National College of Ireland

Developing motion planning algorithms of autonomous delivery robots for smart cities

MSc Research Project
Artificial Intelligence

Jamilya Nurasheva
Student ID: 23284757

School of Computing
National College of Ireland

Supervisor: Arundev Vamadevan

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Nurasheva Jamilya
Student ID:	23284757
Programme:	MSc in Artificial Intelligence **Year:** 2025
Module:	Practicum
Supervisor:	Arundev Vamadevan
Submission Due Date:	11/08/2025
Project Title:	Developing motion planning algorithms of autonomous delivery robots for smart cities
Word Count:	…7929…… **Page Count**……20……

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: …Jamilya……

Date: …07/08/2025……

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Developing motion planning algorithms of autonomous delivery robots for smart cities

Jamilya Nurasheva
23284757

Abstract

The widespread development and integration of artificial intelligence and robotics technologies in all areas of our lives, including logistics, has contributed to the emergence and development of autonomous delivery robots (ADRs). Courier robots can reduce delivery costs and carbon emissions and improve the efficiency of the ‘last mile’. However, developing ADRs is a complex task, as many details must be taken into account, such as static and dynamic obstacles on the terrain, the unpredictable behaviour of pedestrians, cyclists and other road users, as well as the complex infrastructure of the city.

This work is dedicated to the study and comparative analysis of three approaches to motion planning: the classic A* algorithm, the Social-LSTM machine learning model, and the A*+Social-LSTM hybrid. Two open, publicly available datasets were used to train and test the approaches: nuScenes and Stanford Drone Dataset (SDD).

The ADE, FDE, and RMSE metrics were selected to evaluate the performance of the algorithms. The results of the study showed that the classic A* method performs well in static conditions, while the Social-LSTM-based model demonstrated high accuracy in predicting the agent's trajectory but did not take into account the physical infrastructure of the map. The hybrid model performed best, showing a balance between adaptability and structure.

Keywords: motion planning, autonomous delivery robots, Social-LSTM, A*, hybrid methods, trajectory prediction.

1 Introduction

In today's world, artificial intelligence technologies are rapidly developing and being actively implemented and applied everywhere, including in the field of logistics. One promising area in this industry is the development of autonomous delivery robots (ADR) as part of the ‘smart city’ concept. Such systems are being actively developed as part of last-mile delivery solutions using autonomous agents, including ground and aerial robots (Agatz *et al*., 2016). ADR are mobile ground platforms capable of delivering cargo without human intervention. They reduce delivery costs, cut carbon emissions and increase the efficiency of the last stage of the logistics chain. However, the use of ADRs in urban environments is fraught not only with technical and research challenges — such as trajectory planning, predicting the behaviour of other road users, and adapting to the dynamic conditions of the urban environment — but also with social barriers. According to research, society's readiness to accept and interact with robots plays an important role in the implementation of ADR in modern ‘smart cities,’ which is influenced by human trust in them (Patel, 2024).

The aim of this study is to determine the best algorithm applicable to the navigation of autonomous delivery robots and the prediction of the behaviour of surrounding road users. The project proposes a solution to the ADR motion planning problem in a constantly changing environment, taking into account the interaction of the robot with both static and dynamic surrounding objects.

This project discusses both traditional route planning methods, such as the A* algorithm, and modern approaches based on machine learning: deep neural networks (Social-LSTM) and hybrid architectures. Particular attention is paid to pedestrian behaviour prediction models, which allow social interactions to be taken into account and more realistic navigation decisions to be made.

Open datasets are used to train and test the models: nuScenes (for navigation and sensor information) and Stanford Drone Dataset (for agent behaviour analysis). RMSE, ADE, and FDE are proposed as metrics, which will allow for a comprehensive assessment of ADR behaviour in various situations.

The scientific contribution of this research lies in the use of a hybrid architecture: the A* search algorithm and Social-LSTM machine learning. A special feature of the project is the use of two representative city datasets — the Stanford Drone Dataset for training and nuScenes for testing, which allows the models to be evaluated for their applicability to real urban conditions. The project also contributes to the replicability of research through the use of open data and a transparent evaluation methodology.

Since this project will implement a hybrid approach, it is expected to combine the reliability and determinism of classical algorithms, such as A*, with the adaptability of machine learning methods (in this case, Social-LSTM) to dynamic environments. Thus, such a solution will compensate for the weaknesses of each area, creating more flexible and effective solutions for navigation in real urban environments. The basis for the development of these solutions will be formed on the basis of existing scientific works and practical implementations. Since the field of autonomous delivery robots is still in its infancy, the results of this work may not only fill an existing scientific gap, but also serve as a basis for further applied and academic research in the field of mobile robotics and intelligent transport systems.

The report is structured as follows. Section 2 provides an overview of related research, critically reviewing existing classical and neural network approaches to autonomous robot navigation and agent behaviour modelling. Section 3 describes the research methodology, including a description of the selection of models, architectures, tools and data sources. Section 4 provides a specification of the designed architecture. Section 5 is devoted to a more detailed description of the selection and implementation of the proposed solution and the tools used. Section 6 evaluates the results and provides statistical analysis, followed by a detailed discussion. The concluding Section 7 summarises the key findings of the research and formulates suggestions for future work.

Research questions: Which architectures are most effective for improving the quality of motion planning of autonomous robotic couriers in smart city environments? Which architectures are most effective for improving the interaction of autonomous delivery robots with pedestrians and other road users?

2 Related Work

Developing effective navigation systems for autonomous delivery robots in urban, constantly changing environments is an important task. There are many challenges that such systems face, such as the need to adapt to a dynamic environment, taking into account the behaviour of pedestrians and other objects on the road, and selecting the optimal route.

Modern approaches can be divided into two main groups: traditional algorithms (e.g., A*, DWA, Dijkstra) and machine learning-based methods, which use large amounts of data for training, predicting the behaviour of surrounding objects, and making decisions.

Despite the fact that ADR appeared relatively recently – the first prototype was created in 2017 (Hoffmann *et al.*, 2018) – there are already a sufficient number of articles on both the delivery robots themselves and approaches to planning their routes in dynamic conditions.

This chapter provides a critical review of existing scientific work in the field of motion planning. Subsection 2.1 will look at work on traditional (classical) approaches, while subsection 2.2 will focus on machine learning-based models and combined solutions and subsection 2.3 contains conclusions on the articles considered.

2.1 Traditional motion planning algorithms

One of the main tasks of autonomous navigation in urban environments is effective and safe trajectory planning. Planning the movement of autonomous ground-based delivery robots in urban environments requires taking into account a multitude of factors: from the structure of sidewalks and pedestrian traffic intensity to the unpredictable behaviour of other participants in urban mobility. The most common classical methods include A*, Dijkstra, RRT algorithms, potential field-based methods, and their modifications. Below, I review and analyse key contemporary works in this field.

