Rideshare Pilots
Assisted Matching via Planning and Acting in Simulated Environments

Addison Hanrattie
Project link

University of Maryland
ahanratt@umd.edu

Submission Checklist

Done	Item
X	Example completed task.
	Prepare Your Code
X	Push your code to a private GitHub repository. (If your results are not too large, please include your results too.)
X	Be sure your code has all planning models you used in your project.
X	Write a README.md that describes where things are in the repository.
X	Invite makro@umd.edu and dhchan@cs.umd.edu as collaborators to the repository. (I just made it public)
	Fill the content (recommended order)
__	Add your results and evidence
__	Describe your evaluation plan
__	Complete the general discussion of Seciton 5.1
__	Fill in the Approach section
__	Fill in the background as needed to explain results and approach
__	Write the Introduction
	Complete the self assessment (after 12/2)
__	Copy the `grading-template.tex` from Piazza into your project.
__	Complete the self assessment.
__	Sign the pledge.
	Submit your report (after 12/2)
__	Create a PDF of this document
__	Submit the PDF to Gradescope, being sure to select the first page for the first question.
__	Email your PDF to makro@umd.edu.

Assessment

Your report will be distinguished by the following criteria:

- Approach:
  - Clearly states the technical approach of the work in a self-contained way.
  - The acting environment is clearly explained; stronger reports will include an example figure. If the environment started as a Gym environment but was modified, this is clear.
  - Includes a planning and acting component. For the acting, simulation is fine
  - Stronger reports will leverage or manipulate the integration in an interesting way.
- Evaluation:
  - Clear exposition of claims, questions, variables and protocol.
  - Evidence to support the claims in the form of a table or plot and no table or plot has font smaller than \scriptsize (about 6pt font).
  - Plots have axes that are clearly labeled and their captions state their intended meaning.
  - All plots and tables in Section 7 are referenced in the main portion of the paper; supplemental plots that are not referenced should be placed in the appendix.
  - A discussion of findings and how evidence relates to the claims
  - Stronger reports will have a baseline approach; while not required, there is already evidence of this in some reports.
  - It is certainly not required for this report, but you are welcome to include tests of statistical significance if this is something you know how to do.
  - The strongest reports will demonstrate what I call second-order thinking. This is where you conjecture a possible reason for the results you saw. It is even stronger if you run one or more experiments to verify this conjecture or provide analytical results showing why this is the case.
- Scope and Writing
  - Paper tells an interesting or noteworthy story rather than just a chronology of experiments. (Exhibits first-order thinking)
  - Written content is 2-4 pages, excluding floats (figures and tables), and floats are correctly placed in Section 7 to make this assessment easy!
  - Writing is easy to understand
  - Reader is not left ”wondering”. Consider yourself a teacher of your project. What would your ‘student’ need to know to understand the content?
  - Grammar is mostly sound; no obvious typos, misspelled words, sentence fragments, etc.
  - Citations are correct and consistent; URLs are fine for hyperlinks, but, generally, books and articles should use a bibtex entry.

Student Assessment: Please complete the orange boxes, replacing text within <.. >. For example, in the first entry you would replace "<replace:{ 1, 2, 3 } >" with "1" if you did a type 1 project. I suggest you change these one at a time and recomplile each time to make sure each change is correct.

Description	(your answer)
My project type was	1
My report (Sections 1-6) is this many pages long (for partial pages, use 0.3, 0.5, 0.7):	4.5
My planning system was (add rows if you had more than one)	<replace: planning system >
My acting system was (add rows if you had more than one)	<replace: acting system >
I invested approximately the following time, in hours, for each sprint: (this is for the instructor to assess relative difficulty)	
Sprint 1 : Development	<replace: integer [0,C] >
Sprint 1 : Report Writing	<replace: integer [0,C] >
Sprint 2 : Development	<replace: integer [0,C] >
Sprint 2 : Report Writing	<replace: integer [0,C] >
Sprint 3 : Development	<replace: integer [0,C] >
Sprint 3 : Report Writing	<replace: integer [0,C] >
I answered _____ research questions in my main report	<replace: integer [0,C] >
I included _____ plots in my main report	<replace: integer [0,C] >
I wrote _____ lines of code (excluding comments) for this project	<replace: integer [0,C] >
(If included, these were optional) I added an additional _____ plots in my appendix	<replace: integer [0,C] >
In terms of difficulty compared to other semester projects I have done, I would rate this project as (1-5 scale with 1 being easiest 5 being most difficult)	<replace: { 1, 2, 3, 4, 5 } >
In terms of what I learned, I would rate this (1-5 scale with 1 being "a little" and 5 being "a lot")	<replace: { 1, 2, 3, 4, 5 } >

Name: REPLACE THIS TEXT BY TYPING YOUR NAME HERE

(For this report, typing your name here will suffice)

I pledge on my honor that I have not given or received any unauthorized assistance on my programming project or report.

