Strategies for simulating pedestrian navigation with multiple reinforcement learning agents

Martinez-Gil, F., M. Lozano, and F. Fernandez (2015). Strategies for Simulating Pedestrian Navigation with Multiple Reinforcement Learning Agents, Journal of Autonomous Agents and Multiagent Systems, 29(1):98-130.

Presented by Gisela Font Sayeras

Outline

- Introduction
- Model pedestrian navigation as MARL
- State space generalization
- Algorithms
- Experiments
- Conclusion

Introduction

- Why do we want to study a pedestrian simulation?
  - Capacities of facilities in a building
  - Prevent accidents
  - Realism to simulated urban scenarios
- Pedestrian dynamics has two main perspectives.
  - Macroscopic: global functions
  - Microscopic: individual features
    - Collective effects observed in the motion of pedestrian crowds
    - Higher-level decision making without major modifications of the basic behavioral model

Objective

- Simulating groups of pedestrians, based on multi-agent reinforcement learning (MARL) techniques.
- Demonstrate that MARL is a suitable framework for generating plausible simulations (not to reproduce real crowd behaviors) using learned microscopic interactions.
  - Scenario 1: closed room with a single door (exit) that the agents have to reach
  - Scenario 2: crossing of two groups of agents inside a narrow corridor

Benefits of the MARL Approach

1. Low computational cost
2. The richness of the group behavior in terms of variability
3. Model-free design of the problem
4. Emergent collective behaviors

Outline

- Introduction
- Model pedestrian navigation as MARL
- State space generalization
- Algorithms
- Experiments
- Conclusion

Model

- Markov Decision Processes (MDP):
  - 4-tuple
    - State space S
    - Action space A
    - Probabilistic function P

      $$P : S \times A \times S \rightarrow [0, 1]$$

    - Reward function R

      $$R : S \times A \rightarrow \mathbb{R}$$

  - Find policy: maximum discounted expected reward

    $$V(s) = E\{\sum_{t=0}^{\infty} \gamma^t r_t\}$$

    where γ is the discount parameter that sets the influence of future rewards

Model

- Q-learning:
  - uses a table (called Q) to represent the value function
  - the expected accumulated reward of being in state s and doing action a is stored

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_a\{Q(s_{t+1}, a)\} - Q(s_t, a_t)],$$

Scenarios

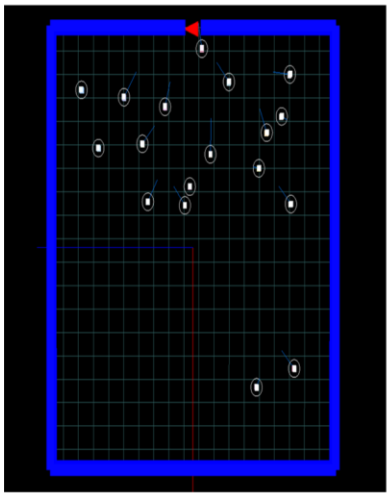
1. Group of pedestrians inside a closed room need to learn how to reach the door and leave the room

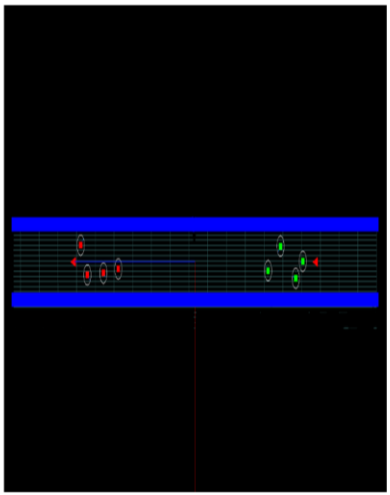
2. Narrow corridor in which two groups of four agents each have to cross to reach the opposite end.

Scenarios: constraints

- Agents
  - Max velocity of 1.8m/s
  - Center of a bounding circumference with radius 0.3m
- Environment
  - Two dimensional continuous plane
  - Scenario 1
    - 225 square meters room
    - Aperture of 0.8 meters (door) in the center of one of the sides
  - Scenario 2
    - Corridor of 15 meters long by 2 meters wide

Scenarios





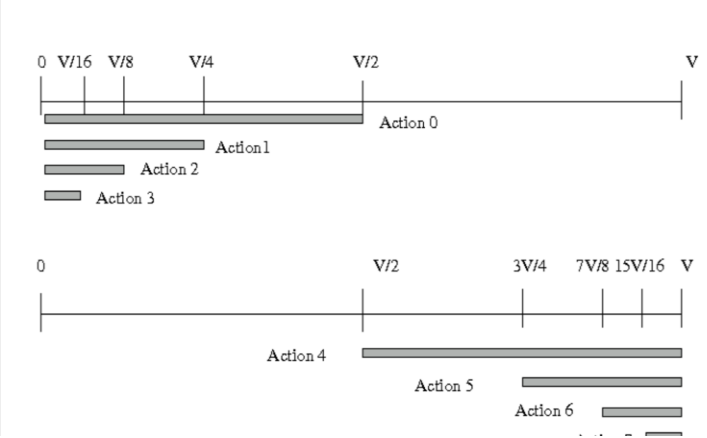
Description of the features of the agent's state