One of the modern modifications of the A* algorithm is Direction-Aware Self-Adaptive A* (DASA), presented by Zhang *et al*. (2025). It uses three key improvements: oriented selection of neighbouring points, self-adaptive search step, and a heuristic function trained on empirical urban navigation data. This model significantly reduces the number of nodes to be explored when constructing a route and reduces the overall planning time. However, for such methods to work effectively, a highly detailed map is required, which is not always available in real-world conditions. Furthermore, this model does not take into account the behaviour of dynamic objects (e.g., pedestrians or cyclists). The same limitations can be seen in hierarchical approaches, such as DHDB (Deformation Hierarchy of Directional Boundaries). This scheme implements a two-level approach consisting of a modified A* providing global planning (terrain, street architecture, etc.) and a local planner implemented through a deformation control method that allows dynamic obstacles to be quickly bypassed (Liu *et al*., 2024). Such drawbacks are generally typical of classical algorithms, including the Dijkstra algorithm. Like A*, it performs well in forming optimal routes, but only in conditions with static obstacles. In dynamic urban environments, however, this method shows poor adaptability and an inability to respond quickly to rapid changes in infrastructure (Yan & Yu, 2025). The authors also emphasise the need to hybridise the Dijkstra algorithm with reinforcement learning methods for greater flexibility and adaptability of the model.

An example of a hybrid scheme is given in the study by Niu *et al*. (2021). The authors propose combining the improved A* algorithm with artificial potential field methods, making this approach smoother and safer for motion planning in dynamic environments compared to the classic A*. However, in this work, experiments were conducted only in the MATLAB simulation environment, which does not take into account real-life scenarios of dense traffic and the social behaviour of surrounding pedestrians. For this reason, the practical applicability of this approach for autonomous delivery robots has not been confirmed.

Some classical approaches still have the potential to adapt to dynamic environments and unpredictable pedestrian behaviour. Such approaches include, for example, the Dynamic Window Approach (DWA), which is characterised by high global route accuracy with reactive modules. However, despite this, such methods still cannot predict the movement of other agents, which is their limitation in the context of navigation in a real city (Fox *et al*., 1997).

If information about the environment is incomplete or inaccurate, this is often solved by methods of potential field estimation and the introduction of risk analysis. Jahanshahi *et al*. (2018) in their work on this topic highlight the importance of introducing energy barriers and collision risk assessment to reduce the probability of a robot falling into local minimum traps. This approach ensures better stability of the constructed route, as the robot can more reliably avoid dangerous areas (e.g., crowded intersections). The method is effective in situations where only a partially known map of the terrain is available, but it requires high accuracy in the configuration of potential field parameters. When selecting them, it is important to bear in mind that parameters that are suitable for one environment may not perform well in a different one.

In a world where technology is developing at a rapid pace, the task of cooperative trajectory planning for multiple agents is no less relevant. This task is considered by Chen *et al*. (2021). The implementation was carried out using evolutionary methods, namely a combination of genetic algorithms (GA) with ant colony methods (ACO), which optimises the joint movement of several robots. By taking into account the surrounding environment, this approach allows for effective handling of urban infrastructure constraints and optimises load distribution. A limitation of this implementation is that as the number of agents increases and the environment becomes more complex, the computational complexity of the algorithm also increases, making it of limited applicability in tasks with rapidly changing parameters or in scenarios with a high degree of uncertainty.

2.2 ML based motion planning algorithms

Since this project deals with the movement of delivery robots in a dynamic environment where constant interaction between the robot and surrounding objects is necessary, classical navigation algorithms often prove to be insufficiently flexible. Machine learning and deep learning are now widely developed for predicting agent behaviour and adaptive movement (ADR). Such approaches allow modelling complex interactions between participants in traffic and, equally importantly, adapting to new situations.

With the introduction of machine learning into navigation systems, it became necessary to develop models that take into account social interactions between agents and the context of the environment. Some of the first works on the transition from classical LSTM models to generative models were described by Alahi *et al*. (2016) and Gupta *et al*. (2018). Alahi *et al*.

(2016) was the first work to propose a model called Social-LSTM, which implements the Social Pooling mechanism. This research laid the foundation for the development of more complex architectures, such as Social-GAN, which was first applied by Gupta *et al.* (2018). This transition made it possible to improve prediction accuracy by generating more realistic trajectories and taking into account social behaviour. However, both models were trained on ETH and UCY data, which only contained pedestrian coordinates, making them unsuitable for urban scenarios where physical infrastructure directly affects movement.

In response to this challenge, a more modern model was created – SoPhie (Sadeghian *et al.*, 2019), which implements two attention mechanisms: Social attention (interaction between agents) and Physical attention (scene map). This model was able to generate trajectories that took into account both the environment and the behaviour of surrounding agents, which is suitable for navigating delivery robots. A significant limitation is the high computational cost and the need for large and detailed data.

Another model with two attention mechanisms was implemented by Fernando (2017). This work combines 'soft' and 'hard' attention for the first time to take into account both local and global behavioural patterns of pedestrians, allowing for more detailed tracking of the influence of surrounding agents on the movement trajectory. The main idea is that pedestrian behaviour depends not only on their previous coordinates, but also on the behaviour of surrounding agents, whose influence is taken into account through an attention scheme. However, the work has some shortcomings: the model relies on a heuristic assignment of "hard" attention weights, which reduces the model's learnability and adaptability, and no maps with any obstacles were used in testing, so the model relies only on the behavioural patterns of other agents.

Adaptive probabilistic models, such as LSTM-MDL (Hug *et al.*, 2017), complement this class of methods by offering a different way of handling uncertainty and trajectory diversity. The authors emphasise that this model with a Mixture Density Layer allows for effective modelling of the probability distribution of pedestrian displacements. However, when testing the architecture, they found that it performs well in simple scenes, but accuracy drops in scenes with route forks. Hug *et al.* emphasise that familiar metrics (such as Final Displacement Error) may not adequately reflect the behaviour of the model and suggest using more comprehensive evaluation methods.

An adaptive method was also implemented in the work of Pang *et al.* (1999). The authors implemented a hybrid approach combining neural networks and fuzzy expert systems. This approach enabled dynamic route selection based on individual user preferences. A distinctive feature of the methodology is the system's ability to adapt and make decisions (predictions) based on previous human decisions. It can be useful when it is necessary to take into account the subjective preferences of the user and the context of the trip. While most models have focused on predicting coordinates, the hybrid fuzzy-neural strategy focuses on creating customised routes. The disadvantage of the strategy is its adaptability to new environments and resource efficiency, as with most current neural architectures such as RNN, GAN, and GCN. In a review by Huang *et al.* (2024), it was noted that such hybrid schemes allow the best results to be achieved.

