THE UNIVERSITY OF ADELAIDE

DOCTORAL THESIS

Towards Optimistic, Imaginative, and Harmonious Reinforcement Learning in Single-Agent and Multi-Agent Environments

Mohammad Mahdi Kazemi Moghaddam

School of Computer Science

Supervisors

Prof. Javen Qinfeng Shi
Dr Qi Wu

A thesis submitted in the fulfilment of the requirements for the degree of Doctor of Philosophy

Faculty of Sciences, Engineering and Technology

September 2022

Contents

List of Figures

List of Tables

I Declaration

I certify that this work contains no material which has been accepted for the award of any other degree or diploma in my name, in any university or other tertiary institution and, to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made in the text. In addition, I certify that no part of this work will, in the future, be used in a submission in my name, for any other degree or diploma in any university or other tertiary institution without the prior approval of the University of Adelaide and where applicable, any partner institution responsible for the joint award of this degree. The author acknowledges that copyright of published works contained within the thesis resides with the copyright holder(s) of those works. I give permission for the digital version of my thesis to be made available on the web, via the University's digital research repository, the Library Search and also through web search engines, unless permission has been granted by the University to restrict access for a period of time.

Signature: Mohammad Mahdi Kazemi Moghaddam
Date: 2022/11/28

II Preface

I received my Bachelor's degree in Electronics Engineering from the Amirkabir University of Technology, in Iran. It was during the last year of my undergraduate studies that I started to become aware of my strong passion for Artificial Intelligence (AI) through advanced programming courses. Participating in, and organising, international robotics competitions, during my undergraduate studies provided me with the opportunity to start to develop foundational computer science skills. Since then I planned a smooth transition to the field of AI and while I continued to work on designing software systems for financial institutions for about five years, as my next step. As those years passed, I became continuously more confident that research in AI was what I was extremely passionate about.

I started my AI research journey as an Honours student at the University of Adelaide (UoA), in July 2018. I graduated with first-class Honours as the Valedictorian of my cohort of graduates, among eight schools in my faculty, in July 2019. Having taken a solid first step in my AI research journey, I was excited to accept a full scholarship offer from UoA to start my PhD immediately. I was so much enjoying what I was doing in my research that I considered a break between my two degrees would not be the best use of my time. During the past three years since then, I have been immersing myself in everything related to AI; from learning how to successfully publish in top conferences, to winning various AI and robotics competitions, and pursuing excellence in developing AI industry solutions. This has made me not only grateful for all the opportunities that I was provided, but also proud of my successful career transition. This thesis covers only my research outcome during the past three years on the topic of real-world reinforcement learning. I leave writing about everything else that I learned along the way for the appropriate time and space.

The more I learned about AI the more I grew enthusiasm for building agents that can learn to make decisions just like humans. Learning such skills is, arguably, one of the most impressive aspects of humans that AI agents are still far from reliable. As such, this thesis reflects my passion for developing AI agents capable of learning to make sequential decisions in the real world through their interactions with the environment. As the primary existing solution to such problems, RL research has gained increasing momentum. Real-world problems, such as navigation and autonomous driving, pose unique challenges to existing RL method which requires dedicated attention. Inspired by that, this thesis provides original contributions to tackle a selection of those challenges. The contributions start from single-agent problems, where the other agents are considered as part of the environment, and end with a more complex multi-agent method.

Reflecting upon my challenging yet fruitful journey, I hope to inspire researchers who wish to work in a similar area or who share a similar starting point with me. I have been asked multiple times along the way, "whether leaving a job for doing research in a different field is worth it", and I will continue to respond a bold "yes".

Mohammad Mahdi Kazemi Moghaddam
September 2022

III Acknowledgements

Even though I have been the sole writer of this thesis, I would not have been able to do so if it was not for the support and contribution of many people around me. First and foremost, I would like to dedicate this work to my lovely wife whose support was the main motivation for me to go through this journey. Her kind support and unconditional love always encourage me to work harder and aim higher. I feel blessed every moment for having her kind, generous, and pure soul in my life. I would, also, like to dedicate this work to my parents, parents-in-law, sisters, and my single brother-in-law. I am wholeheartedly grateful for all their sacrifices and their unconditional love and support. My family's sacrifices provided me with the opportunity to start this journey. My wife's family raised such a lovely daughter and supported both of us in every possible way. My deepest gratitude, respect, and admiration for all of them will never end.

My PhD, similar to many others, was a journey full of gaining new knowledge and learning new skills, many of which will even help me in my daily living. Beyond knowledge and skills, I learned how to learn, how to trust my instinct in the face of uncertainty, and how to never give up when the challenges seem bigger than my ability, at least at the first glance. Among all the people who contributed to this fruitful journey, my deepest gratitude goes to my primary supervisor and mentor, Javen, who is very close to my heart for his unique personality. He held my hand and taught me how to be an excellent researcher and a better human being. His door was always open, and his phone was always handy, for my in and out-of-time messages, requests, and questions. His endless support and trust were always motivating during challenging times. I will never forget his kindness and generosity. Next, I would like to send my best gratitude to my first co-supervisor, Qi, for all his help and support throughout my journey. He would always carefully listen to my naive ideas and teach me how to make them better. He taught me how to think big and be an honest and transparent researcher no matter what. I would, also, like to thank my second co-supervisor and friend, Ehsan, for his continuous support. I will always remember his friendly mentoring discussions, whether in his office or at his warm house. Those three gentlemen shaped who I am today as a researcher.

