ETH zürich

Complex Social Systems:
Modeling Agents, Learning
and Games

AI in Pedestrian Dynamics

M. Eichenberger, I. Heidtmann, A. Jürgens, L. Milanovic, D. Zehnder

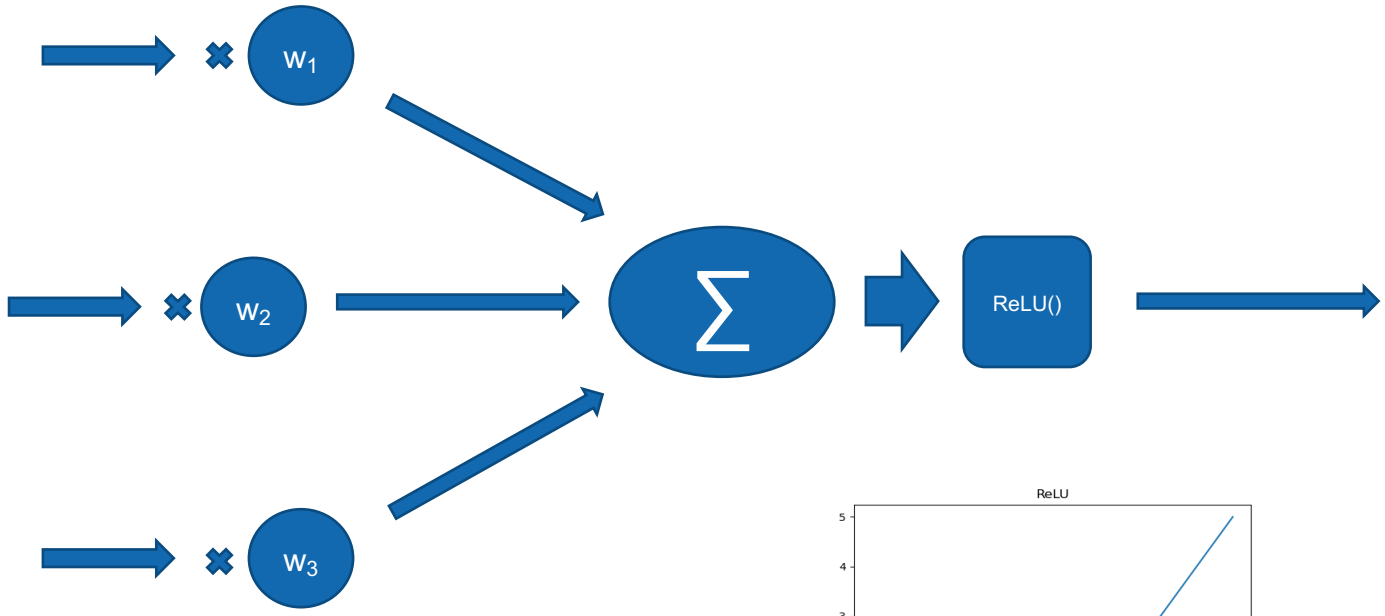
30. November 2020

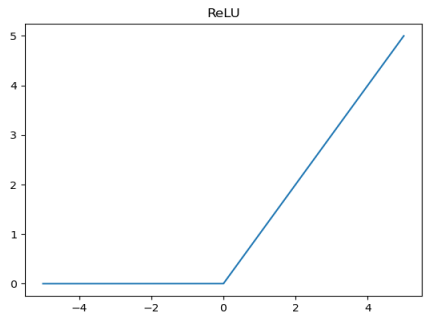
What are Neurons?

Simply:

$$f\colon \mathbb{R}^n \longrightarrow \mathbb{R}$$

- n inputs get multiplied with weights
- Everything gets summed up
- Non-linearity applied



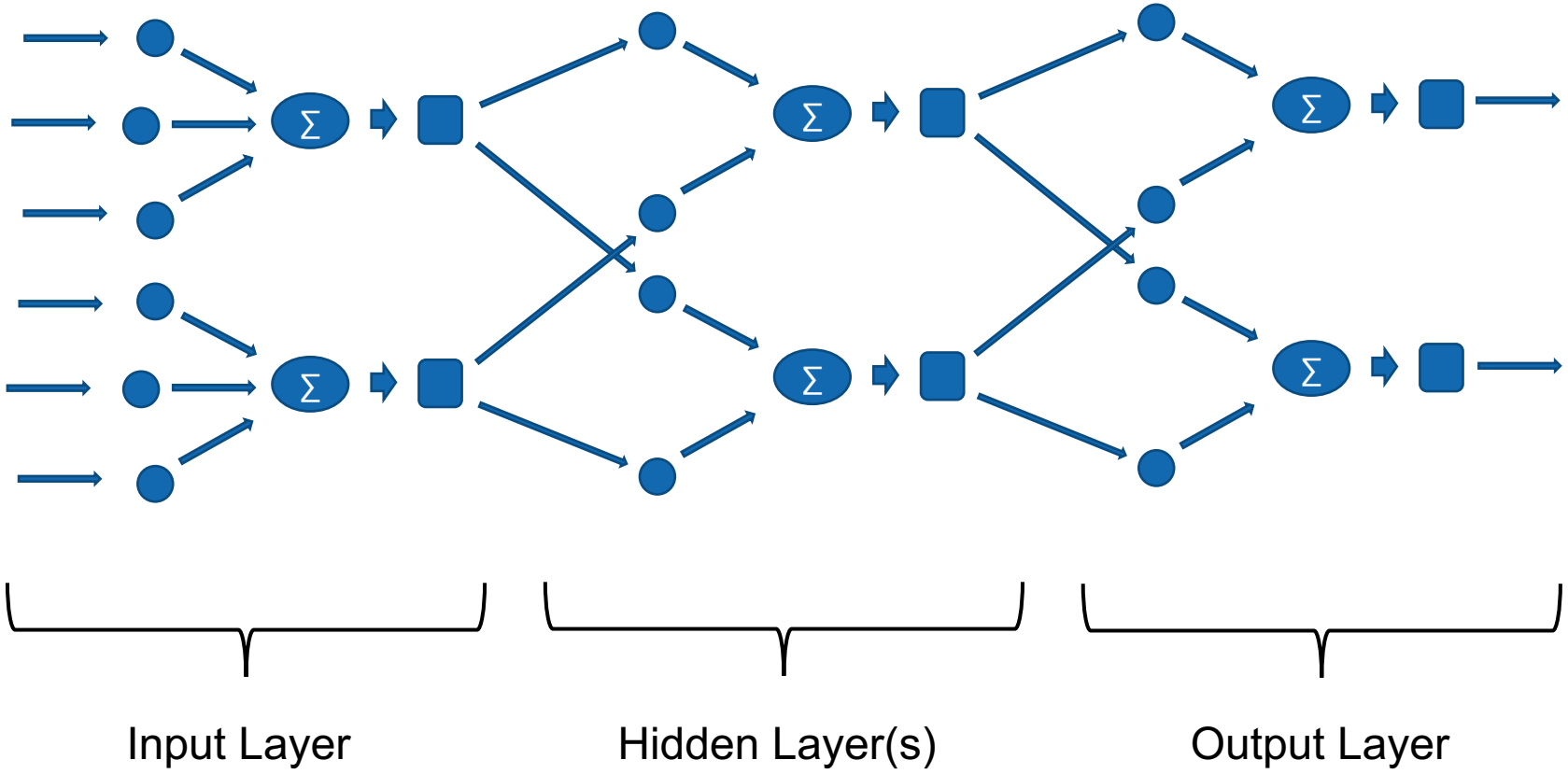


How to make networks out of them?

More complex function:

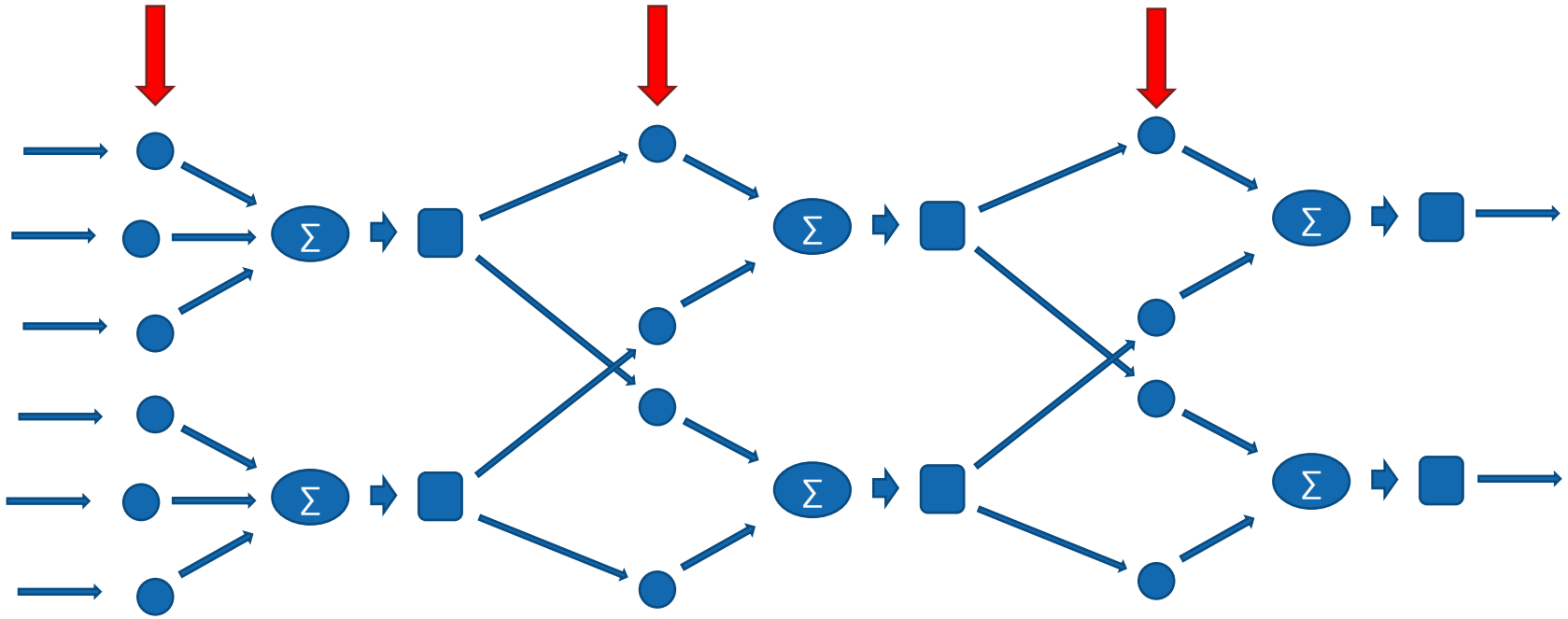
$$f\colon \mathbb{R}^n \longrightarrow \mathbb{R}^m$$

- "Universal approximation theorem"
- They come in all shapes and sizes:
  Feed Forward, RNN, CNN
- They can be trained!



