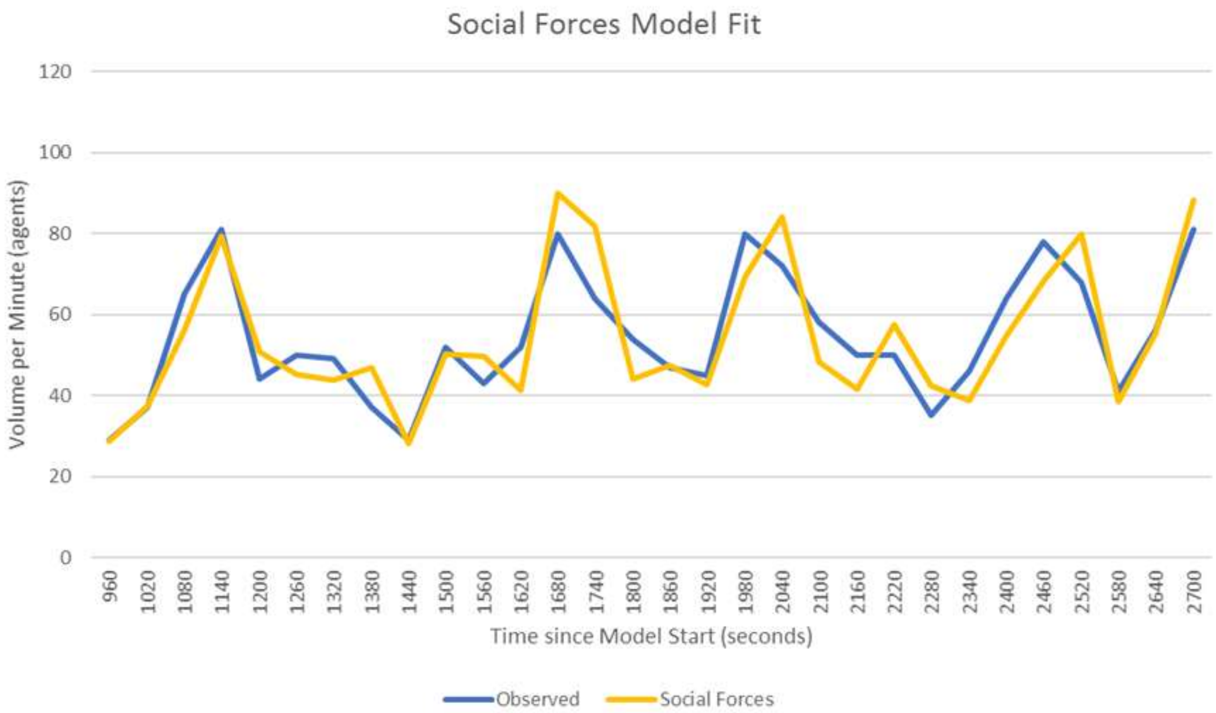
Model Calibration

- All models fit the data relatively well following calibration



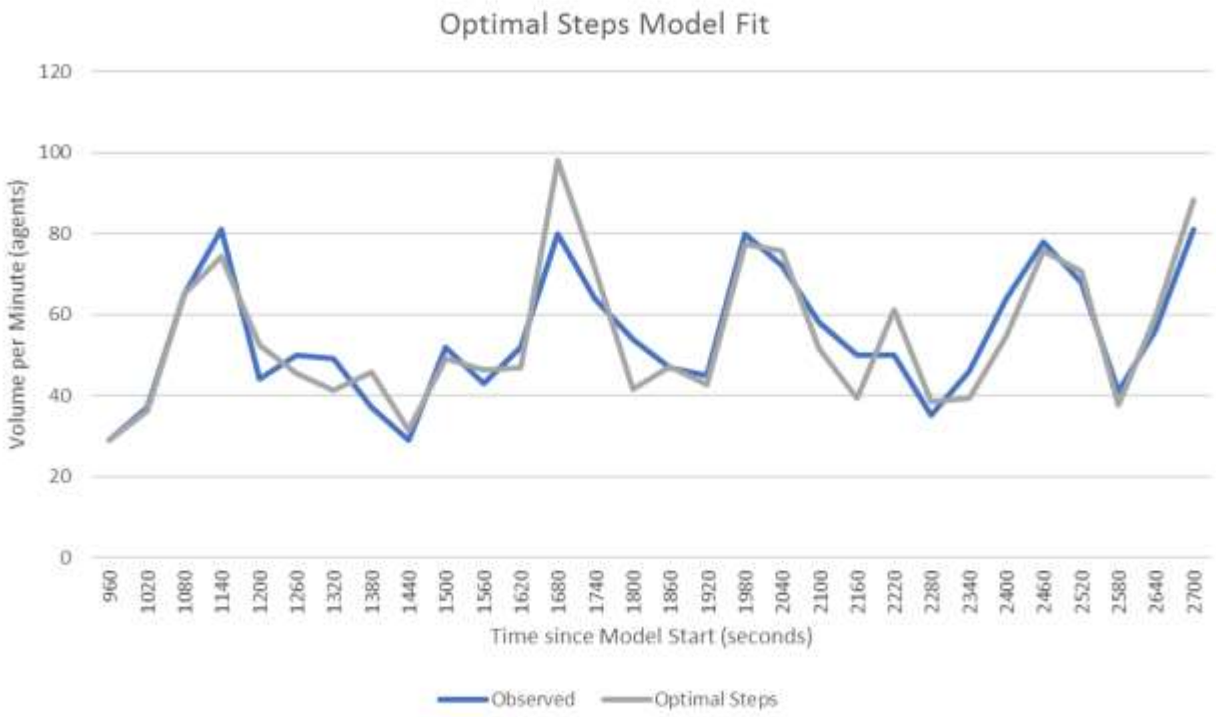
Model Calibration

- All models fit the data relatively well following calibration



Model Calibration

- All models fit the data relatively well following calibration



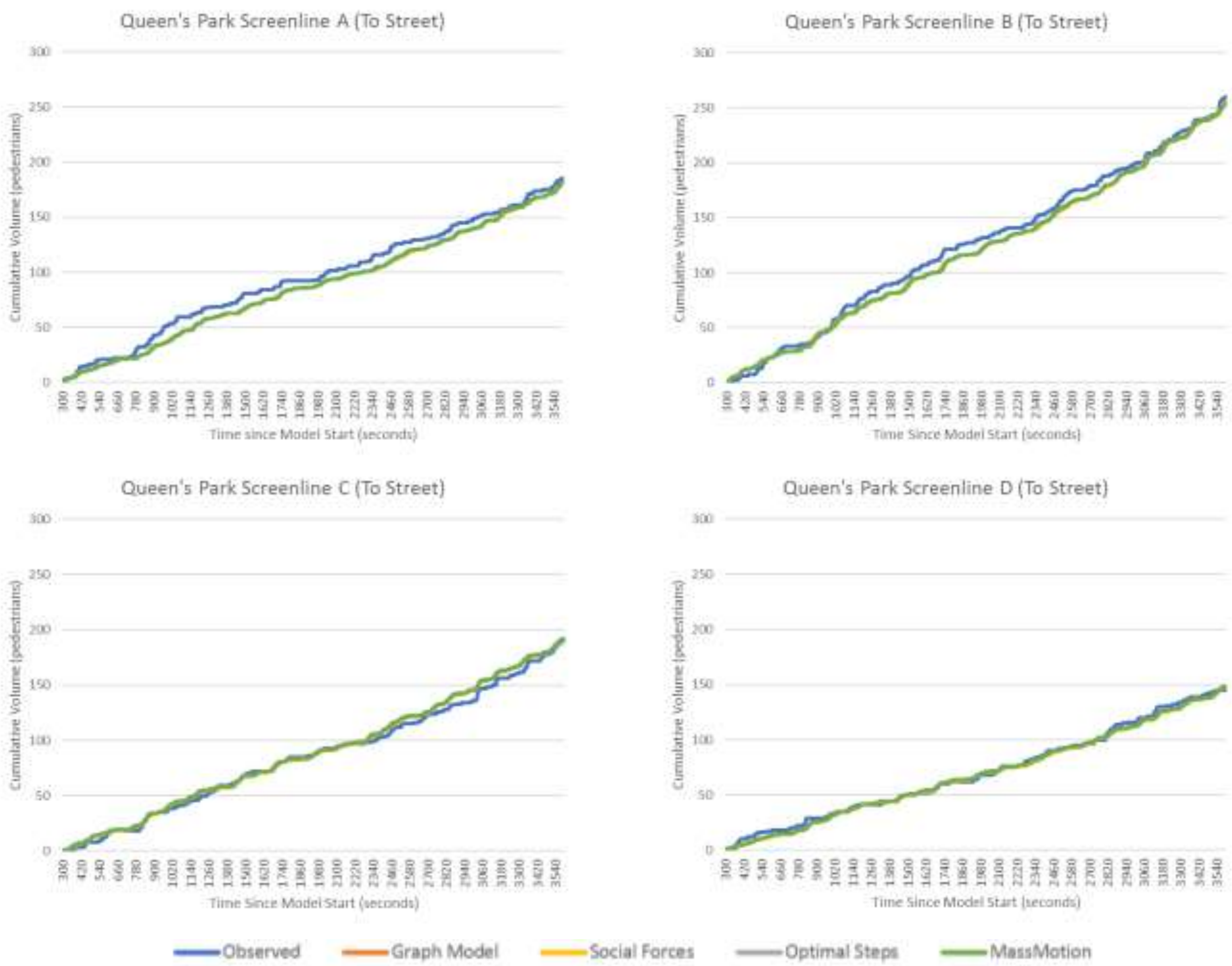
Model Calibration

- All models fit the data relatively well following calibration
- Some pointwise deviations are due to random assignment of agent destinations – not necessarily a bad model
- Overall, all models had similar error values following calibration and could be compared