2.3 Conclusion

Based on the analysis of the above-mentioned works, it can be concluded that despite the variety of studies on trajectory prediction and motion planning, existing approaches still have a number of limitations. Classic approaches described in Chapter 2.1 (A*, DWA, Dijkstra) demonstrate high efficiency in route planning, but are not adapted to the complex dynamics of the urban environment and do not take into account the movement of pedestrians and other road users. Neural network approaches described in 2.2, such as Social-LSTM and Social-GAN, effectively model social interactions but require huge amounts of data for training and show low accuracy when working with crowded scenes.

However, as mentioned in the study by Jiang *et al*. (2025), the accuracy and validity of navigation decisions can be improved by using mapping data containing moving objects and obstacles in research, but this is not always possible in real-world conditions. In addition, Yao *et al*. (2024) emphasise that even modern models that show high accuracy on standard datasets lose their stability when transferred between different scenes and datasets, which will be taken into account in this work.

All these limitations highlight the need for more flexible solutions. Thus, the present study aims to fill the identified gaps.

3 Research Methodology

3.1 Research Methodology Workflow Overview

This study aims to compare three approaches to planning the movement of autonomous delivery robots: the classic A* approach, the Social-LSTM machine learning model, and the hybrid A*+Social-LSTM approach. This work uses a secondary mixed empirical-deductive methodology to develop and evaluate motion planning algorithms in an urban environment, as visually represented in Figure 1.

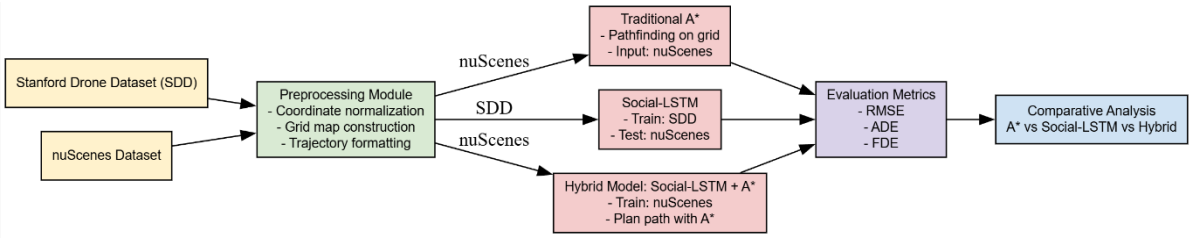

Figure 1: Research Architecture

Two open datasets were used as sources: the Stanford Drone Dataset (SDD), containing trajectories of pedestrians moving in crowded spaces such as a university campus or the sidewalks of a busy street, and nuScenes, which is a comprehensive set of sensor data (such as maps and objects) (Caesar *et al*., 2020).

The study begins with data preprocessing, which includes normalising coordinates, constructing a map in the form of a two-dimensional grid, and forming time sequences for training models. The data is then used in each of the three implemented architectures.

Model A* is implemented on one of the nuScenes maps, then Social-LSTM is trained on SDD data, after which it is tested on the same nuScenes map for transparency of comparison, as well

as to evaluate the model's ability to transfer knowledge, and in the hybrid architecture, Social-LSTM is trained on nuScenes and tested on the same map, respectively.
To evaluate the quality of the models, metrics such as root mean square error (RMSE), average displacement error (ADE), and final destination error (FDE) were used, which are often used in trajectory prediction tasks (Hug *et al.*, 2017).
At the end, a comparison of the results of the architectures is carried out, including visualisation of trajectories on the map and comparison of quantitative metrics. This allows us to determine which approach performs best for the task at hand.

3.2 Data Collection

Two open datasets were used in this study: the Stanford Drone dataset and the nuScenes dataset. Each of them significantly influenced the process of training and testing motion planning algorithms for autonomous delivery robots.
The Stanford Drone Dataset (SDD) contains aerial video recordings of eight scenes, mostly shot on a university campus, with annotations of the trajectories of pedestrians, cyclists, cars, and other agents.
nuScenes is a multimodal dataset providing data from lidars, cameras, radars, and GPS/IMU, as well as annotated road maps. It represents a more technically oriented and sensor-rich description of the urban environment.
Pre-processing included:

- coordinate normalisation and scene alignment,
- construction of fixed-length sequences for LSTM input,
- generation of a two-dimensional grid of urban infrastructure from nuScenes.

It should be noted that during the implementation of the Social-LSTM model, which was trained on SDD data and tested on the nuScenes map, difficulties arose due to significant differences in the structure, density, and format of the data. The data in the datasets differ in terms of viewing angles (SDD — static top view, nuScenes — perspective camera on a car), data format and density (SDD captures only visible agents in the frame, nuScenes contains sensor data for the entire scene), interaction context (SDD — mainly pedestrians on campus; nuScenes — traffic in a dense urban environment).
These differences affected the model's ability to generalise, as a result of which, in the hybrid approach, training and testing were carried out on the same dataset.
To ensure a fair and equitable comparison of the algorithms' performance, one of the scenes from the nuScenes dataset was used.

3.3 Tools and Resources Used

The development and implementation of architectures within this study was carried out in the Jupyter notebook environment, using a laptop based on AMD Ryzen 9 6900HS and AMD Radeon RX 6800S GPU, with 16 GB of RAM, running Windows 11. The Python 3.9 software ecosystem was also used to ensure the correct operation of all libraries used.
Several libraries and frameworks described in Table 1 were also used in this work.

Table 1: Libraries and frameworks used

Library/Framework	Purpose
PyTorch (torch, torch.nn, DataLoader, Dataset)	Model building and training for Social-LSTM, data batching
NumPy	Coordinate processing, vectorized operations
Pandas	Reading and analyzing trajectory tables
Matplotlib / matplotlib.animation	Trajectory visualization and agent animation
OpenCV (cv2)	Displaying scene and objects on images
tqdm	Progress display during iterations and processing
scipy.spatial.cKDTree	Nearest neighbor search for trajectory spatial data
heapq	Priority queue implementation for A* algorithm
collections.defaultdict	Organizing nested dictionaries and grouped data
random / os / glob / json	File system operations and JSON annotation parsing
NuScenes SDK	Access to nuScenes dataset: scenes, objects, annotations

Data and resources:

- **Stanford Drone Dataset (SDD)** — used to train the Social-LSTM model. The data was pre-saved in .pkl format.
- **nuScenes Dataset** — used to build a scene map, visualise agent movements, test the model, and implement the hybrid architecture. Data was downloaded via the official nuscenes-devkit.