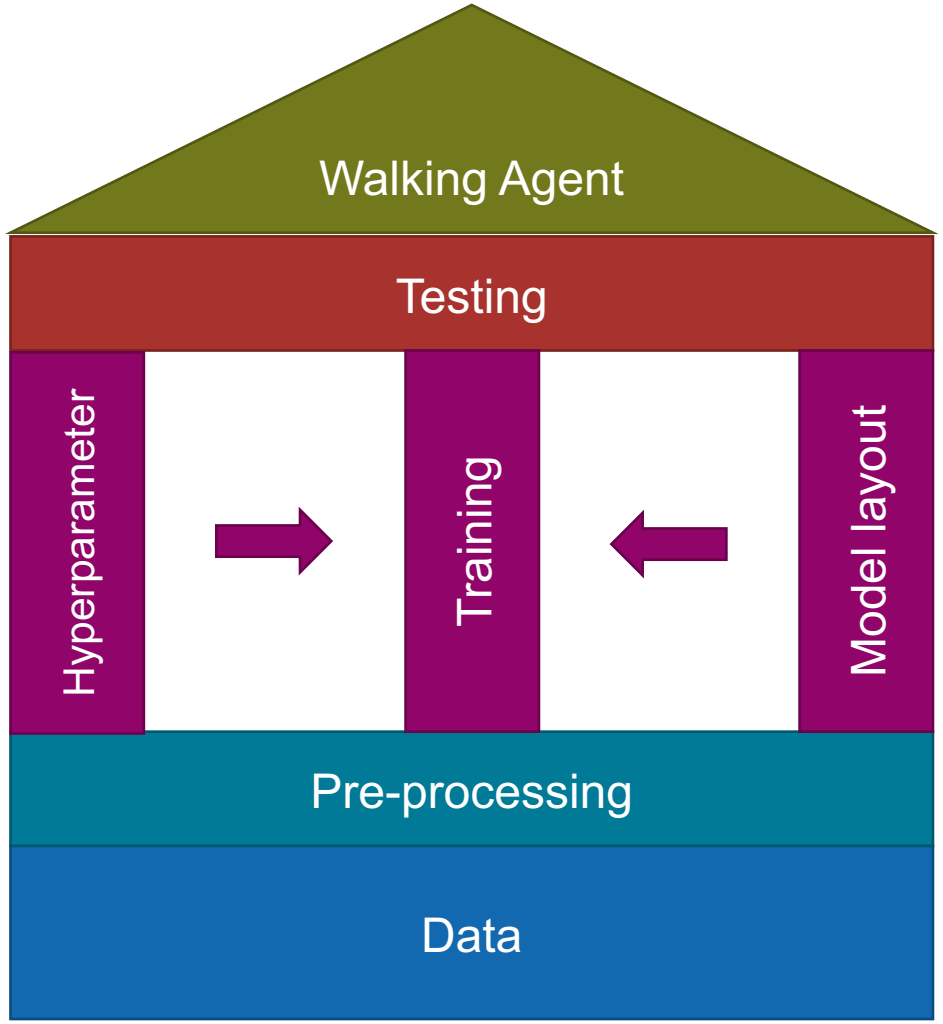
Training a computer?

- Training modifies the weights
- Minimizing the difference between output and “solution”
- The difference is called *Loss*
- Benefit: Network decides what is important and what is not



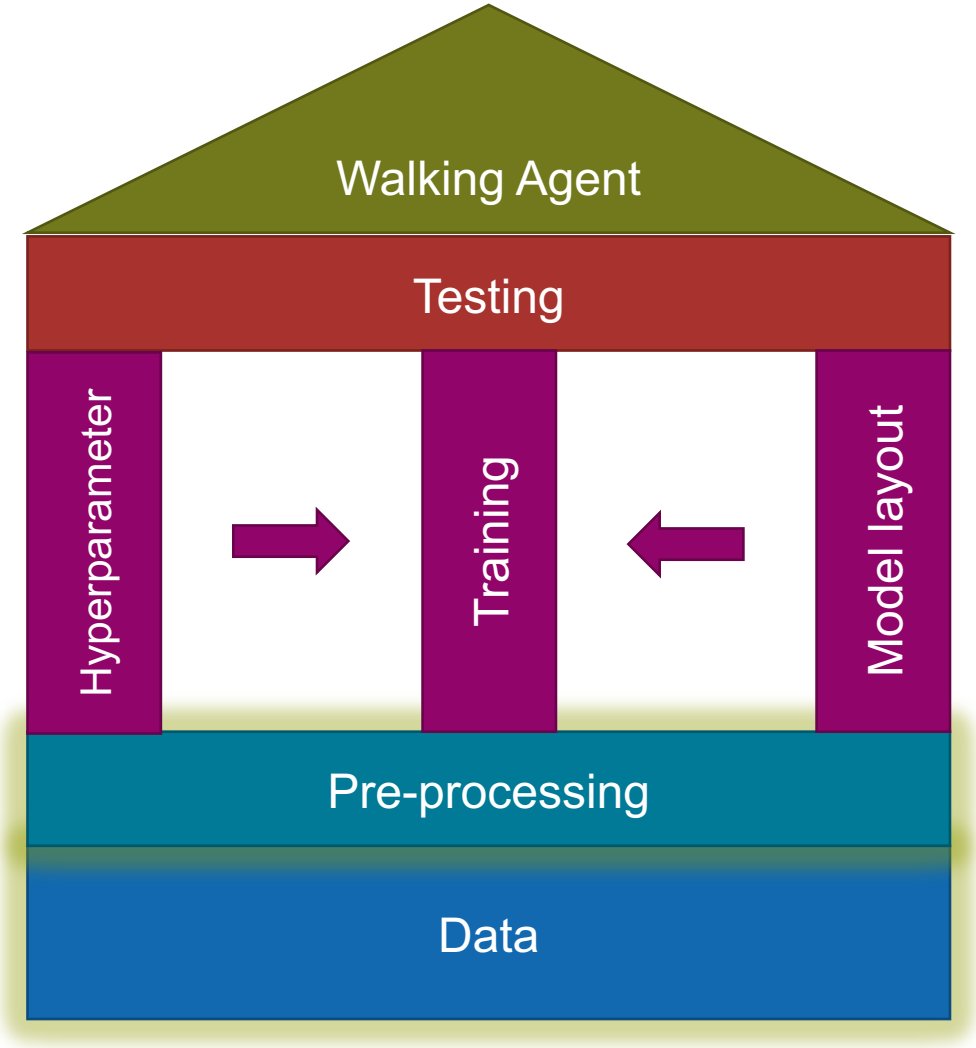
Methode

Overview



Methode

Data & Pre-processing

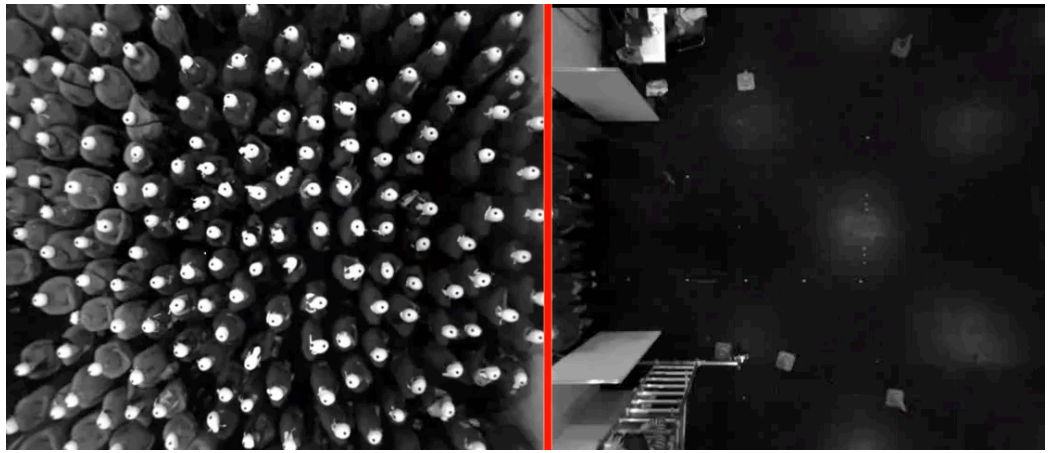


Methode - Data

- Civil Saftety Research, Forschungszentum Jülich, *https://ped.fz-juelich.de/database/doku.php*
- Trajectories generated from tracked video
- Experimental Situation

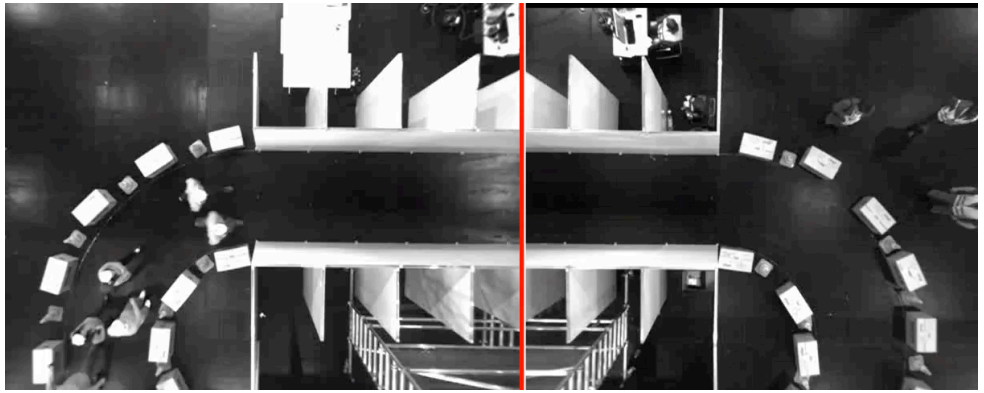
- Instruction : leave in ordered fashion

- Advantages:
  - Pre tracked
  - Experimental setup well documented
  - Different situations



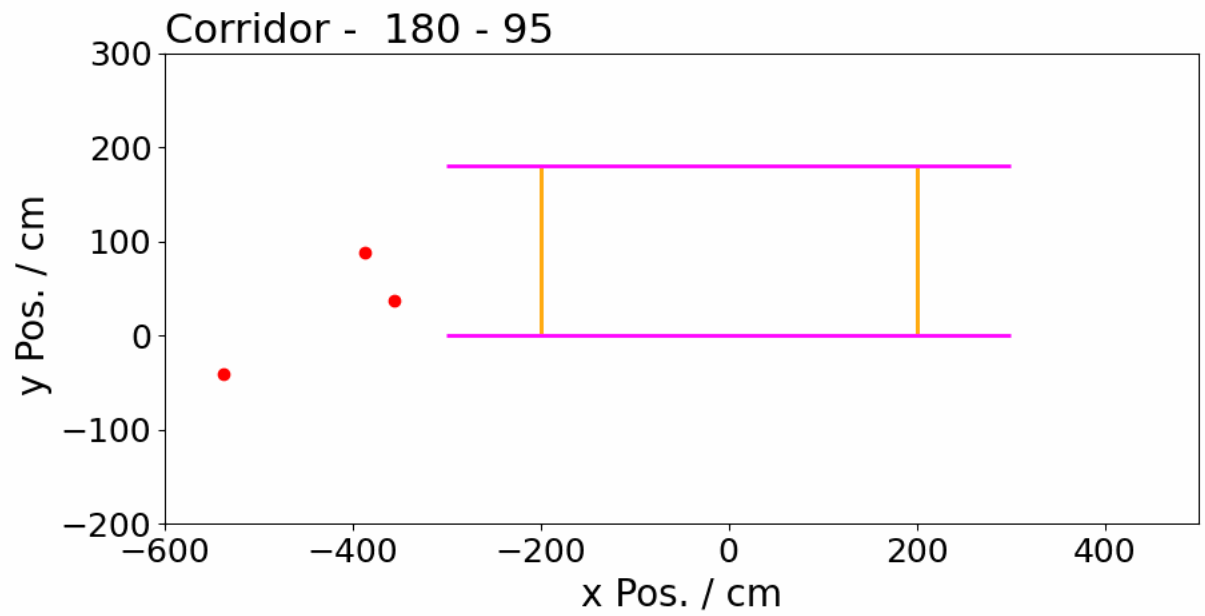
Methode - Data

Video: 16 FPS



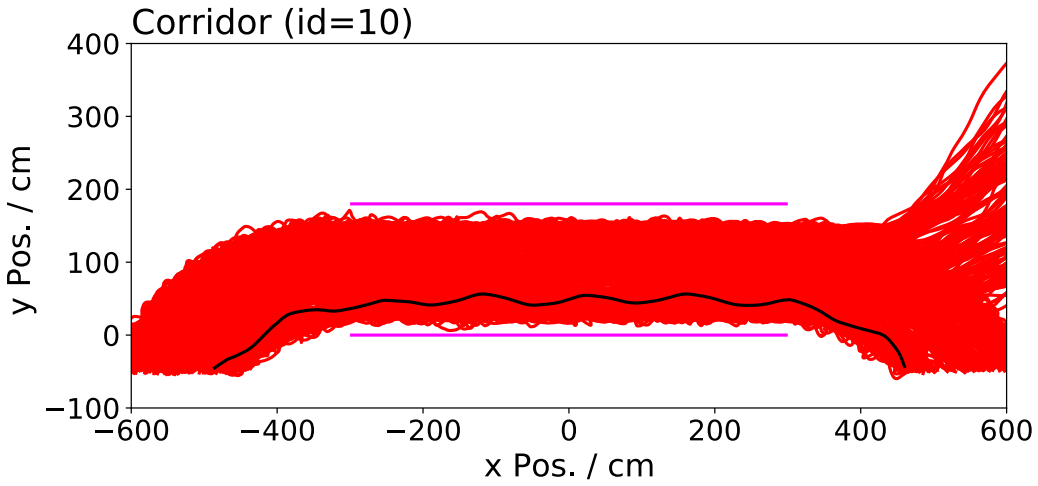
Trajectories:

- Encode geometry



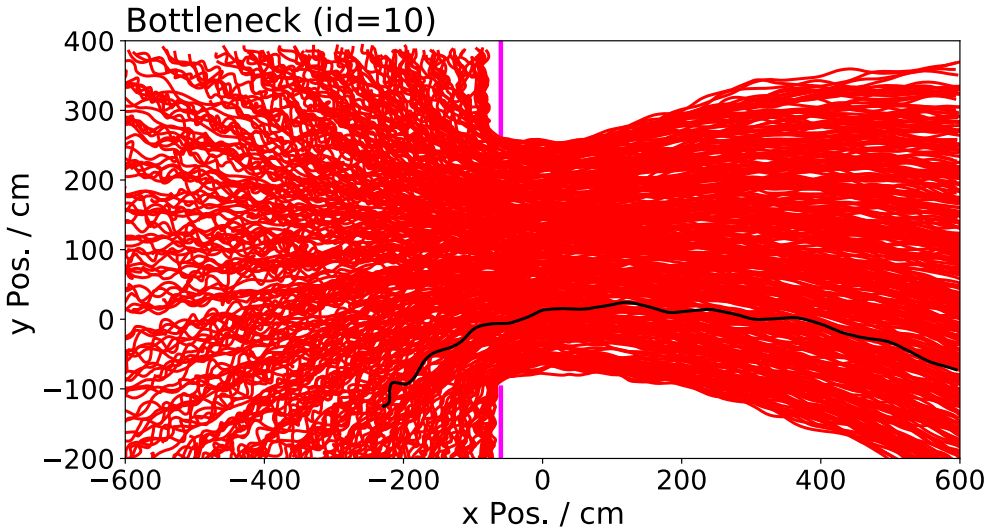
Methode - Data

Corridor



- Width: 180 cm
- Length: 600cm
- People: 15-230
- ~2000 – 40’000 steps for Training

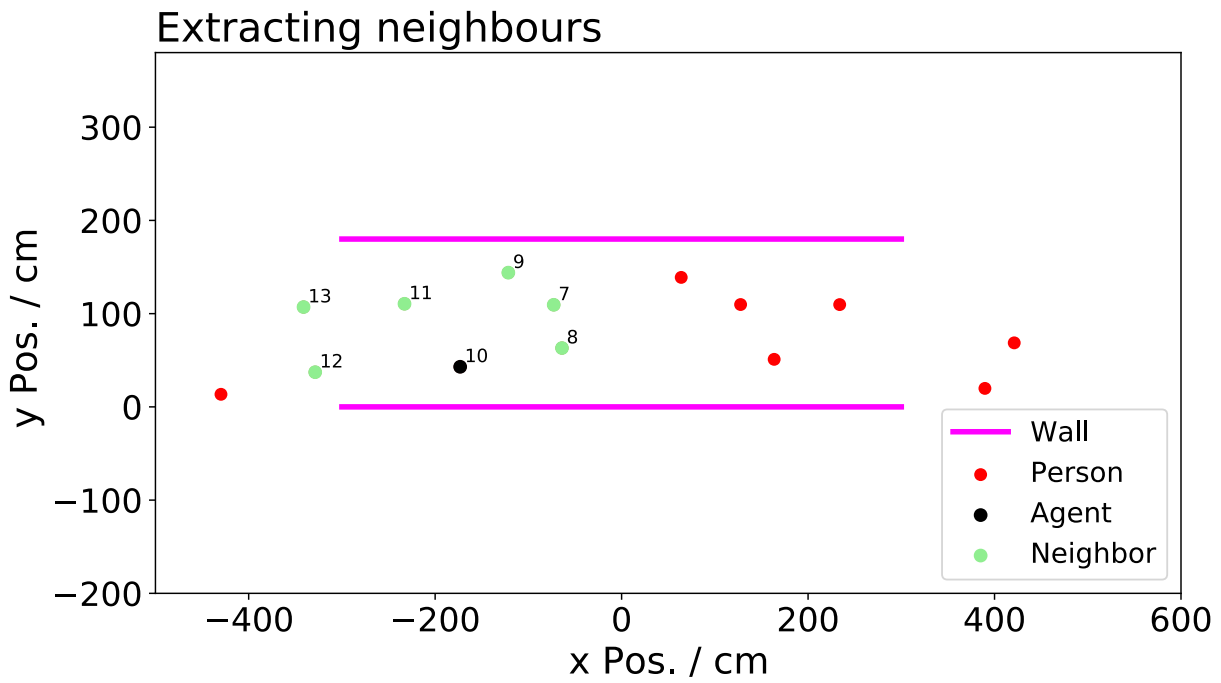
Bottleneck



- Width: 360 cm
- People: 400
- ~20’000 steps for Training

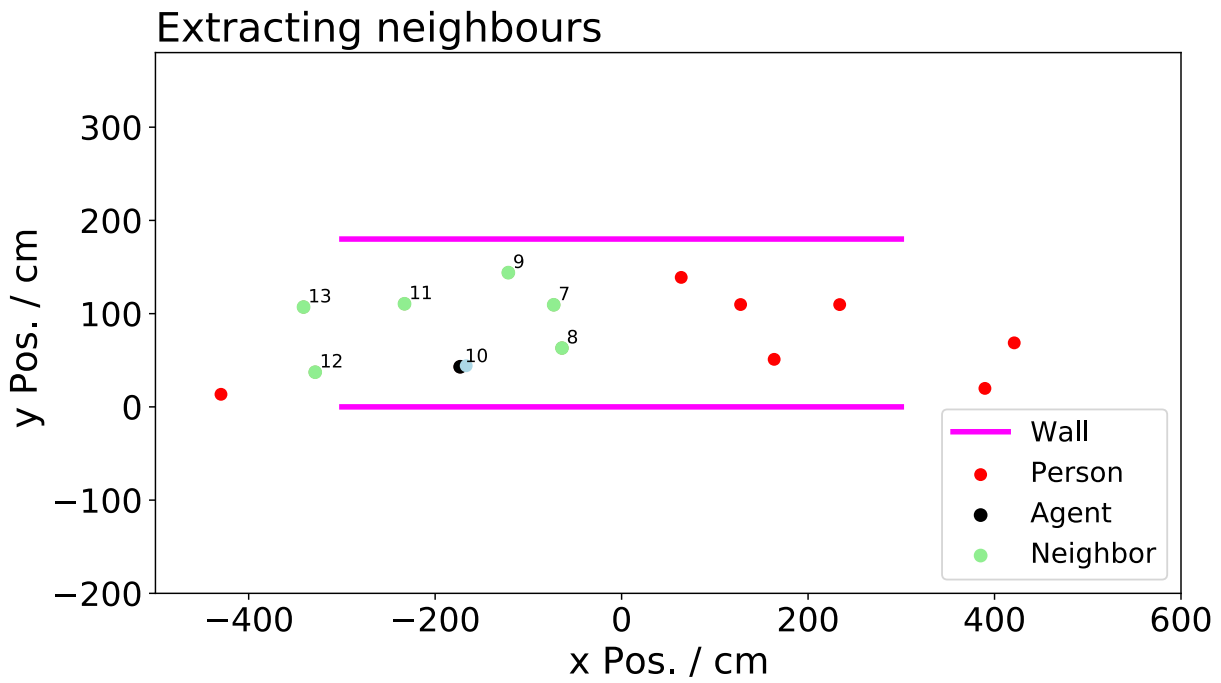
Methode - Data

- Truth extracting
  - Nearest Neighbours



Methode - Data

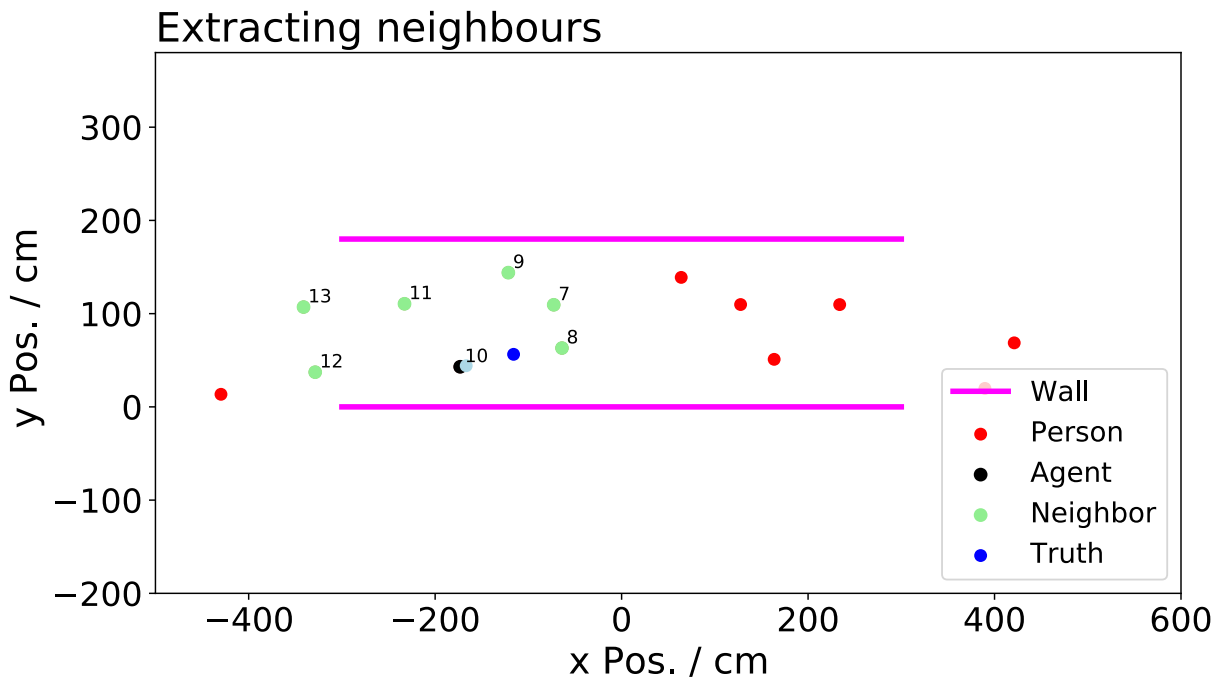
- Truth extracting
  - Nearest Neighbours
  - Next Frame