Validation and Comparison

- Base case validation involved testing all models using the observed Queen's Park scenario
- Comparing observed screenline flows to simulated flows shows some differences, but overall good fits
- At these low volumes, all four models produce extremely similar results

Validation and Comparison



Validation and Comparison

- R-squared and Sum of Squared Errors (SSE) fits were also similar for all models
- Social Forces model had slightly lower total SSE

Model	Screenline						
	A	B	C	D	E	F	
R-Squared (%)							Average
Graph	97.22	98.97	99.23	99.52	99.89	99.77	99.10
Social Forces	97.21	99.00	99.36	99.41	99.89	99.84	99.12
Optimal Steps	97.39	98.99	99.32	99.48	99.89	99.81	99.15
MassMotion	97.24	99.02	99.27	99.51	99.90	99.81	99.12
SSE (1000s)							Total
Graph	228	166	71	27	653	4,706	5,851
Social Forces	228	161	59	33	660	3,242	4,382
Optimal Steps	215	165	63	29	665	4,001	5,137
MassMotion	226	159	67	27	633	4,044	5,157

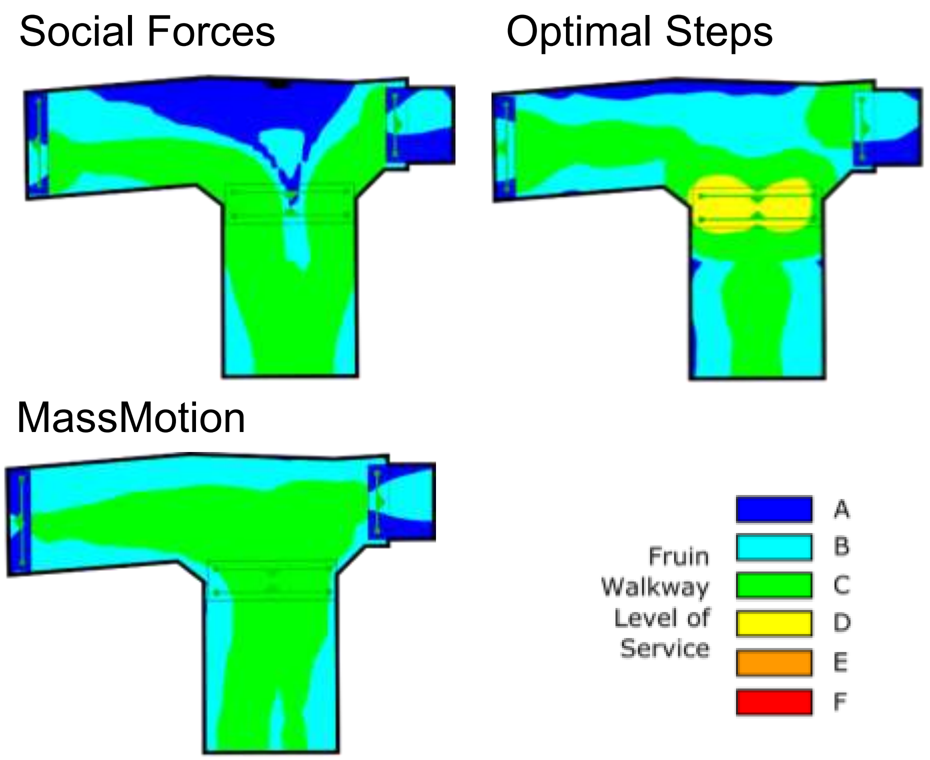
Validation and Comparison

- Project and simulation setup speeds were generally the same, but run speeds differed greatly between the models!
- Graph model was by far the fastest, followed by Social Forces