3.4 Model Architectures

Three different approaches were implemented: (1) the classic A* pathfinding algorithm, (2) the Social-LSTM machine learning model, and (3) a hybrid approach combining both methods. During implementation, each of the presented architectures was tested on one of the maps from the nuScenes dataset (the same map was used for fairness of evaluation). This subsection provides a description of each architecture and the features of their use in the work.

3.4.1 A* Algorithm

A* (A-star) is a heuristic pathfinding algorithm widely used to find the optimal path, minimise time, distance and cost, and adapt to changing conditions, making it one of the most accessible solutions for planning (Rajgure *et al.*, 2025). A* is actively used in robotics, video games, and

autonomous navigation due to its efficiency, versatility, and ability to find the shortest path between any two points on a map. It is also considered an improvement over other classic algorithms, such as Dijkstra (Badamasi *et al.*, 2025).
In this implementation, the A* architecture is fed with a map from the nuScenes dataset, the start and finish points (goal), and a generated occupancy matrix as input data.
Algorithm implementation in this paper is based on the function:

$$f(n) = g(n) + h(n)$$

where:
$g(n)$ – the most economical route from start to goal;

$h(n)$ - heuristic estimation of the distance to the target (in this implementation — Euclidean distance);

$f(n)$ - priority rating of the node on which the queue operates.

In the code, priority is calculated as follows:

```
priority = new_cost + np.linalg.norm(np.array(neighbor) - goal)
heappush(open_set, (priority, new_cost, neighbor))
```

Figure 2: Priority calculation.

Here, new_cost represents $g(n)$, and np.linalg.norm(...) represents — $h(n)$. The final value of priority is $f(n)$, which determines the order in which nodes are visited in the queue. This structure ensures a balance between efficiency (thanks to heuristics) and accuracy (thanks to the consideration of the total path cost).
The output was a deterministic path from start to finish. The path was visualised using the matplotlib.animation and cv2 libraries. Finally, the final trajectory was superimposed on a map of the scene, where key points were also drawn.
This approach was implemented in the project first and served as a basis for comparison with other architectures. Since A* is unable to adapt and adjust to moving surrounding objects Zhang *et al*. (2025), this will allow for an objective assessment of the impact of ML-based models on motion planning.

3.4.2 Social-LSTM

To solve the path planning problem using a machine learning model, a recurrent neural network with a social interaction mechanism, Social-LSTM, was selected. It was first proposed and implemented in the work of Alahi *et al*. (2016). The reason for choosing this model is that it is capable of taking into account dependencies between objects (pedestrians) and paying attention to obstacles in the form of other road users.
In this approach, the model was trained on SDD data, which allowed the model to take into account interactions between agents. The Social-LSTM received pedestrian trajectories represented as coordinates, which were then normalised and fed into the LSTM encoder. The model extracts the agent's social ‘relationship’ to its surroundings (within a radius of 4 metres), which allows it to take social interactions into account. Thus, with an initial trajectory of 8

steps and taking into account neighbours within a radius of 4 metres, the model makes a prediction for the next 12 steps (coordinates) of the pedestrian in question.
After training, the model is tested on the nuScenes dataset map. During the experiment, it was found that due to significant differences in the structure and density of the datasets, the prediction accuracy is significantly reduced.
Thus, this approach, implemented on heterogeneous datasets, revealed some limitations when transferring between datasets.

3.4.3 Hybrid approach: Social-LSTM+A*

After implementing the classic A* algorithm and the Social-LSTM model, it became clear that they had limited effectiveness when used separately for the task at hand. The A* algorithm is unable to take into account social interactions between agents, while the Social-LSTM model does not take into account the map and obstacles on it. Therefore, it was decided to implement a hybrid approach combining both of these approaches.
The hybrid architecture in this project is implemented as follows: first, the Social-LSTM model receives data and learns from it, after which it is able to predict the movements of surrounding agents. These predictions are then transformed into a dynamic representation of the occupancy of intervals on the map. This creates a 'new' map where the trajectories of agents are obstacles, and these gaps are given a high 'cost' weight. Then, the A* algorithm is applied, which, as is already known, builds its trajectory taking into account static and dynamic obstacles.
Drawing conclusions from the previous approach, it was decided to use the nuScenes dataset to train the ML model in a hybrid architecture.
A similar architecture was considered in the work of Khaleq *et al*. (2022), where the A* algorithm was also used as a global path planner for the hybrid approach, and a cost map is constructed based on Social-LSTM predictions.

4 Design Specification

The design specification chapter describes the architectural basis of the ADR traffic planning system. This section focuses on the principles of system design as a whole.

4.1 Overall System Architecture

The work implemented classic A*, Social-LSTM, and hybrid (A*+ Social-LSTM) approaches. This structure made it possible to compare the results of each architecture separately and identify the characteristics of each.
The system components include:

- A* algorithm - a pathfinding algorithm that finds the shortest path between two points on a graph (in this case, between the start and end points on the map) using a heuristic function to estimate the distance between them. In this work, it serves as a basis for comparing other architectures. Its advantages are determinism, explainability, and stability when obstacles are known.

- Social-LSTM – a trainable model first proposed in the work of Alahi *et al.* (2016). It predicts the agent's trajectory based on its previous steps and also takes into account the trajectories of surrounding objects.
- Hybrid model – in this study, it combines the two approaches mentioned above. It uses Social-LSTM to generate a dynamic cost map (obstacles), which is then used to supplement the nuScenes map as it is traversed by the global A* planner. This allows static and dynamic conflicts to be taken into account when planning a route.

When implementing each of the architectures, the main requirements were:

- support for working with images (OpenCV),
- working with graphs and search algorithms (NumPy, heapq),
- training and testing machine learning models (PyTorch),
- processing and structuring trajectory data (json, pandas).

4.2 Design Rationale and Novelty

The most important component of the system is a hybrid module that combines the advantages of classical and learning approaches. Pedestrian coordinate predictions from LSTM are used to construct a dynamic obstacle mask that temporarily blocks the corresponding map cells for A* planning. This allows potentially dangerous areas to be taken into account when constructing a route and avoids collisions with moving pedestrians.

The idea behind this approach was based on experiments that revealed limitations such as the inability of the A* algorithm to adapt to dynamic objects and data incompatibility when testing the Social-LSTM algorithm. The novelty of this approach lies in the construction of a dynamic cost map based on ML model predictions, its integration into the A* algorithm, and the use of an HD map.