Methode - Data

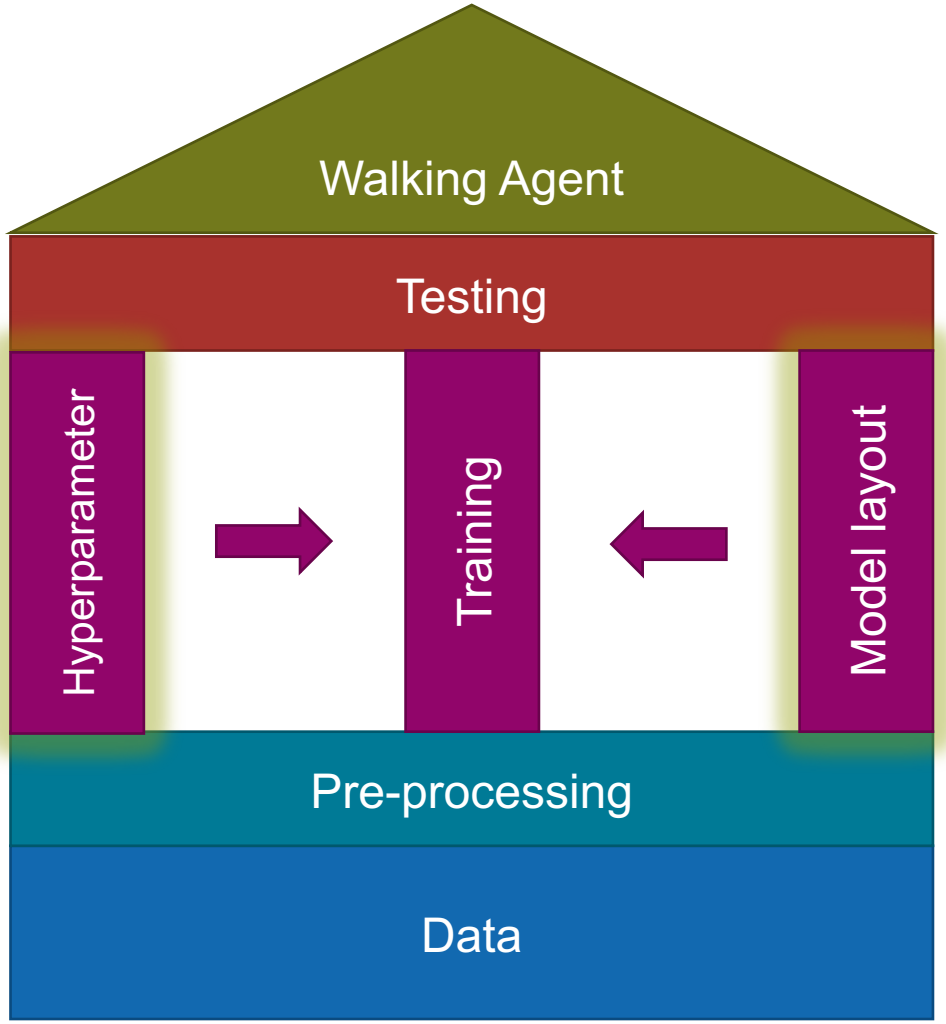
- Truth extracting
  - Nearest Neighbours
  - Next Frame

- Down sampling -> 2FPS
- Wrapping



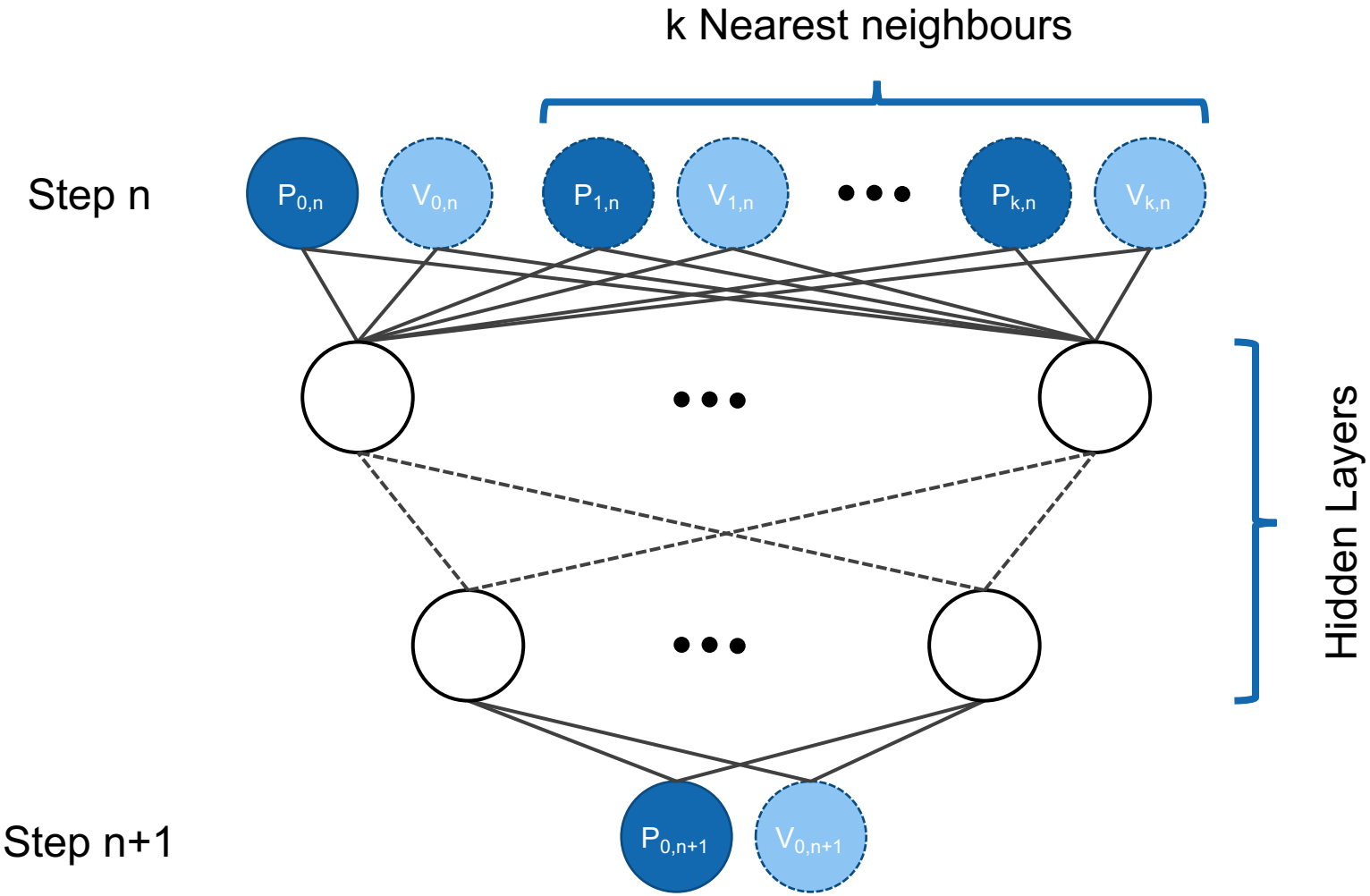
Methode

Model – Feed Forward



Methode - Model

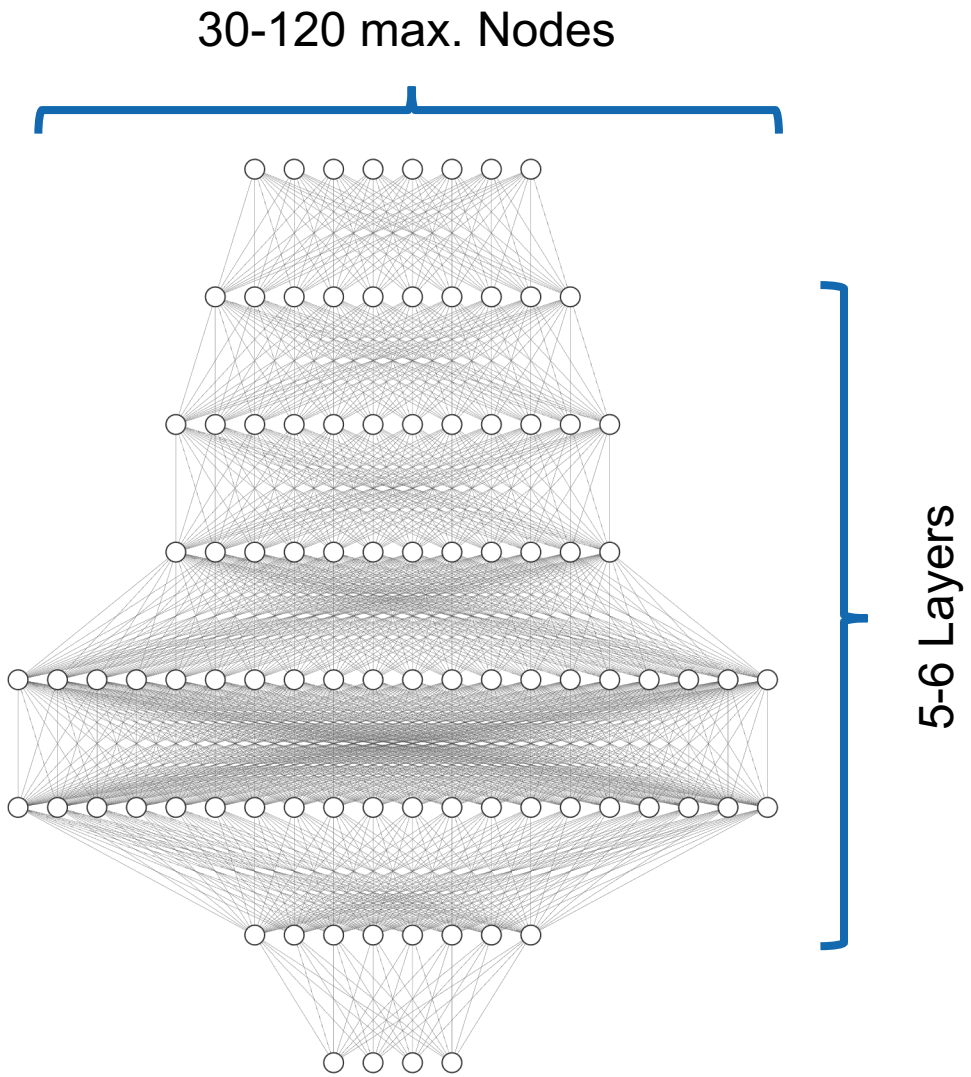
- Input:
  - Number of neighbours
  - Use velocities
  - Use past steps
- Output:
  - Position
  - Velocities



Methode - Model

- Hidden Layers:
  - 5-6 Layers (shallow)
  - Approx. 30-120 Nodes per layer
  - Literature: Vehicle prediction network[1]

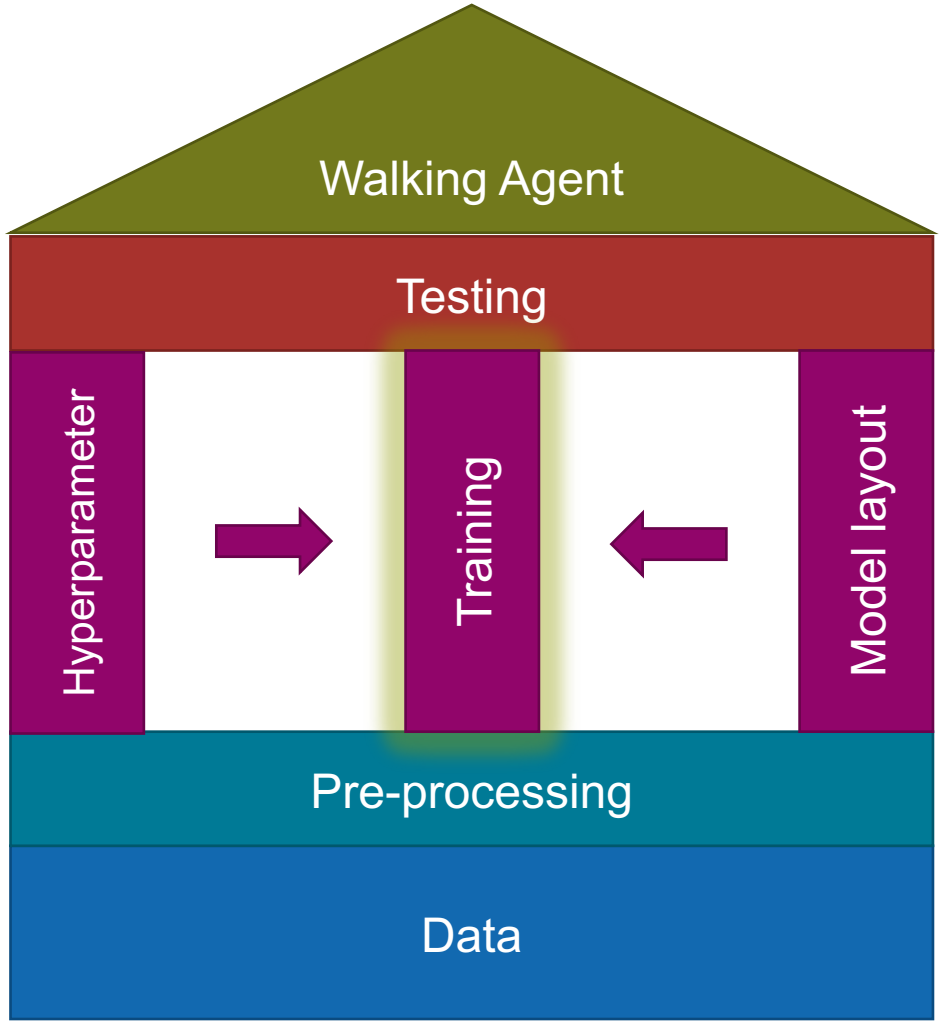
- Current Network :
  - Input: 6 Neighbours with velocities (28)
  - Hidden layers : (32, 32 ,64, 128, 128, 40)
  - Output: Position & Velocities (4)
- Bottleneck: Hidden Layer: (50, 20, 50)