Instructor Assessment (Mak will fill this in)

Approach	Points Available	Earned Instructor Only
Clear technical approach	5	
Planning Environment (or other system) clearly explained	5	
Acting Environment (or other system) clearly explained	5	
Includes Planning and Acting Component	5	
Evaluation	**Points Available**	**Earned**
Clear exposition of claims, questions, variables and protocol	10	
Source code and overall development work	10	
Included baseline approach or ablation study, where appropriate (or demonstrated second order thinking/evaluation)	[5]	
Evidence to support claims is discussed and included in Section 7	10	
Discussion of results	10	
Writing	**Points Available**	**Earned**
Clear story arc, paper within scope for project (2-4 pages of writing)	10	
Followed checklist for placing plots and tables and overall structure; all plots are referenced	5	
Grammar and overall writing is sound and citations are proper	5	
Overall	**Points Available**	**Earned**
Technical Difficulty (related to project type)	10	
Technical gains considering difficulty	10	

1 Introduction

Within the rideshare domain, a very common pain point is connecting with the rider with their matched driver. This is especially true in scenarios where there are likely to be a very large number of riders searching for rides, such as at a concert or sporting event. In these scenarios, it is common for riders to be matched with drivers that are not in their immediate vicinity, and thus the rider must navigate to the driver. This can be a difficult task, especially if the rider is unfamiliar with the area or if there are obstacles in the way.

While much attention in AI planning has been focused on achieving the best possible planning performance and plan quality little attention has been given to leveraging the ability to simulate people as actors in the environment. In this project, we simulate a rideshare user attempting to find their assigned car in a parking lot. Rather than identifying the best possible plan to find the car, we investigate how different environments effect the ability of an agent to find their car when acting in the environment with a simple planning and acting algorithm. This allows us to ultimately draw conclusions about how different environment characteristics effect the ability of an agent to successfully find their car with the hope being that these insights can be used to design better human rideshare experiences in the future. As rideshare services move towards greater autonomy and uniformity in car design, it is important to maximize the efficiency of the particularly human intensive tasks.

2 Background

The motivation for this project builds on concepts introduced in earlier work that highlighted a gap in research on the final 100 yards of on-foot navigation within the broader last-mile problem (Samet and Hanrattie 2024). We acknowledge that the difficulties faced by riders in locating their car once in a general vicinity (ie parking lot) are especially unique as compared to simply walking to said area from an original starting location. Prior work has explored the challenges of on-foot navigation in urban environments, but there is a lack of research specifically addressing the nuances of the search task that is rideshare pickup.

The issue of locating a rideshare vehicle goes far beyond simple time efficiency savings. There have been numerous cases of riders entering the wrong vehicle and being physically harmed such as the case of Samantha Josephson who was murdered after entering a car she thought was her Uber. This case and others motivated the creation of Sami's law (Smith 2023) which requires a study to be done on the number of assaulted riders each year and the safety measures taken. In general there is very little regulation around rideshare safety within the US as compared to countries like China (Stemler, Evans, and Shu Shang 2025) which regulate where and how rideshare pickups can occur. Therefore the lessons learned from this project should not only be used to improve efficiency but also safety of rideshare pickups ensuring that riders can quickly and accurately and safely find their rideshare vehicle.

The main planner for this project is inspired by *HTN-Run-Lookahead*, (Ghallab, Nau, and Traverso 2025), Algorithm 6.3. Briefly, this acting algorithm checks whether the agent has reached the goal. If an existing plan exists and *Simulate* is not failure, it performs the next action in the plan. Otherwise it creates a new plan. It also uses some elements of the repairing scheme from (Ghallab, Nau, and Traverso 2025), Algorithm 6.4.

3 Approach

The primary methodological challenge in this project was modeling the inherently human element of search and decision-making in the final 100 yards of ride-hailing pickup. Two distinct human factors were addressed: (i) the behavioral characteristics of pedestrian movement [one cannot just walk across a busy street] and (ii) imperfect or incomplete knowledge about the locations of vehicles due to limited perception range.