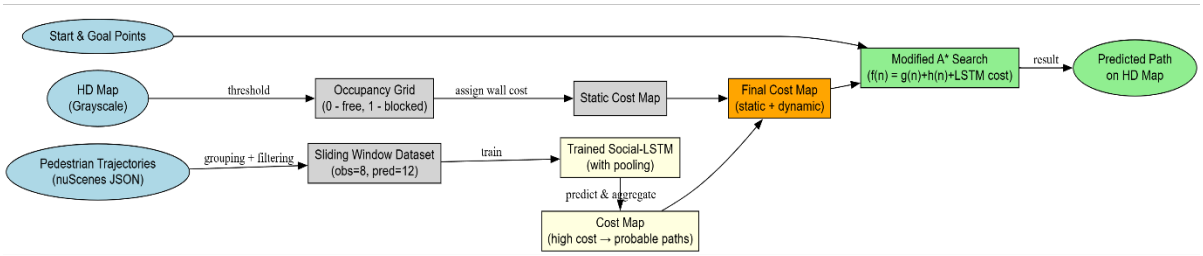

Figure 3: Hybrid approach architecture.

The hybrid approach provides a more realistic trajectory in dense urban environments and increases the system's adaptability to changing conditions. Interaction between modules is organised in such a way that real robot sensors can be connected or advanced simulations can be used in the future.

This approach is similar to that described in the previously mentioned work by Zghair and Al-Araji (2022), but it was not implemented there.

5 Implementation

As mentioned earlier, three independent architectures were implemented during the project: the classic A* path search algorithm, the Social-LSTM model, and a hybrid model (A*+Social-

LSTM). The architectures were implemented in a Jupyter notebook environment, using Python 3.9, with the use of basic libraries (such as PyTorch, OpenCV, NumPy, Matplotlib, json), as well as an environment for working with HD maps and annotations from the nuScenes-devkit dataset.

5.1 Data preparation and transformation stages

5.1.1 nuScenes dataset

The nuScenes v1.0-mini dataset, stored locally, was used to test the models. It was extracted from the official public website in a reduced version. Using the nuScenes API, annotations were filtered by category, from which pedestrian data was extracted.

Next, annotations with the use of nuScenes SDK tool were grouped by unique agent identifiers and coordinates (x, y) were formed from them. For subsequent procedures, only trajectories consisting of at least 20 steps were selected. These trajectories were distributed across scenes and saved in .json format, which were then divided into train/val/test samples in a ratio of 70/15/15.

For machine learning models, the trajectories were divided into 'sliding windows' of 20 steps (8 observed and 12 predicted).

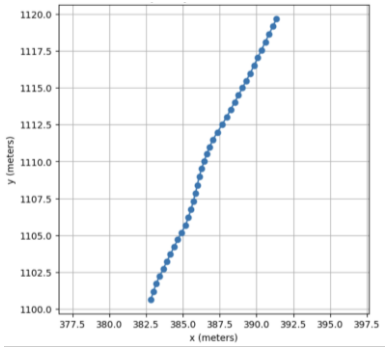

Figure 4: Pedestrian's trajectory.

For testing on this dataset, an HD .png scene map was selected and binarised using OpenCV: free points in the area were assigned a value of 0, and occupied points were assigned a value of 1. This formed the occupancy grid (Fig. 5).

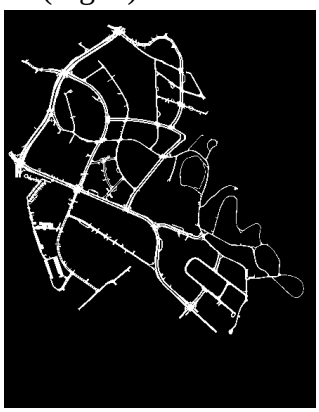

Figure 5: Map used for testing.

5.1.2 Stanford Drone Dataset (SDD)

The SDD dataset was downloaded from the public Kaggle website (Shah, A.,2021). SDD was used exclusively for training the Social-LSTM model. Since at some stage of the work, the incompatibility of the datasets became apparent, it was decided to bring the agent coordinates to a single metric system and normalise each trajectory by offset and scale. Pedestrian annotations were also extracted and filtered from it (Fig. 6). The model accepted data from this dataset in the same format as nuScenes: 20 steps, of which 8 were observed and 12 were predicted.

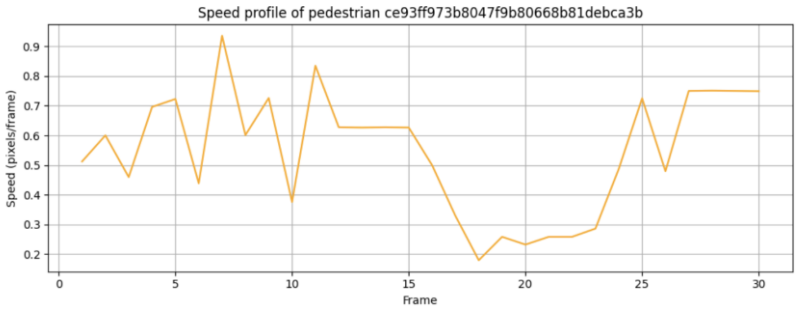

Figure 6: SDD pedestrian's speed profile.

5.2 Models implementation

5.2.1 Baseline models

Baseline models include the A* algorithm and Social-LSTM, as they reflect traditional and neural network approaches that are widely used in motion planning tasks.

The final result of the A* algorithm is visualised on a binarised HD map extracted from nuScenes. The algorithm calculated the shortest path from the start to the destination, taking into account all static obstacles. The Euclidean distance between the current coordinates and the finish line was used as the heuristic function. The result is presented as a list of coordinates, which is the route trajectory. The top left and bottom right passable points were selected as the start and finish points.

Regarding the Social-LSTM model, which was trained on SDD data and tested on the same nuScenes map (Fig. 5): the model took 8 trajectory points from the available data as input and predicted the next 12. After training, the model was saved in .pth format and the trajectory was also drawn on the map.

5.2.2 Experimental model

The experimental model used in this work is called a hybrid model because it has a novel element that is not used in baseline models.

The hybrid approach used a Social-LSTM model trained on nuScenes data to build a cost map. The model took into account surrounding agents and, based on predicted trajectories, transmitted information about potentially occupied sections of the route. This map was combined with the existing static occupancy grid, after which the A* algorithm built its route taking into account all obstacles. As a result, a map with an overlaid trajectory was also obtained.

6 Evaluation

The purpose of this section is to analyse the results obtained and determine their significance from a scientific and practical point of view.

The evaluation includes a comparison of the three algorithms presented in this paper: classic A*, Social-LSTM, and a hybrid based on A*+Social-LSTM. The traditional approach serves as a 'reference point' for evaluating the algorithms, as it is stable and deterministic.