- Deictic representation: register information about objects that are relevant to the task at hand

Sag	Module of the velocity of the agent.
Av	Angle of the velocity vector relative to the reference line.
$Dgoal$	Distance to the goal.
$Srel_i$	Relative scalar velocity of the ith nearest neighbor.
Dag_i	Distance to the ith nearest neighbor.
Aag_i	Angle of the position of the ith nearest neighbor relative to the reference line.
Lag_i [1]	Label to identify the group that the neighbor belongs to.
Dob_j	Distance to the jth nearest static object (walls).
Aob_j	Angle of the position of the jth nearest static object relative to the reference line.

Agent’s decision

- Actions are taken in pairs
  - Speed
  - Velocity vector
- 8 different ratios plus the ‘no operation’ option for both the speed and the angle
  - 81 possible combined actions



Modeling the Agents' Behavior

First Scenario (Agents in a room)	
Crash against other agent	-0.1
Crash against a wall	-2.0
Reach the goal	+100.0
Default	0.0
Second Scenario (Crossing)	
Reach the goal	+100.0
Default	0.0

Outline

- Introduction
- Model pedestrian navigation as MARL
- State space generalization
- Algorithms
- Experiments
- Conclusion

State Space Generalization

- The states are generalized using Vector Quantization (VQ)
  - Map from a state space in a k-dimensional Euclidean space, R^k to a finite set C containing N states.

$$V_Q(x) = \arg\min_{y \in C}\{dist(x, y)\}$$

where C is the prototypes states and x is in R^k and is the sensorized state

  - Generalized Lloyd Algorithm (GLA)
    Linde, Y., Buzo, A., Gray, R.: An algorithm for vector quantizer design. IEEE Trans. on Commun. 28(1), 84–95 (1980)

- Vector Quantization for Q-Learning (VQQL) is a learning schema that uses VQ as the generalization method for the state space and the tabular version of Q-Learning for the learning process

State Space Generalization

Single-agent VQQL Algorithm

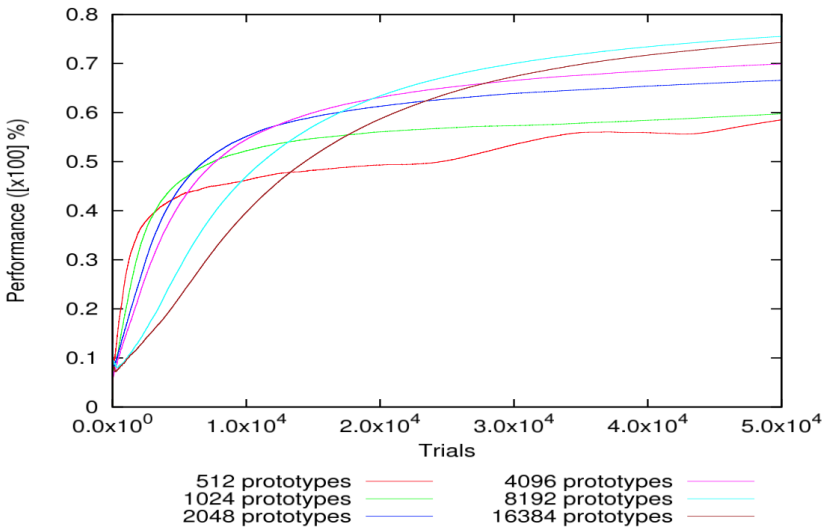
1. Generate the set T of samples of the state space S interacting with the environment using an exploratory policy.
2. Discretize the state space:
   (a) Use GLA to obtain a state space discretization $C \in S$ from the sample set T.
   (b) Let $VQ : S \rightarrow C$ be the function that, given any state in S, returns the discretized value in C.
3. Learn the Q-Table
   While the final condition is not reached
   i. Get an experience tuple $< s_1, a, s_2, r >$ by interacting with the environment.
   ii. Map the states of the experience tuple using VQ. Each adquired tuple of experience $< s_1, a, s_2, r >$ is mapped to $< VQ(s_1), a, VQ(s_2), r >$
   iii. Apply the Q-Learning update function defined in Equation 1 to learn a tabular value function Q: $C \times A \rightarrow \Re$, using the mapped experience tuple.
4. Return Q and VQ

Problems

- Policy guided-bias
- Number of prototypes to use
  - Run 6 experiments in each scenario specifying the number of prototypes
    - Agents placed randomly inside the room
    - Same Maximum number of steps for all episodes and agents
    - At each step, the agent has to make a decision to adjust its velocity