Model	Project Setup (s)		Simulation Setup (s)		Model Run (s)	
	μ	σ	μ	σ	μ	σ
Graph	0.08	0.02	8.59	0.16	24.87	0.50
Social Forces	0.09	0.01	8.67	0.11	75.13	4.25
Optimal Steps	0.08	0.01	8.59	0.10	132.87	1.45
MassMotion	0.09	0.02	8.60	0.18	101.43	0.93

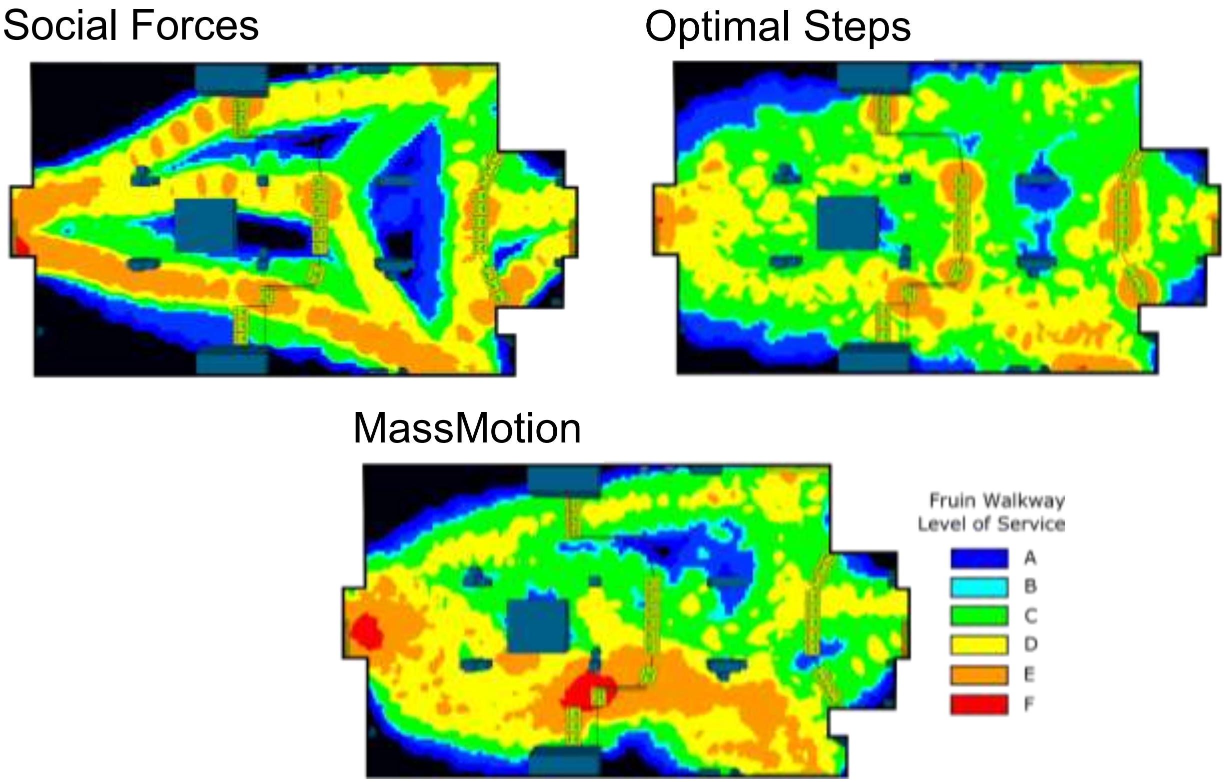
Validation and Comparison

- Comparing average element Level of Service, differences in agent paths became apparent
- Some spots also appeared where agent densities increased due to waiting behaviours