Within the framework of this study, all testing and visualisation were performed on the same HD map extracted from the nuScenes dataset. In addition to the visual representation of the results, quantitative evaluation metrics such as ADE, FDE, and RMSE are also used. These metrics are calculated in the study according to the formulas:

$$ADE = \frac{1}{n \cdot T} \sum_{i=1}^{n} \sum_{t=1}^{T} \|\hat{y}_i^t - y_i^t\|,$$

$$FDE = \frac{1}{n} \sum_{i=1}^{n} \|\hat{y}_i^T - y_i^T\|,$$

$$RMSE = \sqrt{\frac{1}{n \cdot T} \sum_{i=1}^{n} \sum_{t=1}^{T} (\hat{y}_i^t - y_i^t)^2},$$

where:

$\hat{y}_i^t$- predicted coordinate at a point in time, y_i^t – actual coordinate, T – number of time steps, n – number of test trajectories.

The following subsections will analyse the final trajectories and metrics of each approach.

6.1 Experiment / Case Study 1

As part of the first experiment, the A* algorithm was implemented, which served as a 'starting point' for comparing the other approaches. A city map converted into a binary occupancy grid was fed into the system. The start and finish points (the first and last free cells on the map) were also specified. As a result, the points were calculated and had coordinates: (730, 5296) and (16479, 11538) for the start and end points, respectively.

As a result, a continuous trajectory (Fig. 7) connecting these two points was constructed. As part of the study, the algorithm used a heuristic (Euclidean) distance estimate and found the path with the minimum cost.

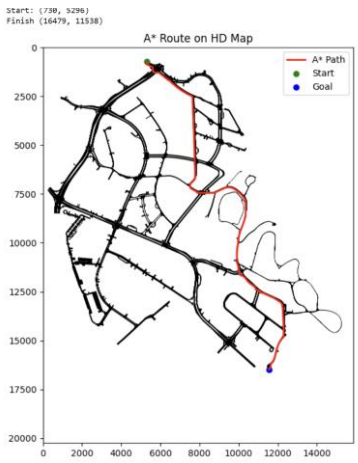

Figure 7: A* trajectory.

It should be noted that at this stage, metric evaluation and quantitative comparison of the path were not performed, since this stage was intended to test the correctness of the route and the map.

6.2 Experiment / Case Study 2

The next model applied in this study is Social-LSTM. It makes predictions based on the agent's past movements and also takes into account surrounding objects. The aim of the experiment is to evaluate and compare the quality of the predicted route, as well as the practical applicability of the model.

Social-LSTM was trained on a sample consisting of 7,220 scenes, using an early stopping mechanism (if the loss values decreased steadily, training was stopped). In this experiment, training ended at epoch 15, and the model with the best training results (train loss = 289.26, validation loss = 170.41) was saved.

```
Number of scenes in training set: 7220
Epoch 1, Train Loss: 333.4920, Val Loss: 356.7685
Best model saved
Epoch 2, Train Loss: 313.5219, Val Loss: 346.4421
Best model saved
Epoch 3, Train Loss: 308.4500, Val Loss: 350.1502
No improvement. Patience: 1/4
Epoch 4, Train Loss: 304.2430, Val Loss: 265.6046
Best model saved
Epoch 5, Train Loss: 301.2268, Val Loss: 195.0398
Best model saved
Epoch 6, Train Loss: 297.6746, Val Loss: 194.2364
Best model saved
Epoch 7, Train Loss: 296.3319, Val Loss: 205.9072
No improvement. Patience: 1/4
Epoch 8, Train Loss: 294.1049, Val Loss: 177.2170
Best model saved
Epoch 9, Train Loss: 293.3960, Val Loss: 193.6691
No improvement. Patience: 1/4
Epoch 10, Train Loss: 291.4394, Val Loss: 175.2436
Best model saved
Epoch 11, Train Loss: 289.2626, Val Loss: 170.4109
Best model saved
Epoch 12, Train Loss: 288.0109, Val Loss: 185.1967
No improvement. Patience: 1/4
Epoch 13, Train Loss: 287.7597, Val Loss: 199.8867
No improvement. Patience: 2/4
Epoch 14, Train Loss: 286.9013, Val Loss: 189.2778
No improvement. Patience: 3/4
Epoch 15, Train Loss: 285.5172, Val Loss: 171.3364
No improvement. Patience: 4/4
Early stopping triggered.
```

Figure 8: Social-LSTM training and validation loss (in pixels)

The image (Fig. 9) shows a visualisation of the predicted trajectory of a random pedestrian. The input data (blue dots), the actual trajectory (green dots) and the prediction itself (red dots) are also shown.

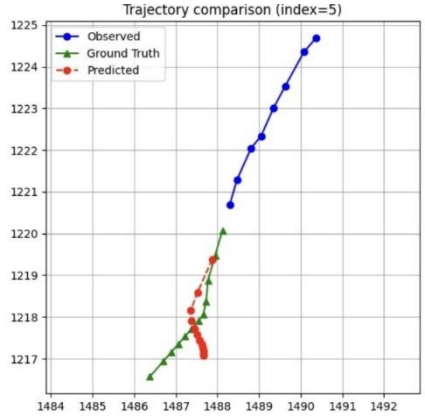

Figure 9: Social-LSTM predicted trajectory comparison.

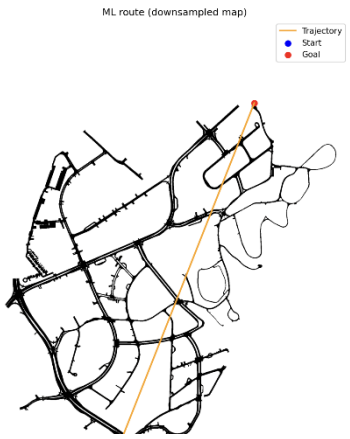

Figure 10: Social-LSTM route on HD-map.

Overall, the model performed well in terms of prediction, with the trajectory direction matching the ground truth, but there is a slight deviation to the right.
The model was also tested on the nuScenes dataset map.
The figure (Fig. 10) shows that the trajectory follows a straight line from start to finish, ignoring the obstacles depicted on the map. This once again highlights the model's inability to take into account the characteristics of the environment, since its goal is to predict the agent's behaviour.
To objectively evaluate the quality of the prediction, the following evaluation metrics were used:

Table 2: Social-LSTM evaluation metrics

Dataset	ADE	FDE	RMSE
Test loader	8.53	7.00	10.88
nuScenes loader	2.69	4.44	2.13

6.3 Experiment / Case Study 3

The last experiment aims to test the hypothesis that a hybrid approach (A*+Social-LSTM) will yield the best results, as it can combine the advantages of both architectures. Here, A* is used as a global planner, and Social-LSTM is used for local route prediction.
The model was trained and tested on the nuScenes dataset.
The training was conducted in 20 epochs, where the numerical error value was 0.358 in the first epoch and 0.049 in the last epoch, indicating good convergence and training quality of the model.
Due to the incompatibility of the SDD and nuScenes datasets, a single dataset was used for training in this experiment. The image (Fig. 11) shows the route of the Social-LSTM model trained on nuScenes. It can be seen that, as in the previous approach, the map shows a straight line from the starting point to the destination, without taking into account the road infrastructure.