Experiments Results

- Best trade-off between performance and computational cost:
- Scenario 1: 4096 prototypes
- Scenario 2: 8192 prototypes



Outline

- Introduction
- Model pedestrian navigation as MARL
- State space generalization
- Algorithms
- Experiments
- Conclusion

Algorithms

- The application of RL to pedestrian navigation faces 3 challenges:
  - Multi-Agent environment
    - Decide how many agents need to learn from scratch
  - The agents can perceive a different number of agents in different situations.
    - S1. The different descriptions of the state space are inherent in the incremental setting of the agents
    - S2. The evolution of the episode in time creates a variable perception of neighbors
  - Random exploration of the environment does not produce a representative dataset with which to generate a correct set of prototypes for the VQ space generalization method

Iterative Schemas

- Iterative Vector Quantization for Q-Learning (ITVQQL)
  - Same number of agents for every learning iteration
  - It models the state space with one set of prototypes with a fixed number of features to represent the neighboring agents
- Incremental Vector Quantization for Q-Learning (IN-VQQL)
  - Stage beginning with one agent in the first iteration and incrementing by one more agent in each following iteration to reach the total number of agents in the last iteration
  - Uses different vector spaces to represent the different perceptions in terms of the neighborhood

Description of the two schemas

Multi-agent ITVQQL/INVQQL schemas

Entry: The number of iterations N

Return: The sets Q_N and V_N (The value table of Q-learning and the vector quantizer respectively).

1. $i \leftarrow 1$
2. **Set p to the initial number of agents in the environment**
3. For each agent, k $(1 \leq k \leq p)$ set:
   - its initial vector quantizer, $V_0^k(s) = \emptyset$
   - its initial policy $\pi_0^k = random$
4. Repeat:
   (a) **Decide whether or not to include new agents.** Set p consequently.
   (b) For each agent, k $(1 \leq k \leq p)$ do:
      i. Collect a dataset T_i^k for agent k using the policies π_{i-1}^k with V_{i-1}^k
      ii. **Build V_i^k using T_i^k for agent k following a transfer learning strategy**
   (c) Learn Q_i^k $\forall k, 1 \leq k \leq p$ and hence the policies π_i^k using Q-Learning (with the option of using transfer of value functions).
   (d) $i \leftarrow i + 1$
   Until $i = N$
5. Return Q_N and V_N

Summary of the settings of the schemas

Feature	ITVQQL	INVQQL
Number of prototypes	Fixed	Variable
Number of features per prototype	Fixed	Variable
Number of agents per iteration	Fixed	Variable
Inter-iteration policy transfer	Yes	Yes
Inter-iteration prototype transfer	No	Yes
Inter-iteration value function transfer	Yes	Yes

Value function transfer procedures

- Transfer of the value function learned in a previous iteration as the initial values for the value function of the next one.
- Euclidean metric

Simple Complexification with a Q-table

---

1. Train with a source representation and save the learned Q-table and the vector quantizier Q_{source}, V_{source}
2. **for** each prototype $q^i_{target} \in C_{target}$
   - Find prototype $q^j_{source} \in C_{source} \mid \min_j \| q^i_{target} - q^j_{source} \|$
   - $Q_{target}(q^i_{target}, action_k) = Q_{source}(q^j_{source}, action_k)\ \forall action_k$

Transfer Learning Evaluation

1. Jumpstart: The improvement in the initial performance of an agent.

2. Asymptotic Performance: The improvement in the final performance

3. Time to Threshold: The learning steps needed by the agent to achieve a pre-specified performance level

Outline

- Introduction
- Model pedestrian navigation as MARL
- State space generalization
- Algorithms
- Experiments
- Conclusion

Experiments

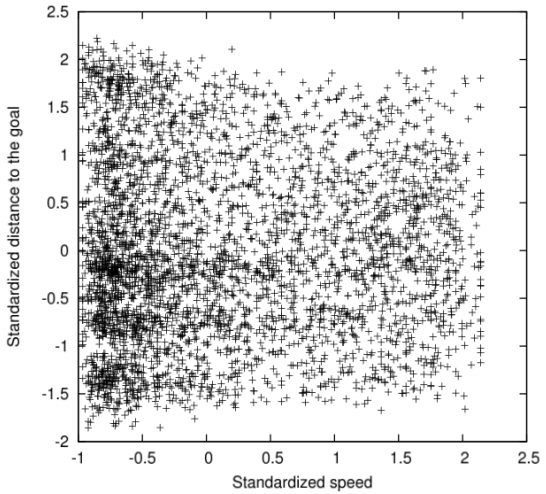
- Results for the first scenario
  - Agents in a closed room scenario
- Results for the second scenario
  - Crossing experiment