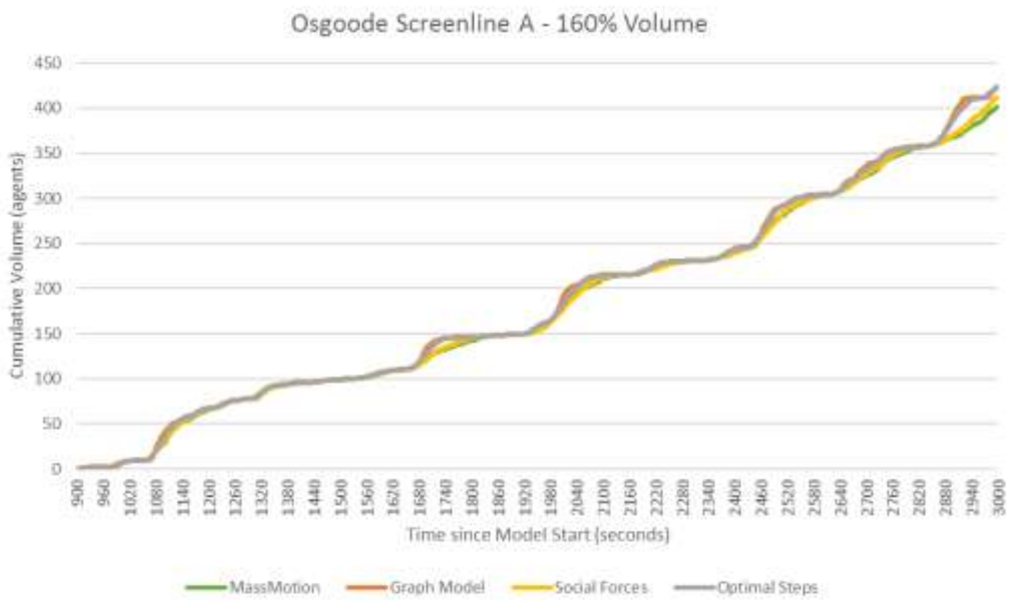
Validation and Comparison

- Comparing maximum concourse densities also highlighted route and crowding differences



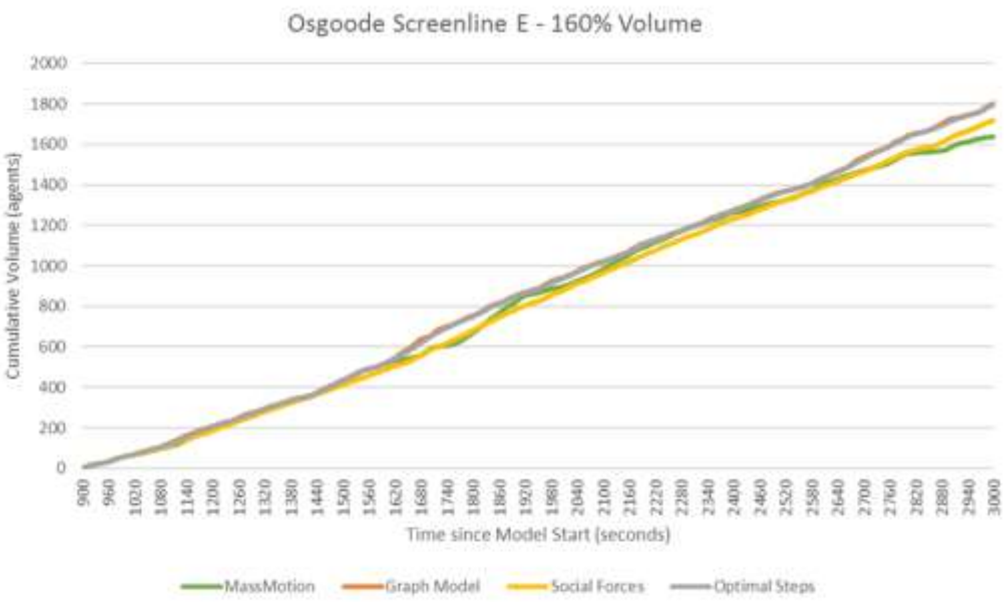
Validation and Comparison

- Increased volume scenarios compared model results to MassMotion, a trusted 'ground truth'
- These showed much greater differences!