To reduce problem complexity while preserving the core decision structure, agent movement was discretized to a uniform grid. Although this abstraction does not fully reflect the continuous motion of real life, it allowed the planning model to operate on a tractable state space and supported reproducible experimentation. In addition, the agent was assumed to perceive vehicles in all directions but could only verify the identity of a vehicle only when physically located near it or near a assistant which could direct the agent. This mirrors real-world conditions in which riders may be able to see a vehicle at a distance but cannot confirm that it is their assigned car until they reach it unless otherwise advised. This assumption also constituted a controlled simplification of the visual uncertainty present in practice.

3.1 Environment

The interactive environment was implemented as a grid-world simulator in python using Gymnasium. The environment was hand crafted to allow for easy modification and to ensure a faithful representation of the rideshare pickup task. The environment consist of a grid representing a parking lot with obstacles, a single agent representing the rider, a single target vehicle representing the assigned rideshare car, numerous distractor vehicles representing other cars in the lot, and numerous assistants representing people in the lot who can help the agent find their car. The agent can move in four cardinal directions, observe its surroundings, and gain insights from nearby assistants. The goal of the agent is to find and reach the target vehicle in as few steps as possible. To prevent the planner from exploiting privileged information, all vehicles are rendered indistinguishable up to a certain viewing distance even if obstacles could be seen further beyond.

When resetting the environment a obstacle generator may be supplied. This generator is responsible for placing obstacles in the environment according to some distribution. The default generator used would take a count of obstacles and their maz size and then uniformly choose random locations to seed the obstacles and then uniformly grow them along the perimeter until they reached their maximum size or were otherwise constrained. It can also be noted that all of the actors are also placed uniformly randomly across the grid.

The perception of the agent is limited in the following manor. After any given step a $N \times N$ square centered on the agent is extracted and then any point which in non navigable within the area is labelled out of sight. Finally if any in-sight vehicles are outside of a manhattan distance of M from the agent they are labelled as unknown vehicles. This process ensures that the agent can only see a limited area around itself and cannot identify (but can observe) vehicles at a distance. However if an assistant is within the viewing distance of the agent then the assistant can inform the agent of the identity of any in-sight vehicles.

3.2 Planning Approach

At the heart of our approach is a simple Hierarchical Task Network (HTN) planner with a few common sense methods for finding the target. The planner is implemented in standard python without the support of any planning libraries. While not directly named the planner has two methods which can be used to form plans: `navigate_to_point`, `explore_environment`. These refinement methods use the base actions of `perceive` and `move` to form themselves. The `perceive` action allows the agent to look around its current position and identify any vehicles, obstacles, or assistants within its viewing distance. The `move` action allows the agent to move one step in any of the four cardinal directions, provided there is no obstacle in the way. The `navigate_to_point` method allows the agent to create a plan to move to a specific point in the environment using A* search (it is actually bi-astar since that runs even faster). The `explore_environment` method allows the agent to create a plan to explore the environment in a systematic way, visiting all points within its viewing distance. Specifically this is done by recursively calling `navigate_to_point` on the nearest unexplored point until the target has been located. Once the target has been located, the `move_to_point` method allows the agent to create a plan to move directly to the target vehicle and begin its ride.

3.3 Acting Approach

Since the environment is dynamic the content that a given space is associated with may change over time. For example, a car may begin as an unknown space but once perceived it may be identified to be a distractor vehicle. If said distractor vehicle is in the way of the agent's current plan to reach the target vehicle then the plan must be repaired. To handle this dynamic environment we focus on using online methods that interleave the planning and acting. The main acting algorithm is inspired by *HTN-Run-Lookahead*. Specifically, at each time step before acting we pop the next move action from the current plan and simulate it. If the simulation is successful (ie the space is safe to move into) then it is executed in the environment. If the simulation fails then a new plan is created from the current state which focuses on identifying a new route to the point of focus. This process continues until the agent reaches the target vehicle.

4 Evaluation Plan

Within the simplified setup described above, venue owners largely have two controls. Firstly they can control the density and prevalence and pattern of obstacles within the parking lot. Secondly they can control the number and distribution of assistants within the parking lot. Therefore we focus our evaluation on how these two independent variables effect the ability of the agent to find their car. As well as how other meta-variables such as the size of the parking lot and the viewing distance of the agent effect performance. This extra analysis is important for better understanding how the environment operates and how different components interact and influence one another. It is important to note here that the distance at which a car can be identified may also be influenced by factors which are within a venues control such as clearly labelling sections of a garage however those improvements and the variable as it is reflected here is not a direct relationship.