Results for the first scenario

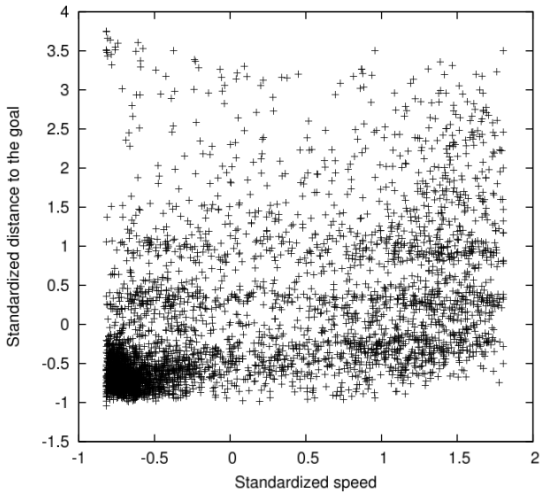
- Number of iterations performed is 18 (N = 18)
- Each iteration has been set empirically to 50,000 episodes
  - Episode ends when
    - All the agents reach the goal
    - Maximum of 150 decisions have been taken
- Same parameter configuration for all agents

$\gamma = 0.9 \qquad \alpha = 0.4 \qquad \epsilon = 0.4$

Visualization of the Prototypes



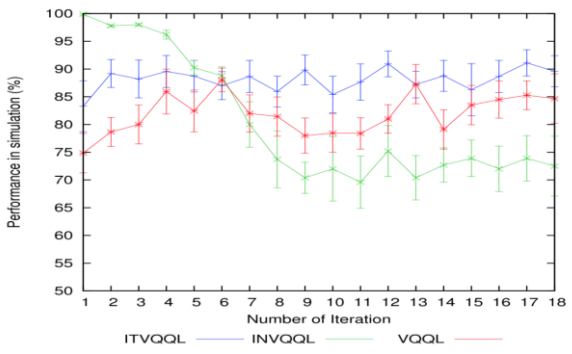
First iteration of the ITVQQL schema in the room scenario

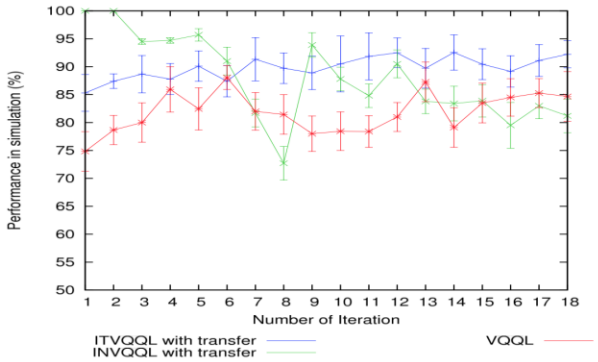


Last iteration of the ITVQQL schema in the room scenario

Learning Process Performance for all the Schemas

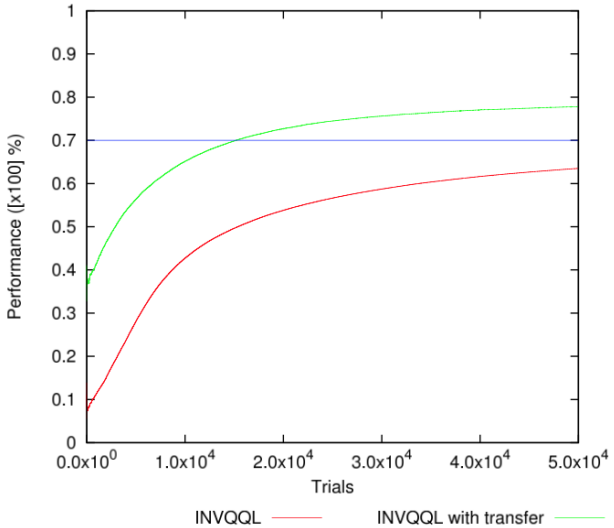
Key	ITVQQL	INVQQL	VQQL
Episodes	50000	50000	900000
Iterations	18	18	1
Prototypes	4096	From 4096 to 32768	4096
Features per prototype	28	From 7 to 28	28
Agents per iteration	18	From 1 to 18	18
Inter-iteration prototype transfer	0	4096	0

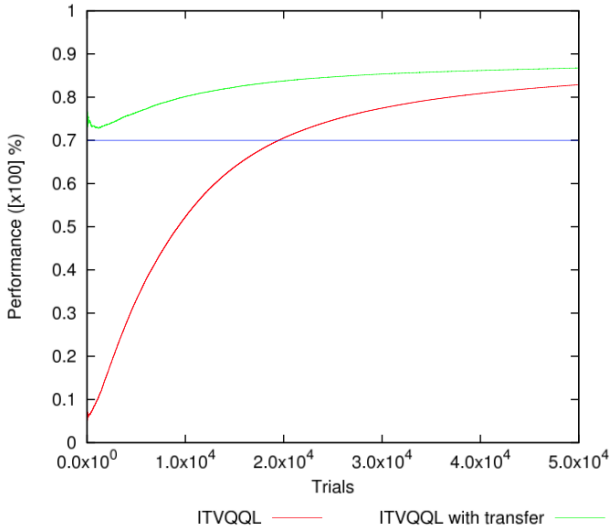




The learning process performance for all the schemas for the closed room scenario. The performance is calculated using a greedy policy over the learned value functions.