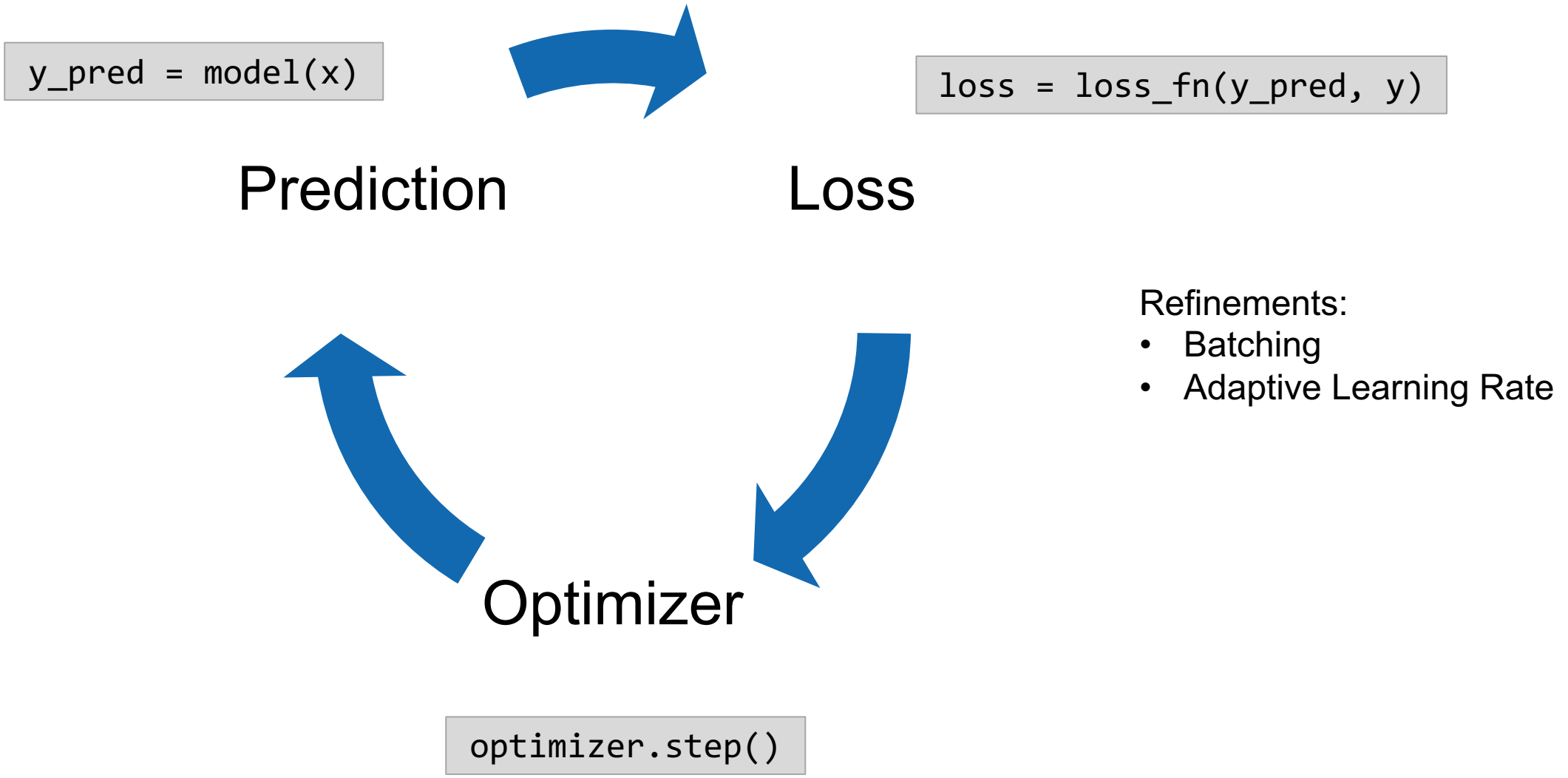
- [1] Feed-forwards meet recurrent networks in vehicle trajec-tory prediction, M. Bahari, A. Alahi, EPFL
- [2] Visualize Architecture of Neural Network, http://alexlenail.me/NN-SVG/index.html

Methode

Training



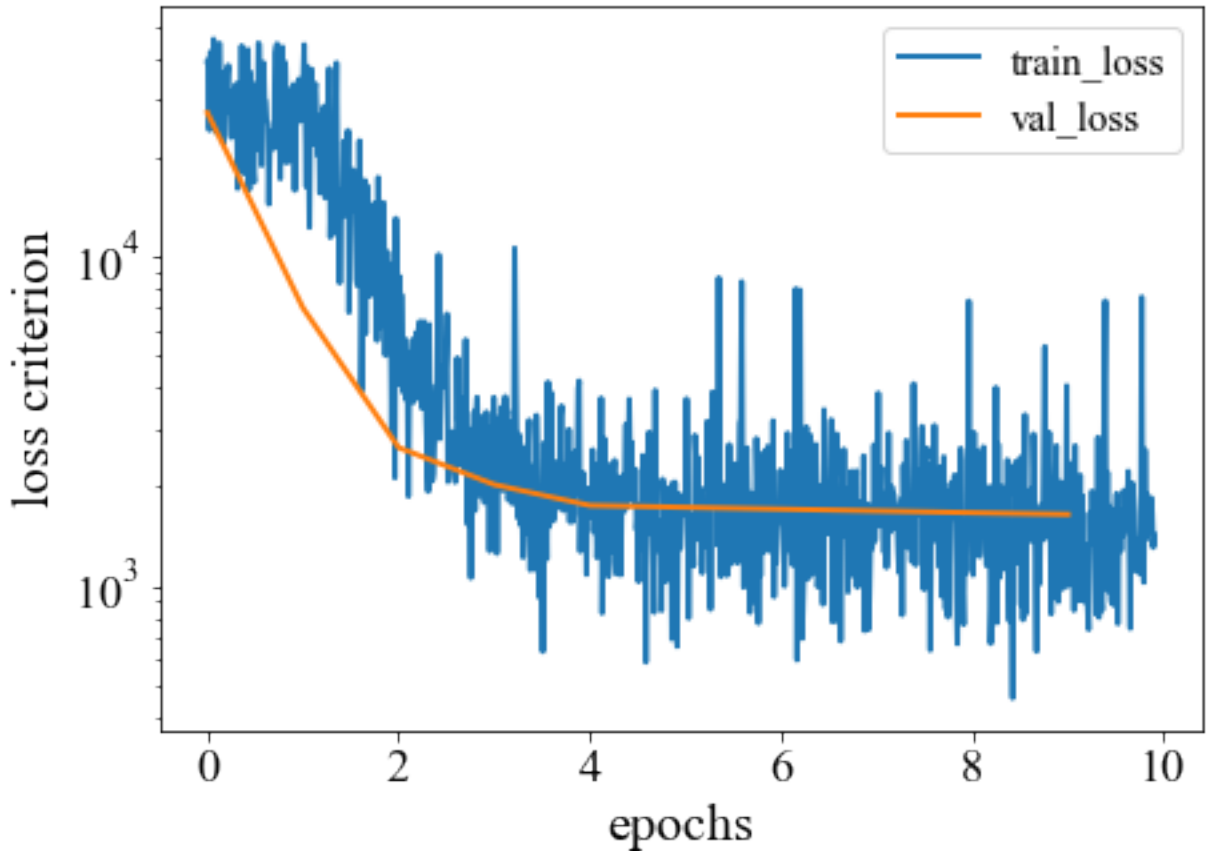
Training the Neural Network



Refinements:

- Batching
- Adaptive Learning Rate

Training the Neural Network



Working process

1

Preparation
- Data sets
- Papers
- Pytorch

2

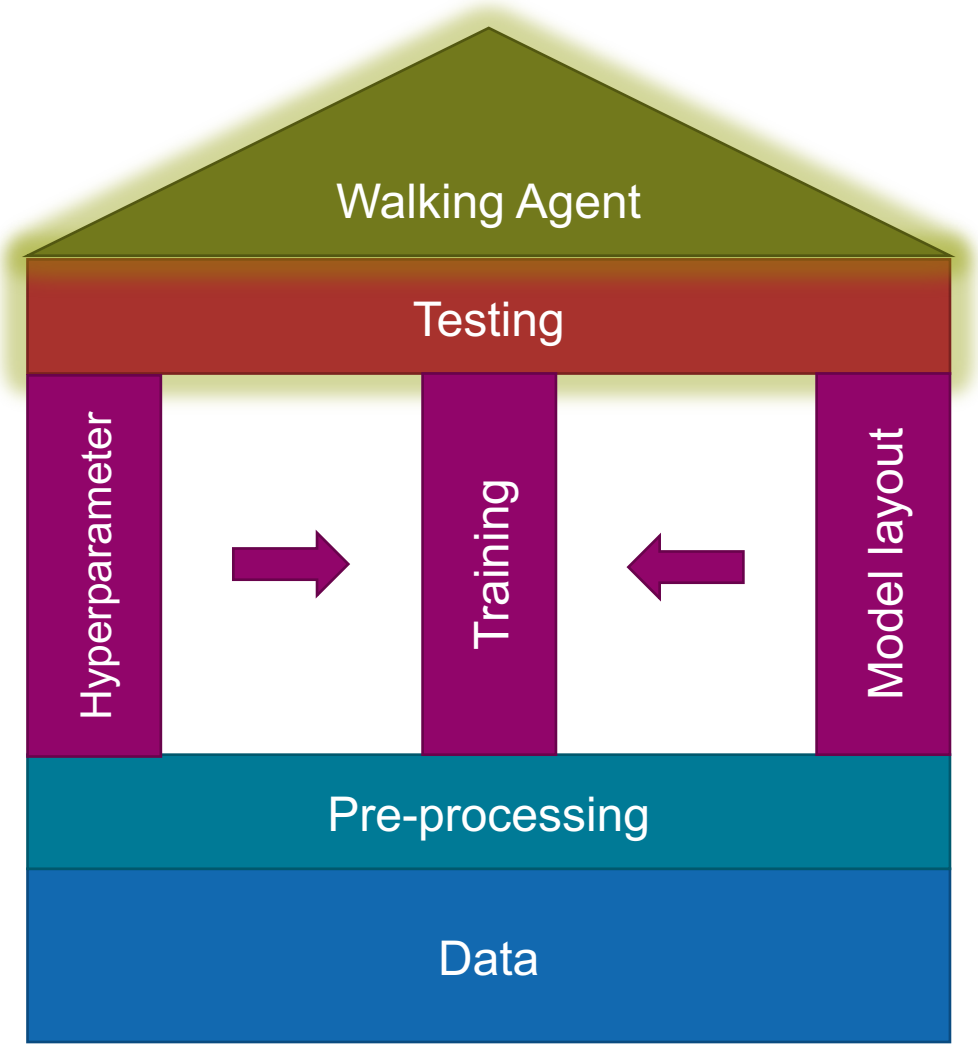
Implementation
- Auxiliary libraries
- Various neural networks