Independent variables For our first experiment we investigate the effects of obstacle density and size on the ability of the agent to find their car. We start with a grid of size 18×18, view sizes of 9×9, no obscurity, and no fake targets. We vary the following independent variables:

- **Obstacle occurrence rate:** This variable controls the maximum density of obstacles within the parking lot. We vary this from 1 obstacle to 10 obstacles.
- **Obstacle size:** This variable controls the maximum size of obstacles within the parking lot. We vary this from small obstacles consisting of 2 blocked spaces up to a size of 20 blocked spaces.

For each of the following experiments we run experiments on each of the following parking lot sizes: a small lot of size 12×12, a medium lot of size 24×24, and a large lot of size 36×36. Additionally for each board size we run the experiment on each of the following combinations of obstacle distributions. This is done to ensure we control that any of the following tested independent variables effects are not due to a preference on the selected obstacle distribution. We ensure across all of the settings that the same number of points are blocked to ensure a fair comparison. The obstacle distributions used are as follows:

- **Many Dots:** Numerous small obstacles scattered randomly throughout the parking lot.
- **Normal:** A moderate number of medium sized obstacles scattered throughout the parking lot.
- **Few Big:** A small number of large obstacles scattered throughout the parking lot.

The first two experiments we run focus on tuning choices of the environment specifically the view size of the agent N and the distance at which a car can be identified M. For the first experiment the number of fake targets is kept constant at 5 and a car can be identified at any distance. For the second experiment the number of fake targets is kept constant at 5 again and the viewing distance is kept constant at 11×11. For both experiments we vary the independent variable as follows: rolling the view distance from 3 to 11 in increments

of 2 and rolling the identify distance from 1 to 11 in increments of 1.

The final experiment we run focuses on the effects of assistant count on the ability of the agent to find their car. For this experiment we keep the viewing distance at 11×11, the number of fake targets at 5, and the identify distance at 3. We vary the independent variable as follows: rolling the assistant count from 0 to 7 in increments of 1.

Dependent variables The main dependent variable of interest across all experiments is the average number of steps taken by the agent to find their car. This metric directly reflects the efficiency of the agent in completing the task. Furthermore, it is a reasonable proxy for rider searching / walking time in real-world scenarios, which is a critical factor in user satisfaction with rideshare services. By minimizing the number of steps taken to locate the vehicle, we can infer that the agent is effectively navigating the environment and leveraging available resources (like assistants) to optimize its search strategy.

5 Results

Local View Size The first of the two meta-variable tuning experiments focused on the effects of viewing distance on the ability of the agent to find their car. The results of this experiment can be seen in Figure 1 and Figure 2. As expected increasing the viewing distance of the agent has a significant positive effect on the ability of the agent to find their car. This is almost definitely due to the fact that with a larger viewing distance the agent can see more of the environment at any given time and thus can make more informed decisions about where to move next. It is also interesting to note that the effect of viewing distance is more pronounced in larger parking lots in terms of the total number of steps saved but worse in terms of percentage improvement. This is likely due to the fact that in larger parking lots there is more area to cover and thus the agent can benefit more from being able to see more of the environment at any given time. However, since the total number of steps taken is also larger in larger parking lots, the percentage improvement is smaller.

The relationship observed was clearly linear and the relationship can be seen in Table 1. For the small, medium, and large parking lots the cost savings per radius increase was found to be 9.30, 29.51, and 50.27 respectively when averaged across the three obstacle distributions. It is interesting to note that the savings seem as though they may scale linearly with the size of the parking lot as well.

Identification Distance The second of the two meta-variable tuning experiments focused on the effects of identification distance on the ability of the agent to find their car. The results of this experiment can be seen in Figure 3 and Figure 4. As expected increasing the identification distance of the agent has a significant positive effect on the ability of the agent to find their car. It seems clear this is due to the fact that with a larger identification distance the agent can more easily identify their car from a distance and thus can make more informed decisions about where to move next and better sort between negative examples and positive examples. Again we see that the effect of identification distance is more pronounced in larger parking lots in terms of the total number of steps saved but worse in terms of percentage improvement. This is likely due to the same reasons as mentioned above.