Learning Process Performance





These curves correspond to the last iteration.
The curves are averages over the 18 agents.

Analysis

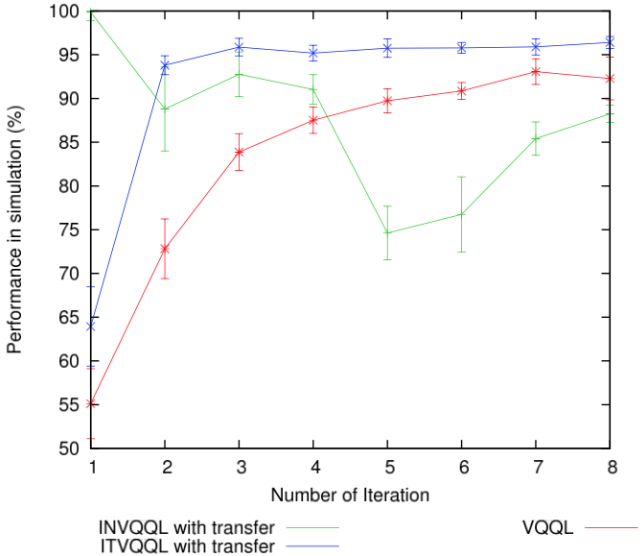
- Learning process converge for the two proposed learning schemas, in the considered domain
- The transfer of knowledge benefits the INVQQL schema in the learning task.

Results for the Second Scenario

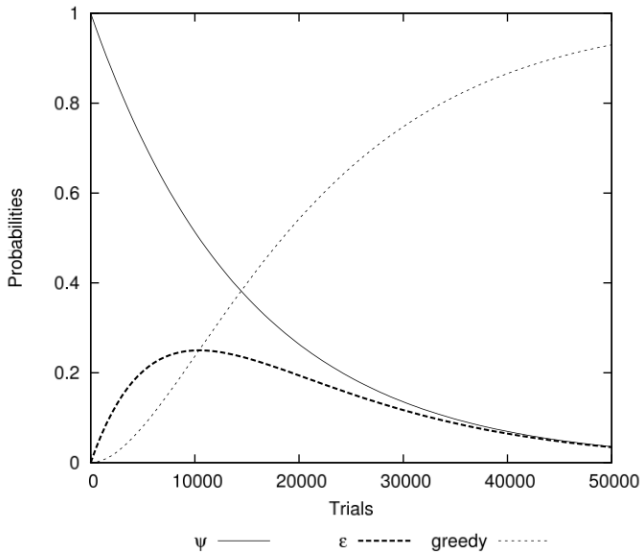
- Eight agents (N = 8) are divided in two groups
- Each iteration has been set empirically to 50,000 episodes
  - Episode ends when
    - All the agents reach the goal
    - Maximum of 150 decisions have been taken
- Same parameter configuration for all agents

γ	0.9
α	from 0.3 to 0.1
ϵ	1.0 (initial value)
ψ	1.0 (initial value)

Learning Process Performance



The learning process performance for all the schemas for the crossing scenario. The performance is calculated using a greedy policy over the learned value functions.



The probability functions used in Policy Reuse transfer technique

$$\begin{cases} \psi & \text{choose the } \pi_0 \text{ policy} \\ (1-\psi)\epsilon & \text{choose an aleatory action} \\ (1-\psi)(1-\epsilon) & \text{choose the greedy policy} \end{cases}$$

Pedestrian simulation

- Quality Evaluation:
  - Local interactions (microscopic level)
  - Macro-dynamics (macroscopic level)
  - Performance: path length, number of decisions per episode and number of fails

Fundamental Diagram

- Calculation of the weighted averaged density and velocity
  - Local density is obtained by averaging over a circular region of radius R
  - The local density at the place r(x,y) at time t