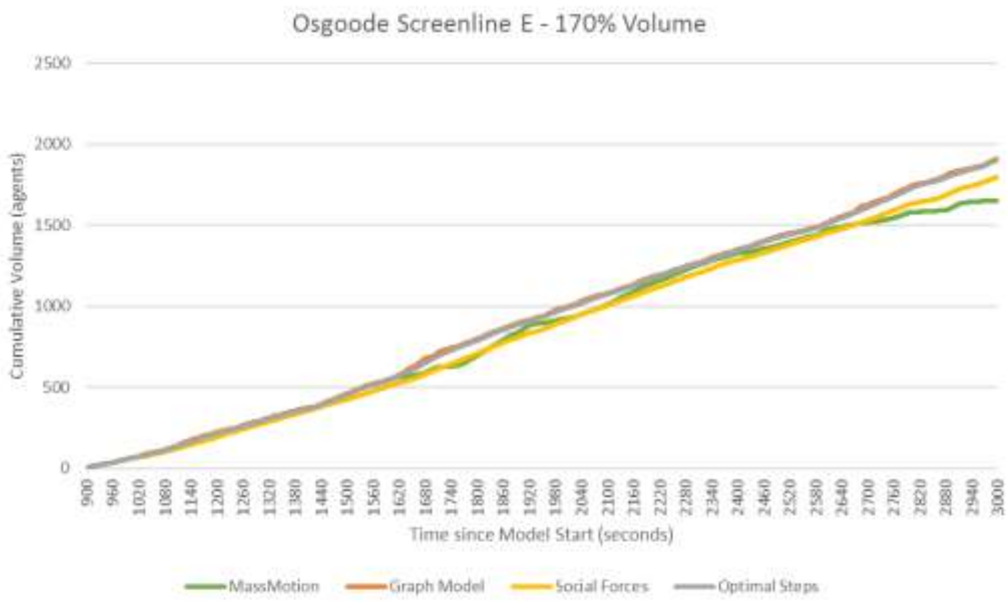
Validation and Comparison

- Increased volume scenarios compared model results to MassMotion, a trusted ‘ground truth’
- These showed much greater differences!



Validation and Comparison

- At very high volumes, Graph and Optimal Steps models failed to respond to congestion – flow rates were not appropriately reduced



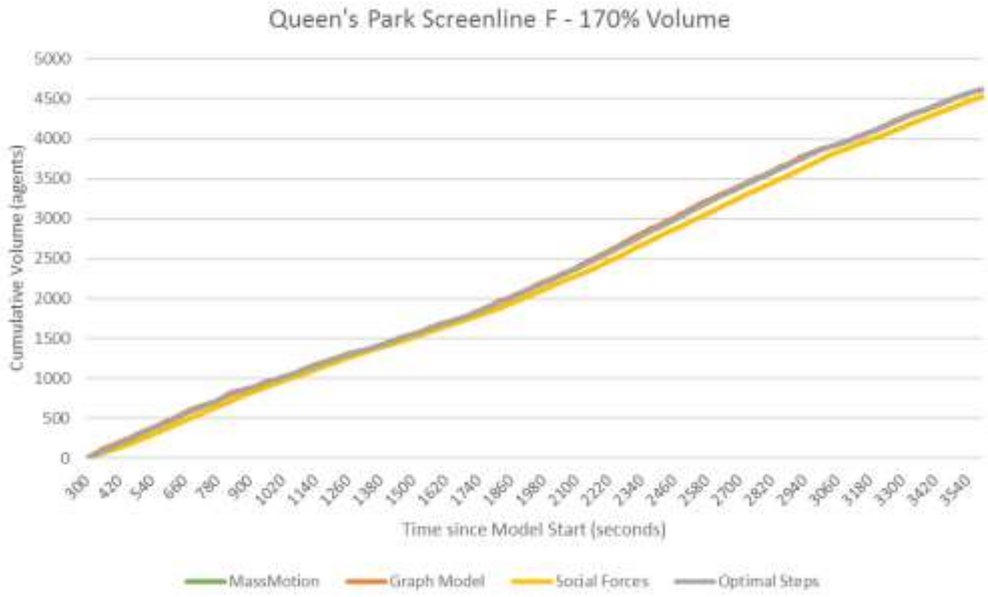
Validation and Comparison

- At very high volumes, Graph and Optimal Steps models failed to respond to congestion – flow rates were not appropriately reduced
- R-squared fits were also noticeably lower