The relationship observed appears to be a exponential decay to a asymptote which is dependent on the board size. It can also be noted that the decay rate is much shallower for the larger parking lots. This would seem to indicate that identification distance is especially important in smaller parking lots but that its importance diminishes in larger parking lots where ability to effectively plan more forward thinking routes is more important.

Obstacle Count and Size The first of our two main experiments focuses on the effects of obstacle count and size on the ability of the agent to find their car. The results of this experiment can be seen in Figure 5. Interestingly here we see that obstacle count and size has a positive effect on the ability of the agent to find their car. This is interesting as one would expect that the more obstacles there are in the environment the harder it would be for the agent to find their car. However, it seems that the obstacles actually help to structure the environment and provide more information to the agent about where to search. This is likely due to the fact that with more obstacles the agent can better segment the environment into smaller areas and thus can make more informed decisions about where to move next. It is also possible that with more obstacles there are more dead ends and thus the agent can more easily eliminate areas of the environment from consideration. This is supported by the fact that the obstacle count seems to have a greater effect than obstacle size which would make sense as a greater count leads to more segmentations of the environment.

Assistant Count The second of our two main experiments focuses on the effects of assistant count on the ability of the agent to find their car. The results of this experiment can be seen in Figure 6 and Figure 7. As expected increasing the number of assistants in the environment has a significant positive effect on the ability of the agent to find their car. Much like with the car identification distance this is likely due to the fact that with more assistants the agent can more easily identify their car from a distance and thus can make more informed decisions about where to move next and better sort between negative examples and positive examples.

In the case of the smaller parking lot nearly all of the benefit is gained by the first assistant with very little gained thereafter. This would seem to suggest that in smaller parking lots a single assistant is sufficient to rapidly speed up the rate at which the agent can find their car. However, in the case of the medium and large parking lots we see that while the first assistant again provides the largest boost in performance there is still significant benefit to be gained from additional assistants (with the large parking lot having the longest tail). Thus it seems that in larger parking lots having multiple assistants is needed to maximally speed up the rate at which the agent can find their car.

Summary In summary, our results indicate that several environmental factors significantly influence the efficiency of an agent in locating its assigned vehicle in a rideshare pickup scenario. Our first finding was that providing structure to the environment in the form of obstacles can actually aid the agent in its search task, likely by segmenting the environment and reducing the effective search space. Secondly, enhancing the agent's perceptual capabilities, such as by having clearer labels for car space numbers (increasing identification distance), substantially improves search efficiency, particularly in smaller environments. Lastly, the presence of even just a single human assistant which can pilot riders to their vehicles dramatically reduces search time.

5.1 Discussion of Tradeoffs and Limitations

The particular difficulty here is using planning and acting methods to gain an understanding of how best to structure real world environments for human riders no matter how effective (or ineffectively) they can plan. Thus the biggest limitation here is the fact that we are using a simplified gridworld environment to model a complex real world task. While this simplification allows us to more easily control and manipulate the environment it also means that our results may not fully generalize to real world scenarios. Additionally, our relatively simple planner does not incorporate more advanced techniques such as learning or probabilistic reasoning which may be more effective in real world scenarios. However this is not necessarily a bad thing as the methods described may be exactly how humans approach the problem of finding their car in a parking lot. Nevertheless, future work could focus on increasing the number of different planning methods utilized to ensure that the results are maximized for all possible planning methods just as one would want to maximize for all possible human behaviors. Following this line of thought it would also be interesting to incorporate more realistic human behavior models into the environment to better simulate how humans would actually behave in these scenarios. The first of which could be as simple as adding some noise to the agent's movement or perception as well as more complex models such as causing agents to forget previous observations over time (allow them to get lost). Another way to generate new planning methods which doesn't involve hard coding elements could be to use a reinforcement learning approach based on a LSTM neural network to learn a policy for finding the car in the environment as this could better simulate how humans learn to navigate environments over time. Another could be to use the method learning techniques as described in (Ghallab, Nau, and Traverso 2025), Chapter 7 to more naturally learn new methods and then sanity check them against human behavior.

Other limitations come in the form of how the environment mimics reality. The generation follows uniform seeding whereas in the real world there is far more structure to these parking lots. Thus future work could focus on generating more realistic parking lot environments to better simulate real world scenarios. This could include generating parking lots with rows and aisles as well as more dynamic obstacles such as crosswalks. Furthermore the agent views the world in a top down sense whereas in reality humans have a first person view of the world. Thus future work could focus on implementing a first person view akin to that of the flatlands to better simulate real world scenarios.