$$\rho(\mathbf{r},t) = \sum_j f(\mathbf{r_j}(t) - \mathbf{r}),$$

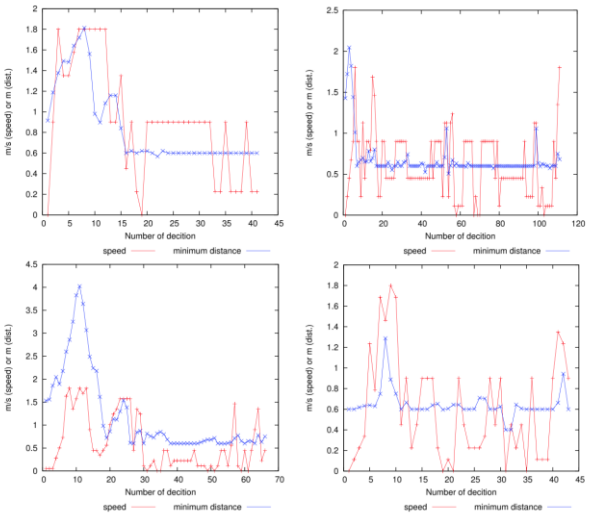
  - Gaussian, distance-dependent, weight function

$$f(\mathbf{r_j}(t) - \mathbf{r}) = \frac{1}{\pi R^2} \exp[\frac{-||\mathbf{r_j} - \mathbf{r}||^2}{R^2}]$$

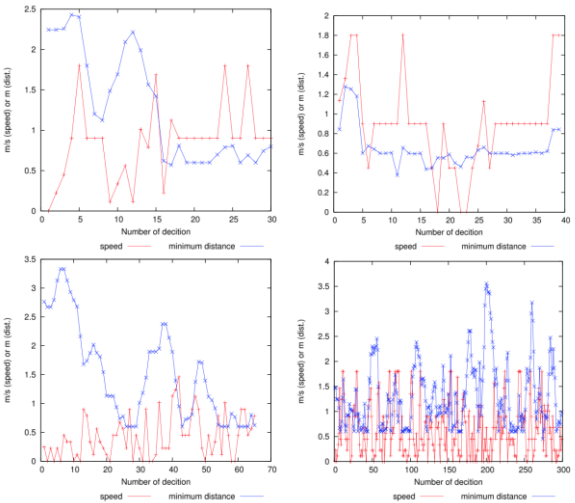
  - Local speeds are defined via the weighted average

$$\mathbf{V}(\mathbf{r},t) = \frac{\sum_j \mathbf{v_j} f(\mathbf{r_j}(t) - \mathbf{r})}{\sum_j f(\mathbf{r_j}(t) - \mathbf{r})}.$$

Local Interactions Analysis for the First Scenario



Local interaction results for an agent in an episode. All the schemas have used transfer in the learning phase. ITVQQL in the first row and INVQQL in the second row.



Baseline experiments for the local interaction results. VQQL in the first row and RANDOM in the second row.

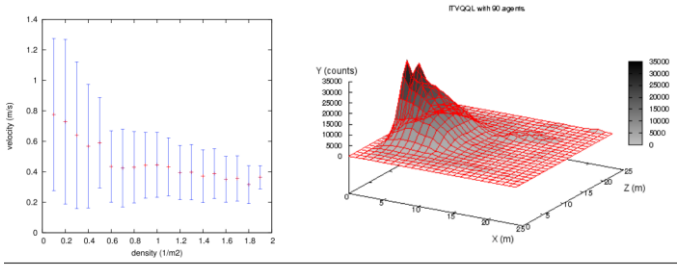
The first column is without scaling (18 agents) and the second column is with scaling (90 agents)

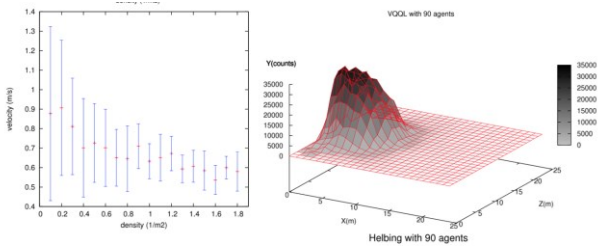
Macro-dynamics

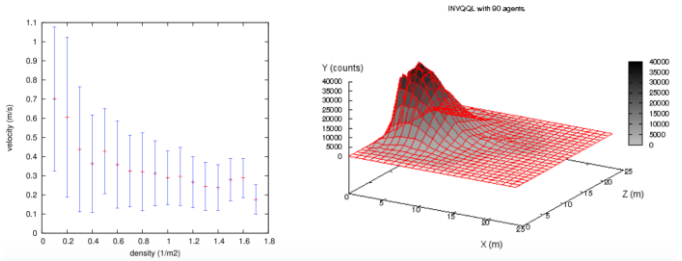
- Comparison with Helbing model
  Helbing, D., Farkas, I., Vicsek, T.: Simulating dynamical features of escape panic. Nature 407, 487 (2000)

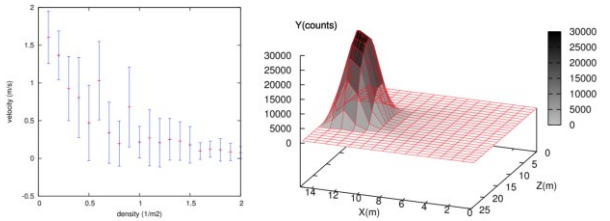
- Fundamental diagram summarizes the micro-behavior, showing the relation between the speed and the density of all the agents involved.