3

Wrapping Up
- Comparing results in pedestrian dynamics context

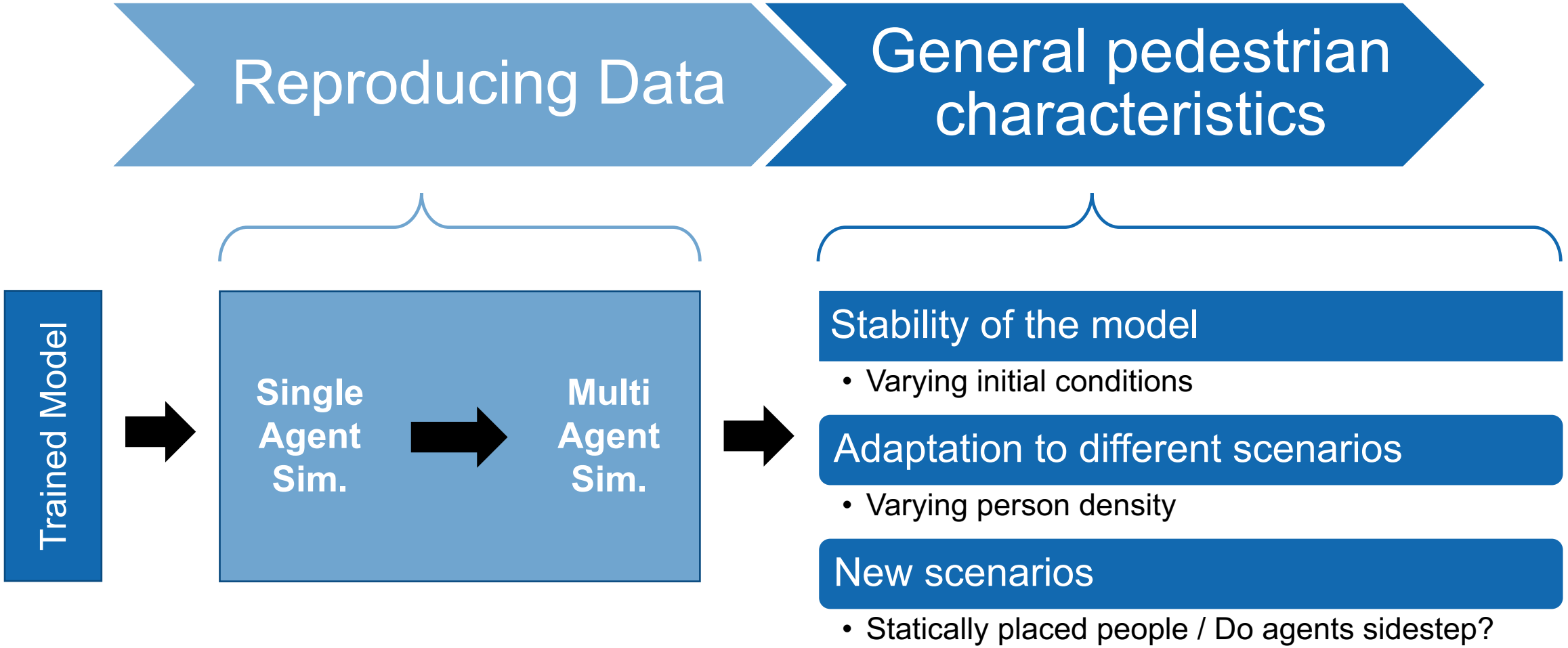
Methode

Walking Agent(s)



Results: **Overview**

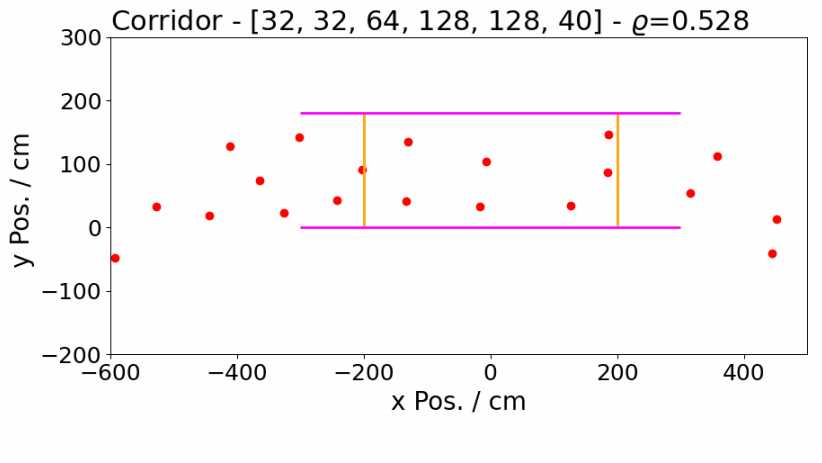
How good does our trained model capture the behavior of a pedestrian?