Finally, as with all projects more data analysis and experiments could be run to better understand the effects of the independent variables on the dependent variable. This could include running more trials to ensure statistical significance as well as running more experiments to better understand the interactions between the independent variables such as increasing board sizes beyond 36×36 or varying the number of fake targets in the environment to be even greater. Future work could also take a more nuanced work on the dependent variable such as setting benchmarks and seeing how many trials fall within a certain threshold of the benchmark or also measuring the variance of the results to better understand the consistency of the agent's performance.

6 What I learned

- How to implement a Gymnasium environment from scratch and test it effectively.

  How to debug said environment when it inevitably breaks lots of times.
- How to write strongly typed python code using type hints and how to keep consist dependency management using uv (This code is heavily productionized and other than the simplifications I would feel confident sending it to venue owners and letting them add their venues in as maps).
- How to optimize my code using numpy so that it runs really fast (we can initialize 2000 games per second and take over 15,000 steps per second on my laptop).
- How to implement a simple HTN planner and acting algorithm in python.
- How to design and run experiments on planning and acting systems which investigate the effects of environmental factors on planning and acting performance.

7 Summary Figures and Tables

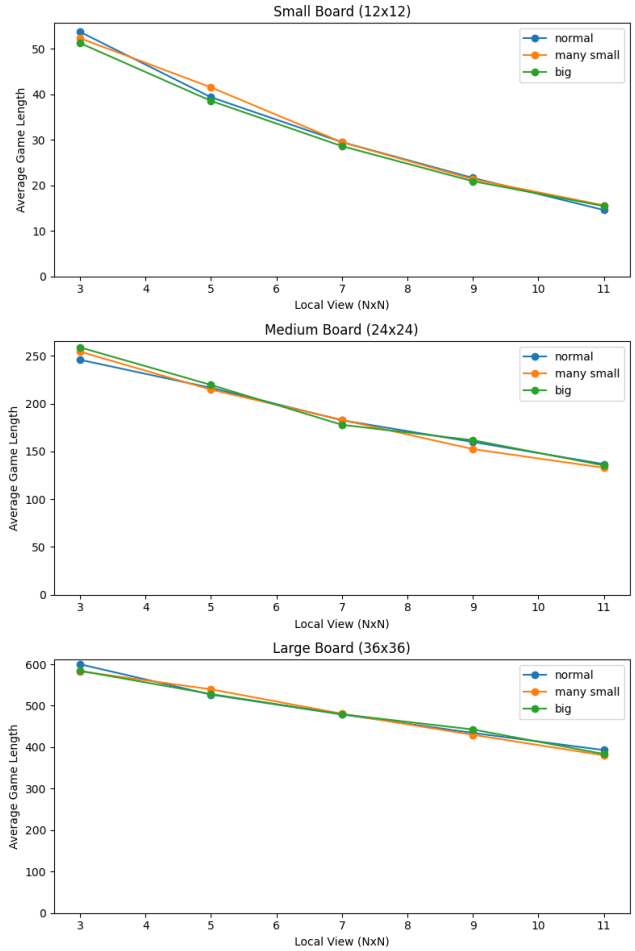

Figure 1: Effect of viewing distance on the ability of the agent to find their car across all trials.

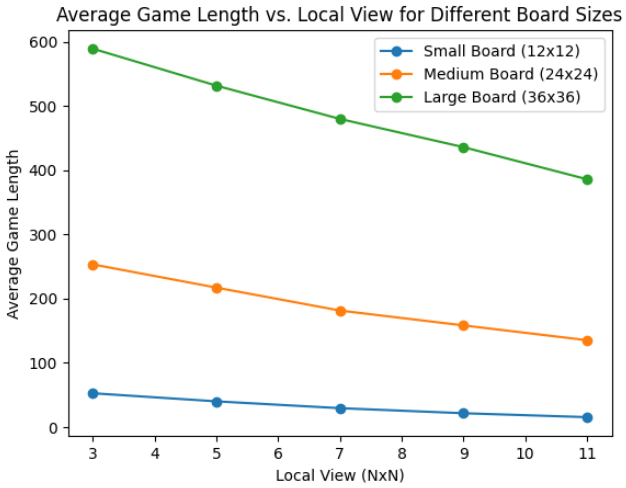

Figure 2: Effect of viewing distance on the ability of the agent to find their car averaged across obstacle distributions.