Macro-dynamics









Helbing with 90 agents

Fundamental diagrams (mean velocity and std deviation vs. density) and density maps with 90 agents

Performance

Averaged lengths and standard deviation for the paths in meters

#Ag.	IT	IN	TF_IT	TF_IN	VQQL
18	14 ± 7	20 ± 21	15 ± 7	18 ± 10	15 ± 11
36	13 ± 5	18 ± 15	15 ± 7	18 ± 8	15 ± 17
54	14 ± 6	18 ± 13	16 ± 8	19 ± 10	15 ± 11
72	15 ± 6	18 ± 12	16 ± 8	19 ± 11	15 ± 10
90	15 ± 7	18 ± 11	17 ± 9	20 ± 11	17 ± 10

Averaged number and standard deviation of decision per episode

#Ag.	IT	IN	TF_IT	TF_IN	VQQL
18	28 ± 56	41 ± 36	28 ± 56	63 ± 35	27 ± 21
36	32 ± 41	75 ± 52	67 ± 53	93 ± 58	43 ± 60
54	51 ± 55	118 ± 77	95 ± 74	135 ± 86	52 ± 60
72	68 ± 70	121 ± 78	120 ± 84	175 ± 112	69 ± 62
90	84 ± 85	144 ± 95	145 ± 96	211 ± 131	105 ± 135

Performance

Medians and means (in parenthesis) for the agents that do not reach the door (fails) when scaling up the number of agents. Median values separated by letters for the same number of agents (within a row), are significantly different (P<= 0.05) according to Kruskal-Wallis test.

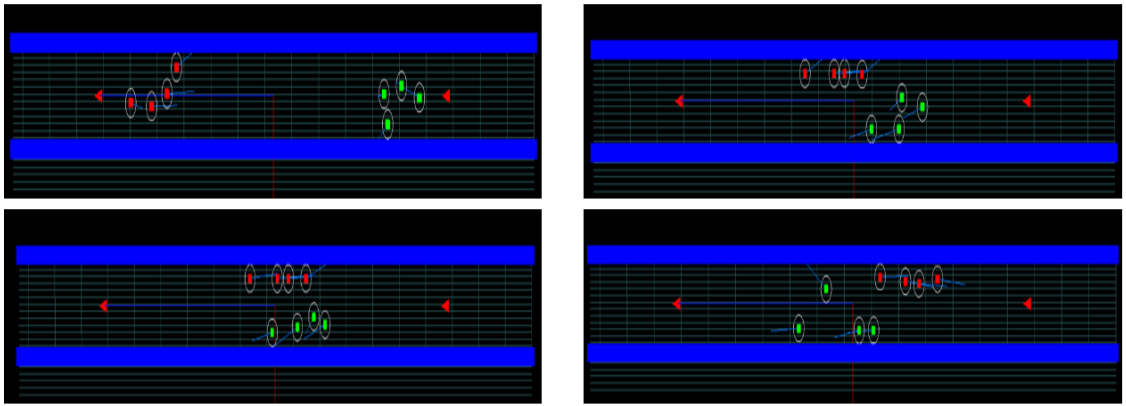
#Ag.	IT	IN	TF_IT	TF_IN	VQQL	P-value
18	1 b (2.4)	4 c (4.2)	0 a (1.0)	0 a (2.9)	1 b (2.8)	0
36	1 ab (6.5)	4 c (4.8)	0 a (2.8)	3.5 bc (6.2)	0 ab (6.8)	4×10^{-9}
54	1 a (6.4)	4 b (6.0)	0 a (3.6)	4 b (6.4)	10 ab (15.7)	7×10^{-10}
72	1 ab (9.4)	4 b (5.6)	1 a (3.9)	4 b (6.6)	4 b (22.1)	4×10^{-8}
90	1 ab (10.3)	4 b (6.0)	1 a (4.1)	3 b (6.5)	18.5 c (25.0)	4×10^{-10}

Performance Analysis

1. The analysis of the graphs of speed and distance to the nearest neighbors, shows that all the schemas provide behaviors with a correlation between the curves.
2. The fundamental diagrams and density maps reveal that the main characteristics of the pedestrian dynamics and collective behavior appear in all the schemas.
3. The scalability tests displayed in the tables show properties of real pedestrian behavior and demonstrate empirically that the learned behaviors are generalizable to other configurations with more agents.
4. VQQL baseline algorithm has worse performance (highest mean) than the rest of the schemas when scaling up the number of agents. IF_IT has the best performance (lowest mean)