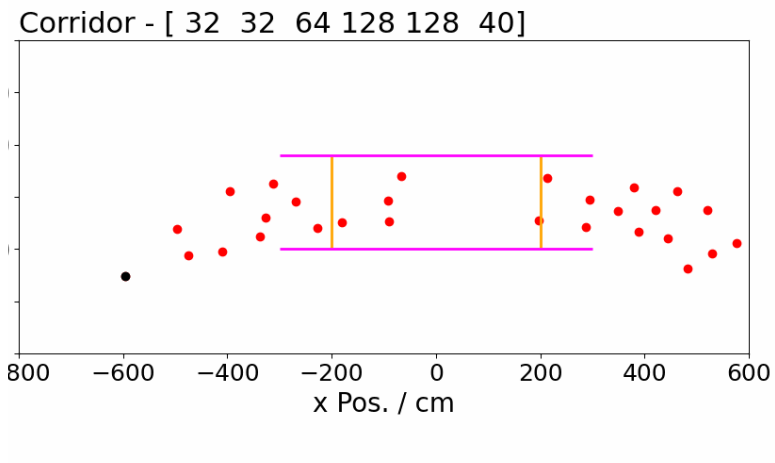


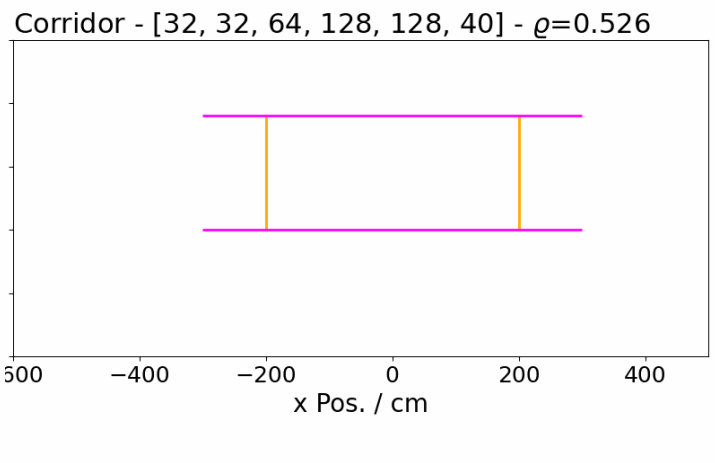
Results: **Reproducing Data**

Raw data → **Single Agent** → **All Agents**

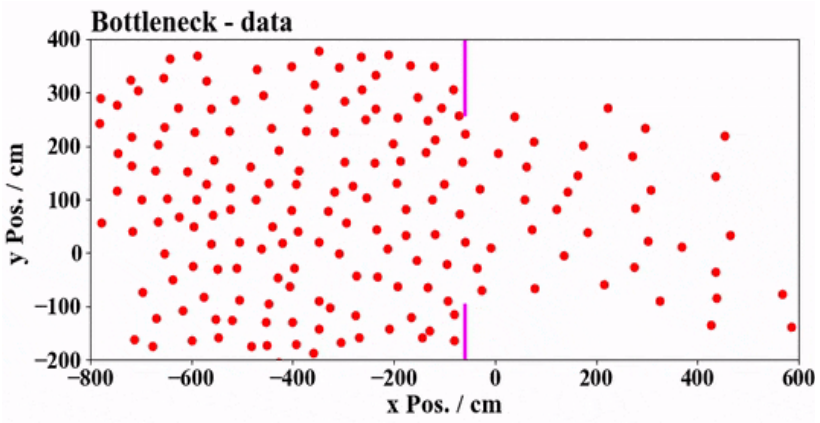
Corridor

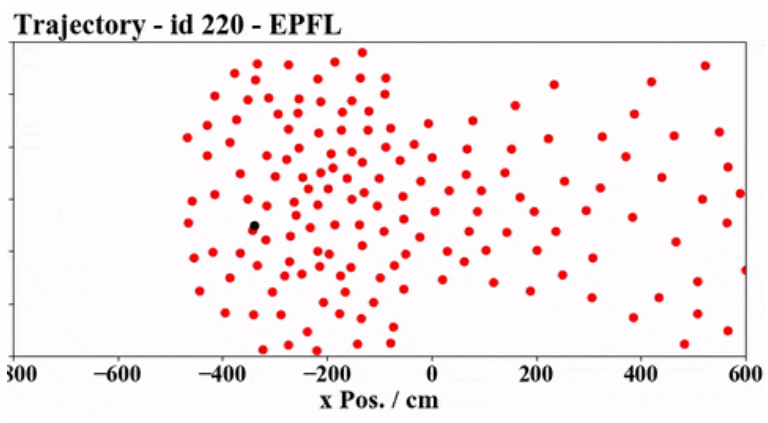


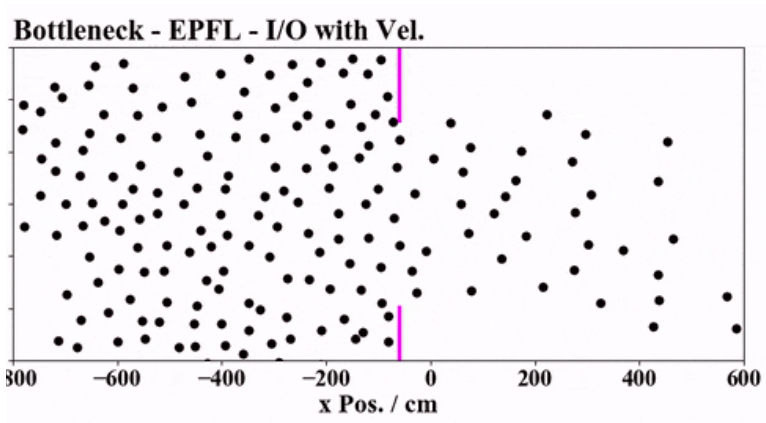




Bottleneck



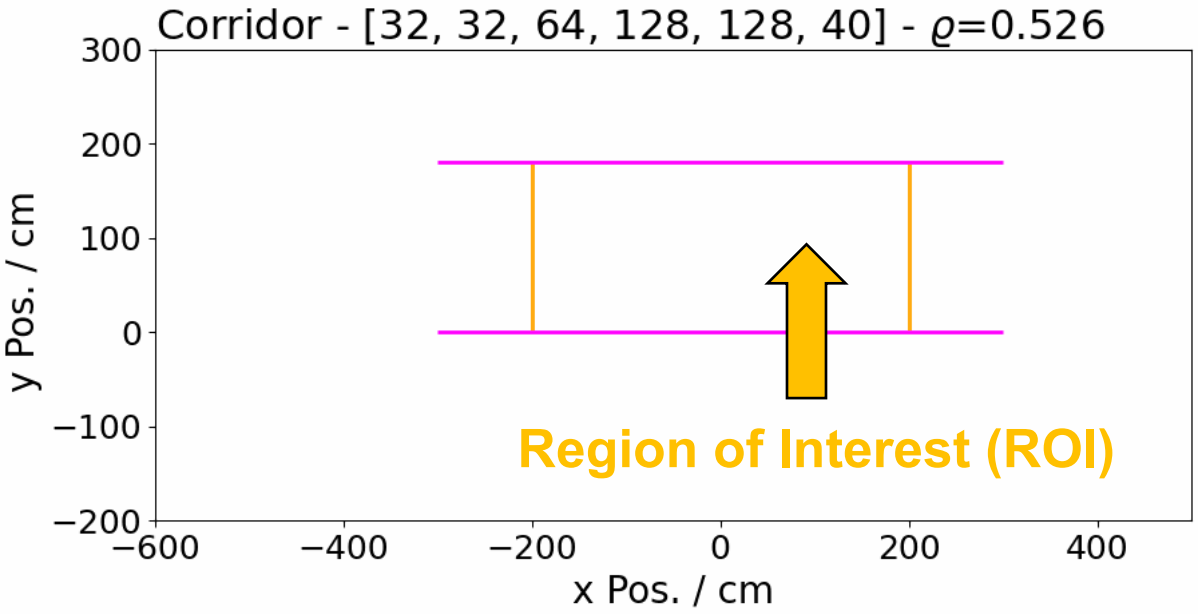


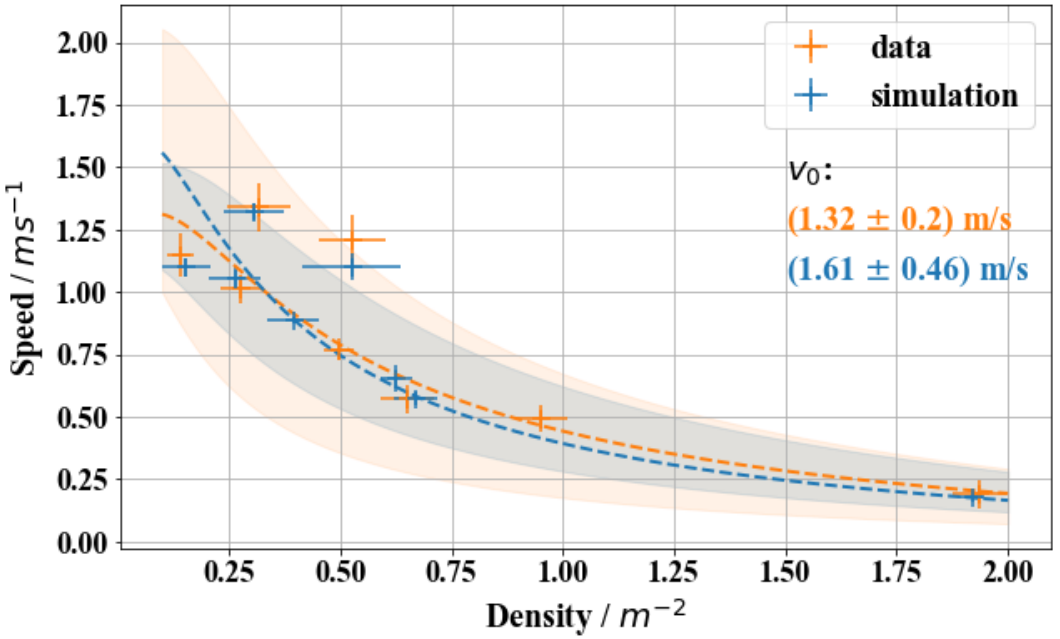


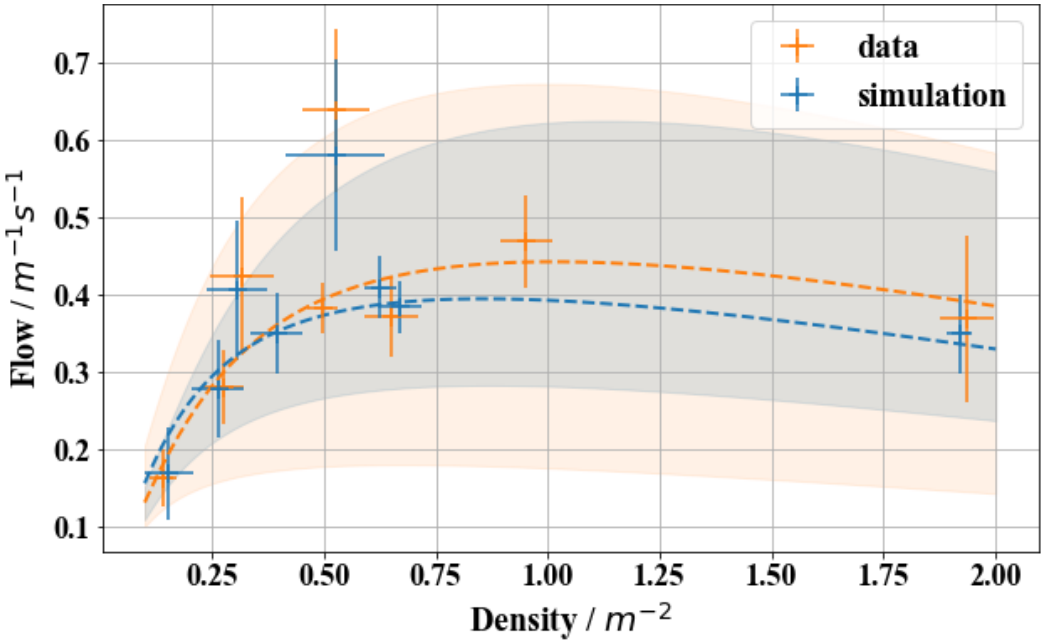
Results: **Corridor**

- **Apply <u>one</u> Model to different scenarios**
  vary people density
- Measure mean density and speed in **ROI**
- Average over frames

Problematic:
Inhomogeneous raw data (fluctuations in speed)



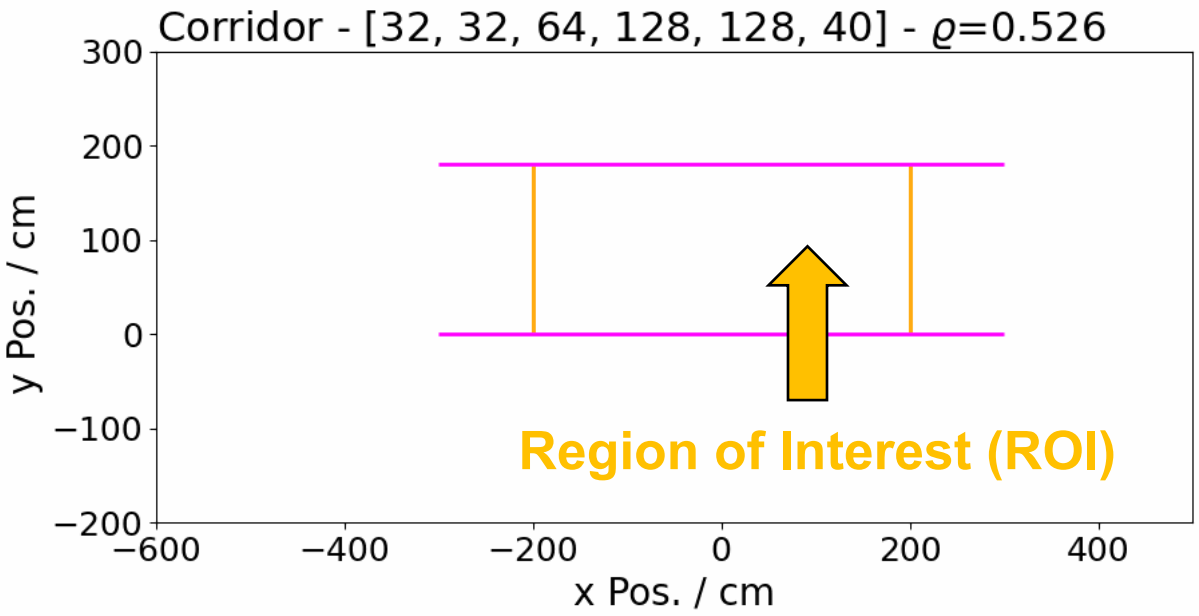


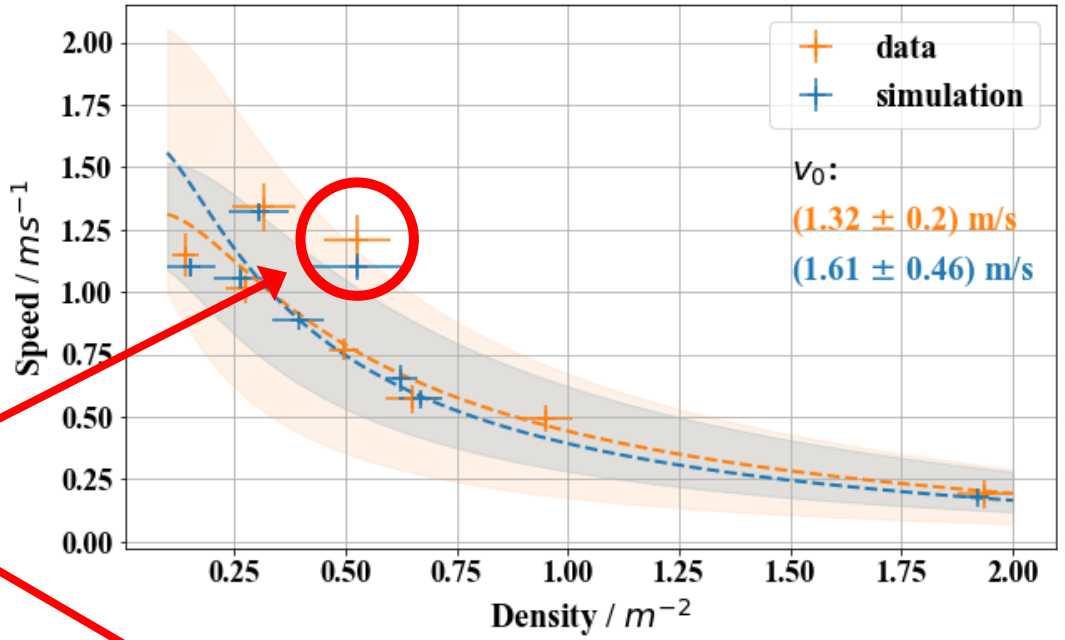


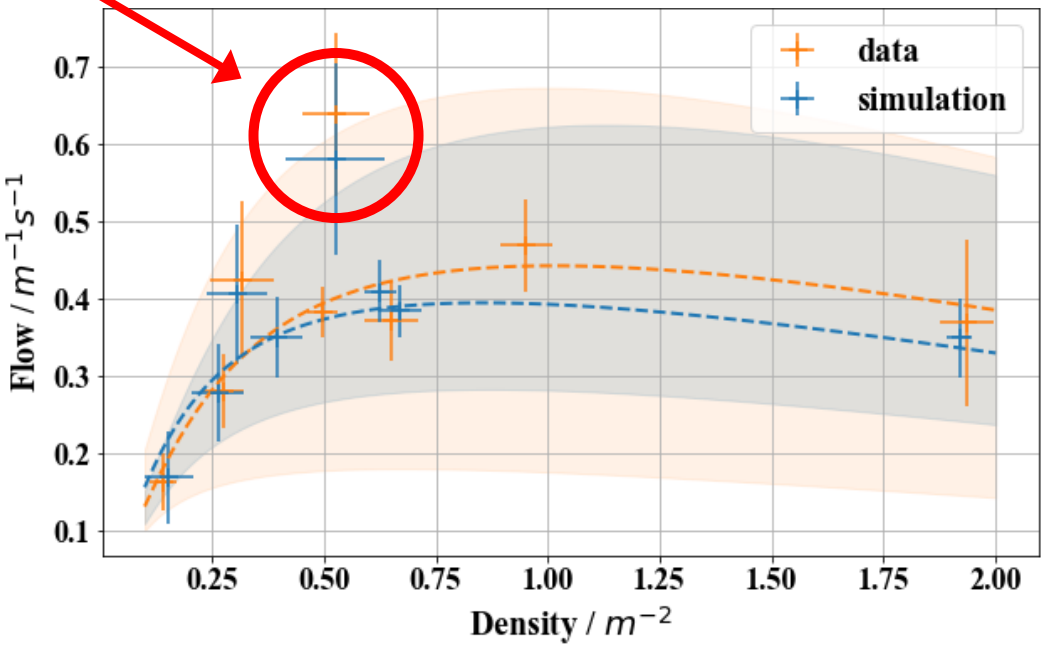
Results: **Corridor**

- **Apply <u>one</u> Model to different scenarios**
  vary people density
- Measure mean density and speed in **ROI**
- Average over frames

Problematic:
Inhomogeneous raw data (fluctuations in speed)