Board Size / Scheme	Slope	Intercept	R^2
Small Normal	-4.80	65.38	0.98
Small Many Small	-4.68	64.85	0.98
Small Big	-4.46	62.23	0.98
Medium Normal	-13.75	284.72	0.99
Medium Many Small	-15.26	294.45	0.99
Medium Big	-15.25	297.56	0.97
Large Normal	-25.32	664.00	0.99
Large Many Small	-25.76	663.14	1.00
Large Big	-24.33	654.07	1.00

Table 1: Linear fit parameters for game length vs. diameter across board sizes and obstacle schemes.

References

Ghallab, M.; Nau, D.; and Traverso, P. 2025. *Acting, Planning, and Learning*. Cambridge: Cambridge University Press. ISBN 978-1-009-57938-4.

Samet, H.; and Hanrattie, A. 2024. PILOT: Piloting the Last 100 Yards. In *Proceedings of the 32nd ACM International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '24, 625–628. New York, NY, USA: Association for Computing Machinery. ISBN 979-8-4007-1107-7.

Smith, C. 2023. Sami's Law.

Stemler, A.; Evans, J. W.; and Shu Shang, C. 2025. Data Privacy and the Regulation of Ridesharing Platforms. *American Business Law Journal*, 62(2): 117–139.

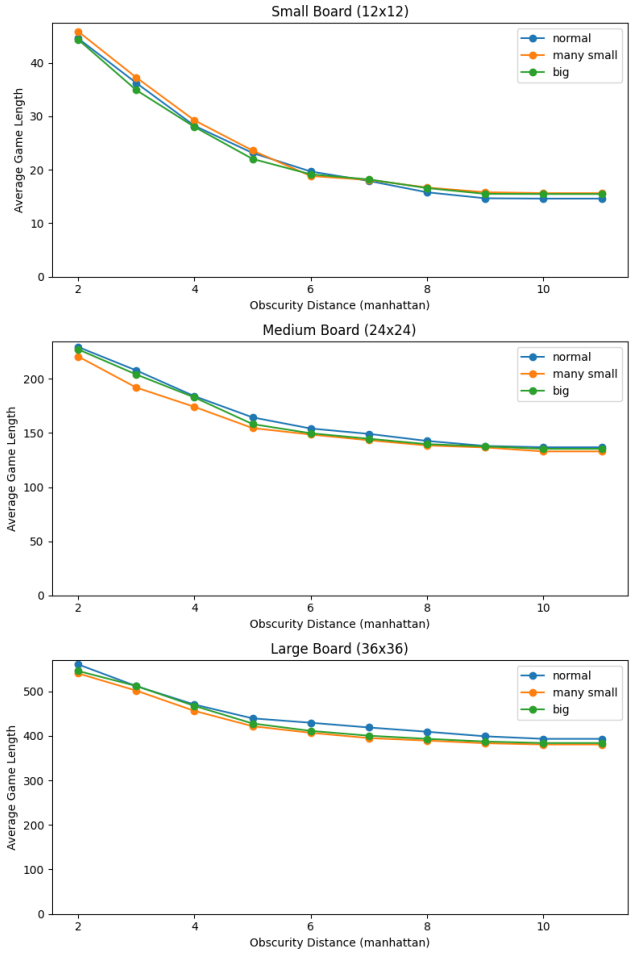

Figure 3: Effect of identification distance on the ability of the agent to find their car across all trials.

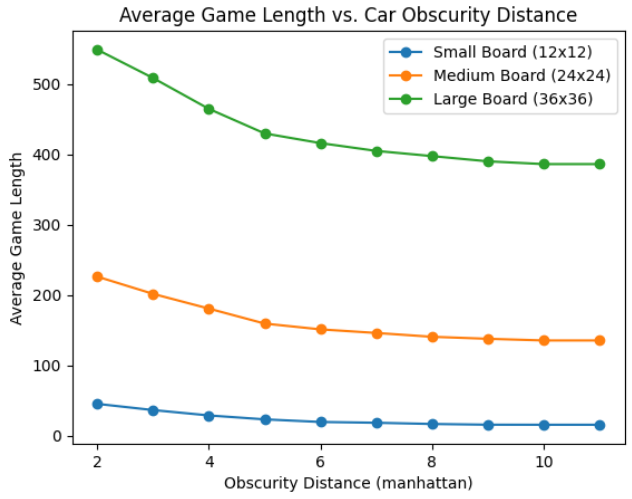

Figure 4: Effect of identification distance on the ability of the agent to find their car averaged across obstacle distributions.