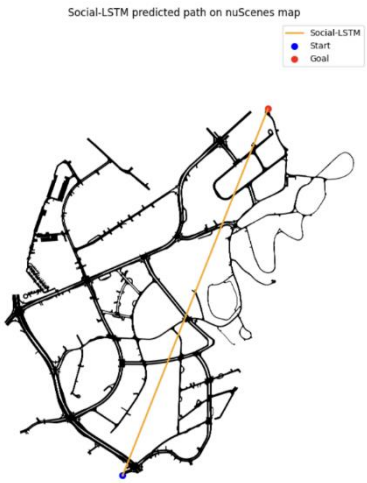

Figure 11: Social-LSTM trained on nuScenes predicted path.

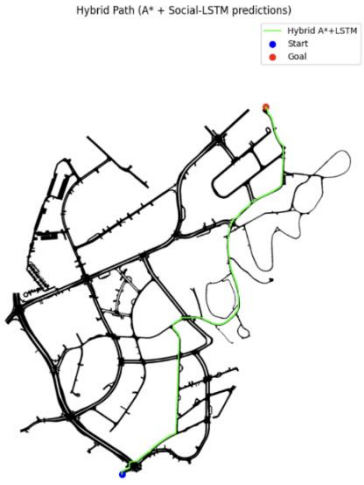

Figure 12: Hybrid model's trajectory.

The second map (Fig. 12) shows the result of visualising the hybrid model on the nuScenes map. This route combines global A* planning and local Social-LSTM motion generation. A* sort of 'corrected' the direction shown in Figure 11.

For this experiment, evaluation metrics were also calculated:

Table 3: Hybrid architecture's evaluation metrics.

Metric	Value (in pixels)
ADE	0.76
FDE	1.25
RMSE	0.75

Compared to the previous approach, the metric values in the hybrid model, trained and tested on the same dataset, have significantly improved.

6.4 Discussion

In Experiment 1, a trajectory from the starting point to the target was successfully constructed using the A* algorithm. It is worth noting that it looks realistic: it passes only through accessible areas, avoiding obstacles. The advantage of this approach is the logical and optimal nature of the resulting trajectory. However, as already described in Chapter 2.1 and noted in the works of Yang and Yoo (2025) and Fox *et al*. (1997), the limitation of traditional architectures such as A*, Dijkstra, and DWA is their inability to take social interactions into account. For this reason, the use of such algorithms in their classical form for delivery robots in the real world is also limited.

The Social-LSTM model described in Experiment 2 and trained on SDD data demonstrated a completely unnatural trajectory during testing: a straight line that did not take any obstacles into account. This is because the model in this experiment was trained without taking into account the scene or any objects; only the coordinates of the agents were fed into the input. Regarding the metrics, their error values were quite low, and the points on the graph comparing the actual and predicted trajectories were also predicted fairly accurately. A similar conclusion was described in the work of Hug *et al*. (2017). The paper notes that without taking into account the surrounding infrastructure, models are prone to errors.

The hybrid architecture, which is noted as innovative and combines A* and Social-LSTM, showed high accuracy in the evaluation metrics: ADE: 0.7677, FDE: 1.2561, RMSE: 0.7547, which is directly related to the training and testing samples (since they are taken from the same dataset). Another interesting observation is that the trajectory constructed by the hybrid architecture is identical to the trajectory of the classic A* algorithm. This is because A* is the global planner here, and Social-LSTM only corrects the trajectory with local offsets. In this experiment, due to the absence of neighbouring agents and the small amount of training data, the model has no significant reason to deviate significantly from the global path plan. Thus, case study 3 confirmed the theoretical basis described in Khaleq *et al*. (2022).

If we evaluate this work objectively, despite the positive aspects of individual experiments, this study can hardly be called applicable in the practical context of autonomous delivery robots navigation. The experiments conducted yielded some interesting and useful observations, but

no new insights were gained that could influence the further development of trajectory planning approaches.

The weaknesses of this work include the less than ideal choice of datasets due to their incompatibility. The model trained on the Stanford Drone Dataset proved to be poorly adapted to a different environment. Also, during implementation, before the final trajectory was obtained, there were many failed attempts where the trajectory went completely off the map or consisted of broken lines that did not make sense. In addition, the implemented hybrid model is not new, as the prototype was described in Khaleq *et al.*(2022). Furthermore, in this approach, A* works independently of Social-LSTM, which indicates that there is no integration of learning with global map information.

Nevertheless, one valuable aspect of the research is the rarity of the implementation of the described hybrid architecture. Although it was theoretically described in the article Intelligent Hybrid Path Planning Algorithms for Autonomous Mobile Robots, such an architecture had not previously been implemented taking into account real dynamic data. Also, given the limited variety of work on the topic of autonomous delivery robot motion planning, this study served to confirm the claims described in the above-mentioned article.

Possible improvements include using the same dataset for both training and testing (conducting two experiments on different datasets) to increase the stability and accuracy of the results; the hybrid architecture could also be improved by using attention mechanisms, as described in the work of Fernando (2017); implementing other neural network models (e.g., Social-GAN, Trajectron++ or Transformer-based) would make the evaluation more objective.

7 Conclusion and Future Work

Within the scope of this study, the following research questions were posed:

- Which architectures are most effective for improving the quality of motion planning of autonomous robotic couriers in smart city environments?
- Which architectures are most effective for improving the interaction of autonomous delivery robots with pedestrians and other road users?

To answer these questions, objectives were set and steps were taken to study existing research on motion planning and architectures actively used for this task; to implement the selected architectures (A*, ML-based Social-LSTM model, Hybrid (A*+Social-LSTM)); and to compare them using the nuScenes map and the values of the metrics obtained. Each model was tested to identify its advantages and limitations.

The study provided partial answers to both questions:

- For high-quality route planning for autonomous delivery robots, the hybrid model performed best, demonstrating a logical and realistic trajectory and high accuracy in estimation.
- In terms of ADR interaction with pedestrians and other road users, Social-LSTM took social interactions into account but did not demonstrate any significant advantages.

Future research directions could include the application of other architectures, the introduction of reinforcement learning in this field, and the use of data that is closer to urban conditions. Once stable, useful, and insightful systems have been developed, their practical application could also be considered: logistics companies, delivery services, robot engineers.