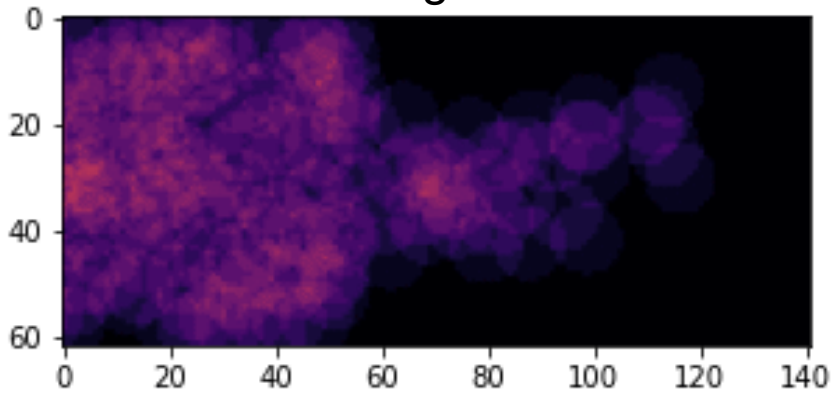




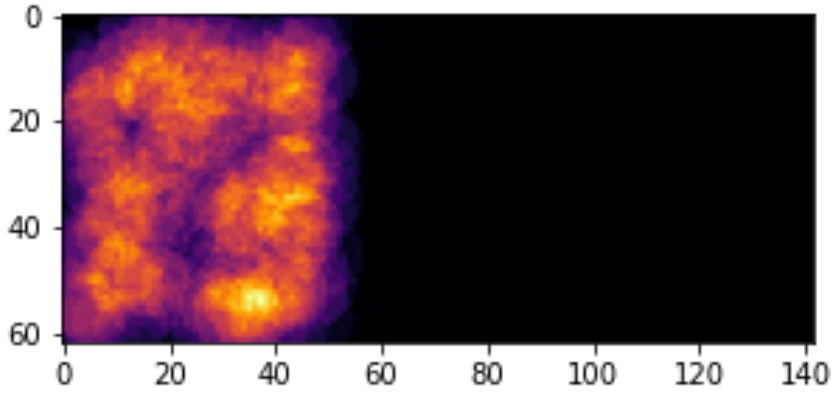


Results - Bottleneck

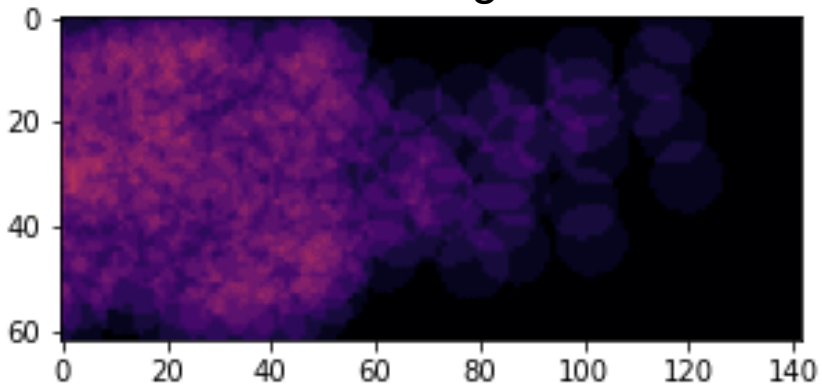
Training Data



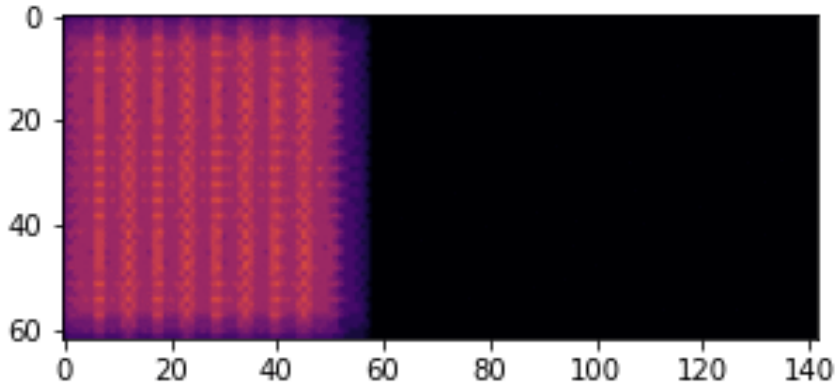
Simulation with random initial positions



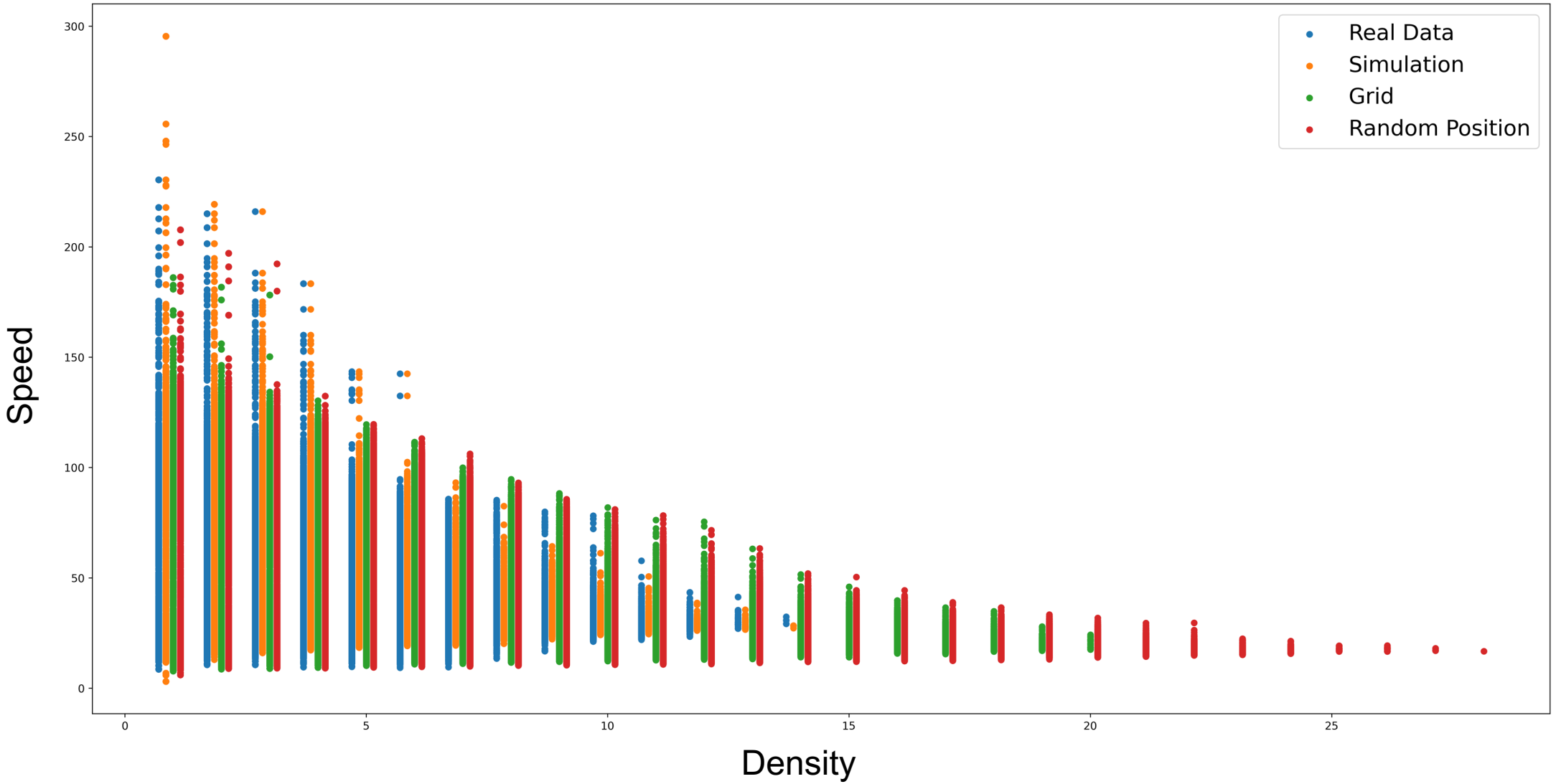
Simulation with initial positions from trainig data



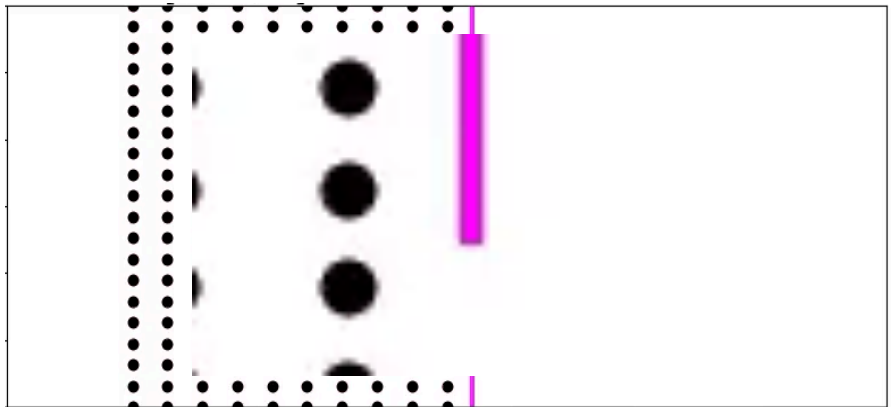
Simulation with homogenously distributed initial positions

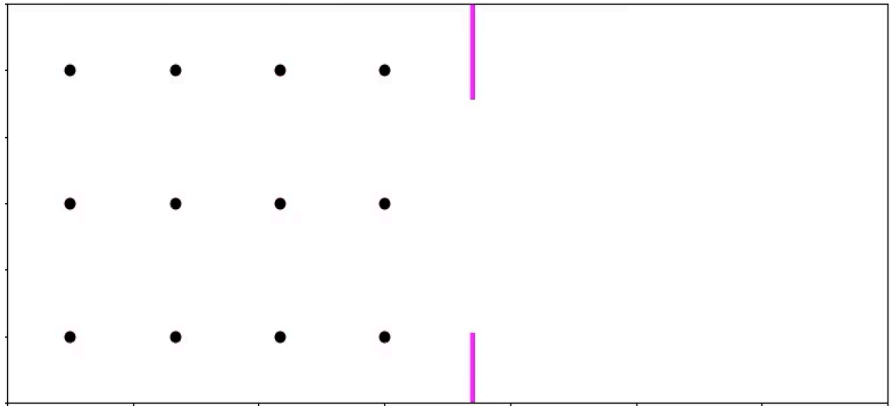


Results - Bottleneck



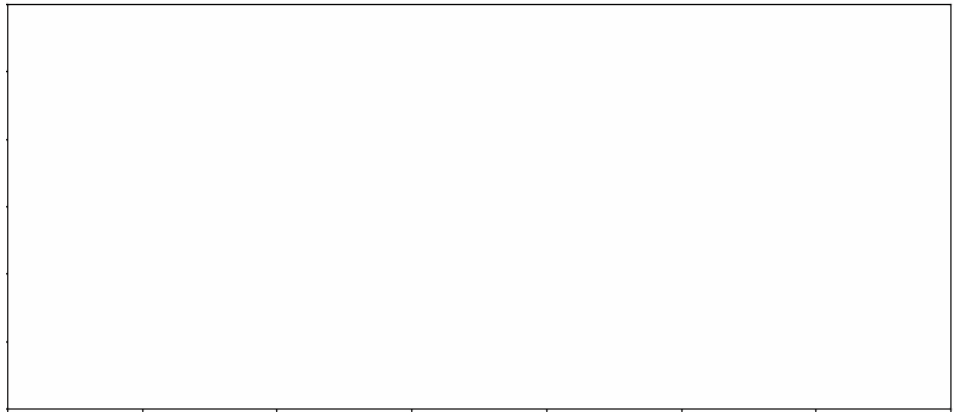
Limitations of our approach





- Geometry of problem learnt on a data-driven basis
  - Difficult to adapt to new environment
  - Agents not forced to respect the geometry of the Problem
- Agents adhere less to the geometry at low densities
- Does not simulate fundamental changes in human behaviour such as panicking

Outlook



Possible Future Approaches

- Modify the architecture to ensure geometry compatibility
  - Hard-code geometry into network architecture
  - LSTMs?
  - Generative Adversarial Networks?

- Apply same method to similar problems
  - Bidirectional Flow, Intersections, …
  - Different Agents such as cars, animals, …
  - 3D setups such as stairs, hills, roads,…