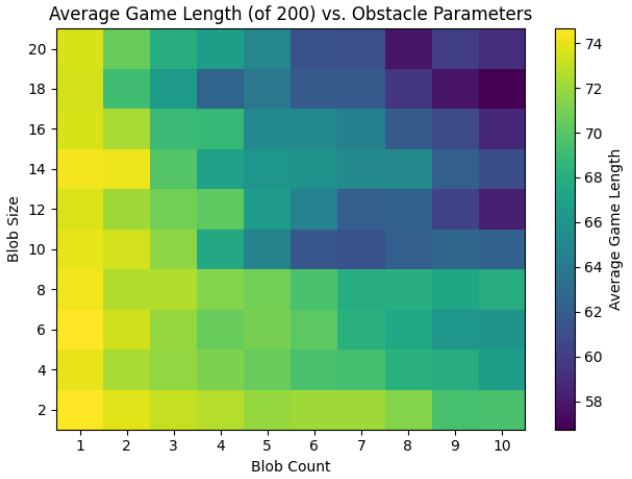

Figure 5: Effect of obstacle count and size on the ability of the agent to find their car averaged across viewing distances.

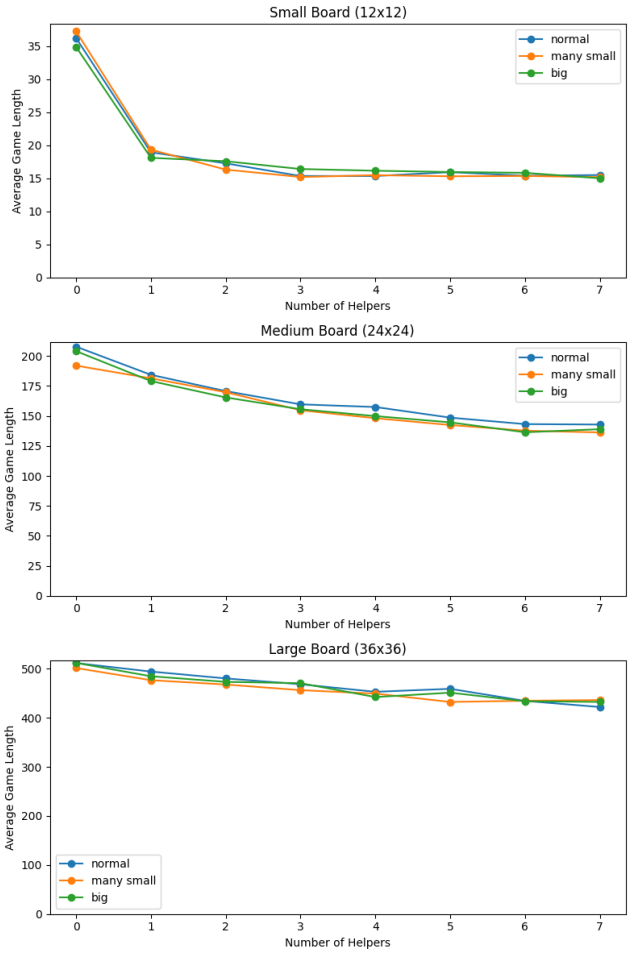

Figure 6: Effect of assistant count on the ability of the agent to find their car across all trials.

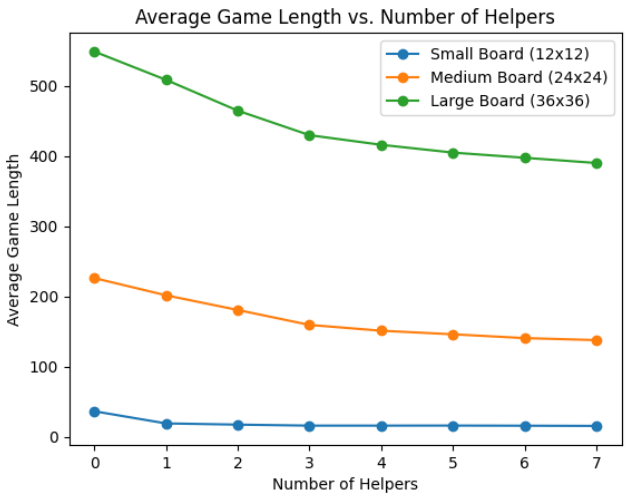

Figure 7: Effect of assistant count on the ability of the agent to find their car averaged across obstacle distributions.