Macroscopic analysis for the second scenario

- Study the emergence of collective behaviors
  - Formation of lines



Four steps of a simulation from the crossing scenario with the TF_IT schema

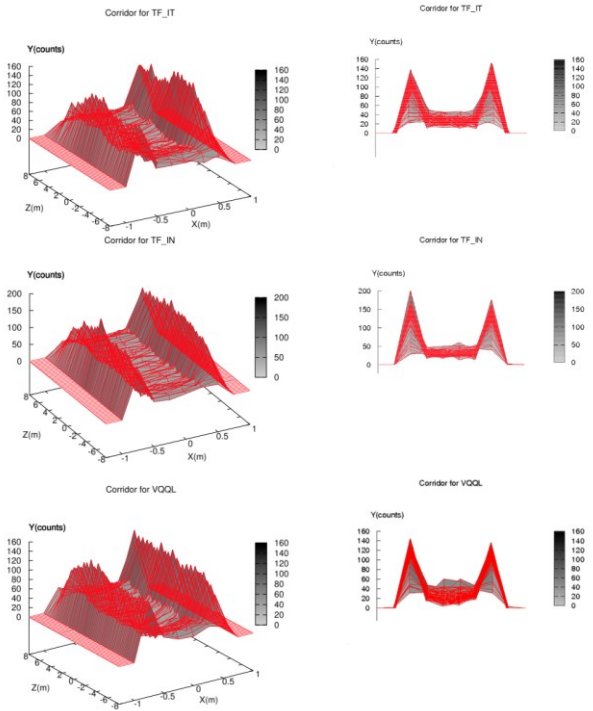
Macroscopic analysis for the second scenario

Mean of the number of episodes that end successfully from a series of 100 episodes

TF_IT	TF_IN	VQQL	P-value
81 a	52 b	63 c	0.0000

Macroscopic Analysis for the Second Scenario

Density maps of the schemas for the crossing scenario. The right column shows a side view of the same map scenario to compare the density of the middle region.



Second Scenario Analysis

1. The emergence of the lanes occurs in all the learning approaches evaluated. In all the cases, the lanes have similar structure.
2. The TF_IT has better performance than the TF_IN schema and the VQQL baseline experiment

Outline

- Introduction
- Model pedestrian navigation as MARL
- State space generalization
- Algorithms
- Experiments
- Conclusion

Paper Conclusions

- The use of learning techniques provides several advantages for the pedestrian simulation field
  - The agents learn independently
  - The agent's learning is offline, so it has a very low computational overhead
  - Model free techniques such as Q-learning avoid introducing hand-coded domain knowledge into the system
  - The agent is not a closed system where its knowledge comes only from its own experience. RL incorporates external knowledge using transfer techniques.

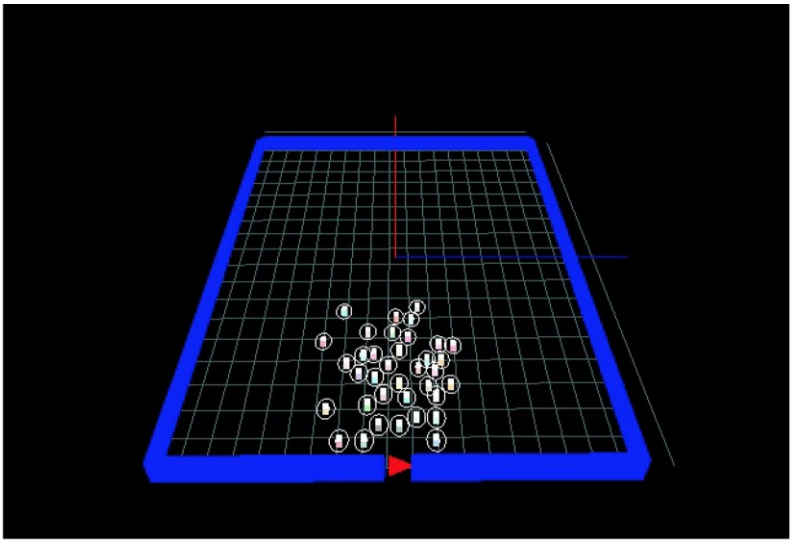
Paper Conclusions

- VQQL algorithm and its derivative schemas are convergent
- The simulation of these policies has confirmed the learning of the basic rules of pedestrian dynamics.
- The similarities with the Helbing model show that the agents have developed plausible behaviors of pedestrian.

My Conclusions

- Incorporate properties to the agents, such as age, gender, etc. to make a more realistic simulation.
- Room sizes are fixed. What would be the results in a more complex scenario with different room sizes?
- Adding multiple exits to the first scenario. The agents need to choose the best exit (closest and less crowded).
- Change of speed other than just when there is a possibility of collision.
- It would be interesting to see the results of the second scenario if there is not an even distribution of agents in the two groups.
- Make decisions depending on other agents' velocity to reach the goal faster (second scenario)

Video



https://www.uv.es/agentes/RL/itvqql.htm

Thank you

Any Questions?