Scenario	Model	Average R^2 (%)	Worst R^2 (%)	Model Run (s)
Osgoode 100% (base)	Graph	99.97	99.95	9.96
	Social Forces	99.98	99.96	20.63
	Optimal Steps	99.97	99.93	53.93
Osgoode 160%	Graph	99.47	99.00	15.70
	Social Forces	99.91	99.76	64.92
	Optimal Steps	99.52	99.10	91.82
Osgoode 170%	Graph	99.14	97.85	15.91
	Social Forces	99.87	99.49	76.42
	Optimal Steps	99.24	98.15	99.75

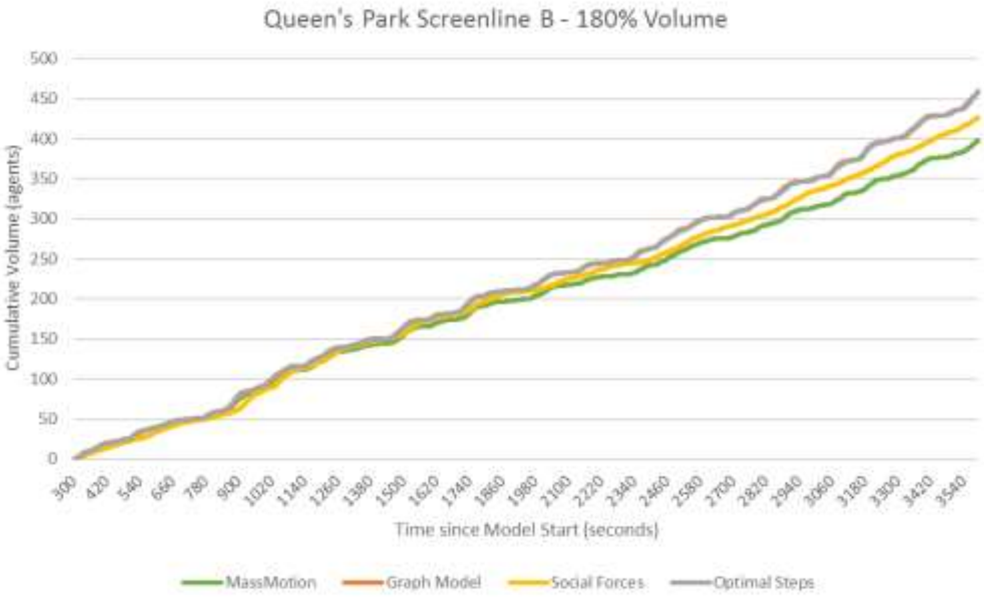
Validation and Comparison

- Similar trends for Queen's Park volume increase scenarios, but Social Forces model seemed overly sensitive to congestion



Validation and Comparison

- High volumes led MassMotion to predict system breakdown, which no other model showed
- Even low volume screenlines are visibly affected



Validation and Comparison

- High volumes led MassMotion to predict system breakdown, which no other model showed
- Even low volume screenlines are visibly affected

Scenario	Model	Average R^2 (%)	Worst R^2 (%)	Model Run (s)
Queen's Park 100%	Graph	99.99	99.99	24.87
	Social Forces	99.98	99.97	75.13
	Optimal Steps	99.99	99.99	132.87
Queen's Park 170%	Graph	99.98	99.97	44.48
	Social Forces	99.75	99.48	266.64
	Optimal Steps	99.98	99.95	286.29
Queen's Park 180%	Graph	94.77	90.97	47.39
	Social Forces	97.04	91.85	358.23
	Optimal Steps	94.96	91.45	319.47

Conclusions

- In all cases, Graph model is fastest to run, followed by Social Forces, MassMotion, and Optimal Steps
- Base case tests show good fits for all models, suggesting low volume stations don't require complex models

Conclusions

- Agent densities and paths show differences, especially with Social Forces model, but we have no ‘ground truth’ to compare against
- High volume tests indicate reasonable performance up to a point – once flow reducing congestion occurs, MassMotion is the only choice

Future Work

- Collect microscopic pedestrian movement data, perhaps using Bluetooth/Wi-Fi tracking and video recording, for better calibration
- Collect higher-volume data at busier stations or during disruption events
- Modify and improve the MassMotion SDK, allowing other models to integrate with the software and improving performance

Acknowledgements

UNIVERSITY OF TORONTO
FACULTY OF APPLIED SCIENCE & ENGINEERING
Transportation Research Institute

ARUP

MassMotion

NSERC
CRSNG

Foundation
Fondation

Thank You

Questions?