References

Abdul Khaleq, N., & Al-Araji, A. S. (2022). *Intelligent Hybrid Path Planning Algorithms for Autonomous Mobile Robots*. International Journal of Intelligent Engineering and Systems, 15(5), 309–325. https://doi.org/10.22266/ijies2022.1031.28

Agatz, N., Bouman, P. and Schmidt, M. (2018) 'Optimization approaches for the traveling salesman problem with drone', *Transportation Science*, 52(4), pp. 965-981. https://doi.org/10.1287/trsc.2017.0791

Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L. and Savarese, S. (2016) 'Social LSTM: Human trajectory prediction in crowded spaces', in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA, 27-30 June 2016, pp. 961–971. https://doi.org/10.1109/CVPR.2016.110

Badamasi, M. A., Kabir, I. K., Ahmed, G. and El-Ferik, S. (2025) 'Autonomous mobile robot path planning techniques: A review of classical and heuristic techniques', *IEEE Access*, 13, pp. 117999 – 118022. https://doi.org/10.1109/ACCESS.2025.3579863

Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G. and Beijbom, O. (2020) 'nuScenes: A multimodal dataset for autonomous driving', in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Seattle, WA, USA, 13-19 June 2020, pp. 11618–11628. https://doi.org/10.1109/CVPR42600.2020.01164

Chen, Y., Chen, M., Chen, Z., Cheng, L., Yang, Y. and Li, H. (2021) 'Delivery path planning of heterogeneous robot system under road network constraints', *Computers & Electrical Engineering*, 92, 107197. https://doi.org/10.1016/j.compeleceng.2021.107197

Fernandez, J. B. R. (2019) *Error metrics for trajectory prediction accuracy*. Available at: https://jaimefernandezdcu.wordpress.com/2019/02/07/error-metrics-for-trajectory-prediction-accuracy/ [Accessed 5 August 2025].

Fernando, T., Denman, S., Sridharan, S. and Fookes, C. (2017) 'Soft + hardwired attention: An LSTM framework for human trajectory prediction and abnormal event detection', *arXiv preprint*, arXiv:1702.05552. https://arxiv.org/abs/1702.05552

Fox, D., Burgard, W. and Thrun, S. (1997) 'The dynamic window approach to collision avoidance', *IEEE Robotics & Automation Magazine*, 4(1), pp. 23–33. https://doi.org/10.1109/100.580977

Gupta, A., Johnson, J., Fei-Fei, L., Savarese, S. and Alahi, A. (2018) 'Social GAN: Socially acceptable trajectories with generative adversarial networks', *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2255–2264. https://arxiv.org/abs/1803.10892

Hoffmann, T. and Prause, G. (2018) 'On the regulatory framework for last-mile delivery robots', *Machines*, 6(3), 33. https://doi.org/10.3390/machines6030033

Huang, S., Wei, R., Lian, L., Lo, S. and Lu, S. (2024) 'Review of the application of neural network approaches in pedestrian dynamics studies', *Heliyon*, 10, e30659. https://doi.org/10.1016/j.heliyon.2024.e30659

Hug, R., Becker, S., Hübner, W. and Arens, M. (2017) 'On the reliability of LSTM-MDL models for pedestrian trajectory prediction', in *International Workshop on Representations, Analysis and Recognition of Shape and Motion from Imaging Data*. Savoie, France, 17-20 December 2017, pp. 20-34. https://www.researchgate.net/publication/321965516

Jahanshahi, H., & Najafizadeh Sari, N. (2018) 'Robot Path Planning Algorithms: A Review of Theory and Experiment', *University of Tehran*. https://doi.org/10.48550/arXiv.1805.08137

Jiang, J., Yan, K., Xia, X. and Yang, B. (2025) 'A survey of deep learning-based pedestrian trajectory prediction: Challenges and solutions', *Sensors (Basel)*, 25(3), p. 957. https://doi.org/10.3390/s25030957

Liu, S. and Han, K. (2025) ‘Global-local 3D path planning method of inspection UAVs with DHDB analysis’, in *2025 5th International Conference on Computer, Control and Robotics (ICCCR)*, pp. 207–212. https://doi.org/10.1109/ICCCR65461.2025.11072655

Morris, B. T. and Trivedi, M. M. (2011) ‘Trajectory learning for activity understanding: Unsupervised, multilevel, and long-term adaptive approach’, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(11), pp. 2287–2301. https://doi.org/10.1109/TPAMI.2011.64

Niu, C., Li, A., Huang, X., Li, W. and Xu, C. (2021) ‘Research on global dynamic path planning method based on improved A∗ algorithm’, *Mathematical Problems in Engineering*, 2021(1), 4977041. https://doi.org/10.1155/2021/4977041

Pang, G. K. H., Takahashi, K., Yokota, T. and Takenaga, H. (1999) ‘Adaptive route selection for dynamic route guidance system based on fuzzy-neural approaches’, *IEEE Transactions on Vehicular Technology*, 48(6), pp. 2028–2041. https://doi.org/10.1109/25.806795

Patel, S. (2024) ‘Human interaction with autonomous delivery robots: Navigating the intersection of psychological acceptance and societal integration’, *in Tareq Ahram and Waldemar Karwowski (eds) Human Factors in Design, Engineering, and Computing. AHFE (2024) International Conference. AHFE Open Access*, vol. 159. AHFE International, USA. http://doi.org/10.54941/ahfe1005649

Rajgure, N., Koparde, S., Munot, B., Nalawade, T. and Keswad, T. (2024) ‘Survey on Top 10 Algorithms in Last Decade for Travel Planning’. *Unpublished manuscript*, Zeal College of Engineering and Research, Pune, India.

Sadeghian, A., Kosaraju, V., Sadeghian, A., Hirose, N., Rezatofighi, S. H. and Savarese, S. (2019) ‘SoPhie: An attentive GAN for predicting paths compliant to social and physical constraints’, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1349–1358. https://arxiv.org/abs/1806.01482

Shah, A. (2021) *Stanford Drone Dataset*. Available at: https://www.kaggle.com/datasets/aryashah2k/stanford-drone-dataset [Accessed 5 August 2025].

Yang, Q. and Yoo, S.-J. (2025) ‘Optimizing intersection navigation with multi-shot update mechanism: A real-time distributed reinforcement learning for ground and aerial moving objects’, *IEEE Internet of Things Journal*, in press. https://doi.org/10.1109/JIOT.2025.3589567

Yao, P., Zhu, Y., Bi, H., Mao, T. and Wang, Z. (2024) ‘TrajCLIP: Pedestrian trajectory prediction method using contrastive learning and idempotent networks’, *Proceedings of the Thirty-eighth Annual Conference on Neural Information Processing Systems (NeurIPS 2024)*. Available at: https://openreview.net/forum?id=fUBFy8tb3z

Zhang, X., Tan, L. and Chai, J. (2025) ‘DASA: a direction-aware and self-adaptive A* algorithm with learned heuristic for UAV path planning of smart city’, *Research Square*, preprint. https://doi.org/10.21203/rs.3.rs-6725799/v1