There were more kind-hearted people around me who contributed significantly to my success. Among those, I would like to first send my most sincere gratitude to Ian, our head of school. His fatherly support and mentorship made me more courageous in the face of challenges. He has my best admiration and appreciation. I would, also, like to thank Yasir, my mentor and friend, who always treated me like his younger brother, and unconditionally provided me with his wealth of knowledge and experience. I would, also, like to thank Markus, our post-graduate coordinator, for being keenly open to help whenever needed. Finally, I wish to thank Prof. Michael Liebelt, the Dean of Graduate Studies, for working hard to make all students feel supported, including myself.

During the last few months of my PhD, I had the privilege to work with, and learn from, Stefano, and his amazing research group members, most importantly Arrasy, at the University of Edinburgh. They made me feel very welcome in their warm and united group. I learned how to more effectively participate in group discussions from them. I am grateful for their contribution and the opportunity they provided me.

The list of my friends, colleagues, and collaborators who supported me in different ways and contributed to my success is longer than I could fit here; Tom, Sam, Georgia, Ravi, Michael, Greg, James, Matt, Anthony, Dong, Rachel, Kathy, and Anton from AIML; Ali and Alex from the school of Civil, Environmental and Mining Engineering; Hao and Peter from AMI-Fusion; Hamid from Monash University; Ross, Sam, Alois, Lewis, and Sue from AIS and Paralympics Australia; those are among the names that I will never forget, no matter where on earth I physically am. I genuinely feel blessed for having them all in my life, and I cherish all the great memories we have made together.

Finally, I would like to acknowledge the hard work and contribution of our school, our AIML institute, and our university's administrative and executive team to the success of all the members, including myself. My PhD was supported by the Adelaide Graduate Research Scholarship (AGRS) for which I am extremely grateful.

IV Publications

Some of the contents of this thesis is based on the following conference publications:

- Kazemi Moghaddam, Mahdi, Ehsan Abbasnejad, Qi Wu, Javen Qinfeng Shi, and Anton Van Den Hengel. "ForeSI: Success-Aware Visual Navigation Agent." In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 691-700. 2022. This work is discussed in chapter 4.
- Kazemi Moghaddam, Mahdi, Qi Wu, Ehsan Abbasnejad, and Javen Shi. "Optimistic agent: Accurate graph-based value estimation for more successful visual navigation." In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 3733-3742. 2021. This work is discussed in chapter 5.

In Addition, the following journal publication has been excluded from this thesis due to different topics:

- Ghiasi, Alireza, Mahdi Kazemi Moghaddam, Ching-Tai Ng, Abdul Hamid Sheikh, and Javen Qinfeng Shi. "Damage classification of in-service steel railway bridges using a novel vibration-based convolutional neural network." *Engineering Structures* 264 (2022): 114474.

Finally, the following work is in its final stages at the time of submitting this thesis:

- Kazemi Moghaddam, Mahdi, Arrasy Rahman, Stefano V. Albrecht, Javen Qinfeng Shi. "NePO: Self-Interested Agents Empower Neighbours.", *Work in Progress*, 2022. This work is discussed in chapter 6.

V Abstract

Reinforcement Learning (RL) has recently gained tremendous attention from the research community. Different algorithms have been proposed to tackle a variety of single-agent and multi-agent problems. The fast pace of growth has primarily been driven by the availability of several simplistic toy simulation environments, such as Atari and DeepMind Control Suite. The capability of most of those algorithms to solve complex problems in partially-observable real-world 3D environments, such as visual navigation and autonomous driving, however, remains limited. In real-world problems, the evaluation environment is often unseen during the training which imposes further challenges. Developing robust and efficient RL algorithms for real-world problems that can generalise to unseen environments remains an open problem.

One such limitation of RL algorithms is their lack of ability to remain optimistic in the face of tasks that require longer trajectories to complete. That lack of optimism in agents trained using previous RL methods often leads to a lower evaluated success rate. For instance, such an agent gives up on finding an object only after a few steps of searching for it while a longer search is likely to be successful. We hypothesise that such a lack of optimism is manifested in the agent's underestimation of the expected future reward, i.e. the *state-value function*. To alleviate the issue we propose to enhance the agent's state-value function approximator with more global information. In visual navigation, we do so by learning the spatio-temporal relationship between objects present in the environment.

Another limitation of previously introduced RL algorithms is their lack of explicit modelling of the outcome of an action before committing to it, i.e. lack of *imagination*. Model-based RL algorithms have recently been successful in alleviating such limitations in simple toy environments. Building an accurate model of the environment dynamics in 3D visually complex scenes, however, remains infeasible. Therefore, in our second contribution, we hypothesise that a simpler dynamics model that only imagines the (sub-)goal state can achieve the best of both worlds; it avoids complicated modelling of the future per timestep while still alleviating the shortcomings resulting from the lack of imagination.

Finally, in our third contribution, we take a step forward beyond single-agent problems to

learn multi-agent interactions. In many real-world problems, e.g. autonomous driving, an agent needs to learn to interact with other potentially learning agents while maximising its own individual reward. Such selfish reward optimisation by every agent often leads to aggressive behaviour. We hypothesise that introducing an intrinsic reward for each agent that encourages caring for neighbours can alleviate this problem. As such, we introduce a new optimisation objective that uses information theory to promote less selfish behaviour across the population of the agents.

Overall, our three contributions address three main limitations of single-agent and multi-agent RL algorithms for solving real-world problems. Through empirical studies, we validate our three hypotheses and show our proposed methods outperform previous state-of-the-art.

Chapter 1

Introduction

One of the ultimate goals of research in Artificial Intelligence (AI) is to develop intelligent agents capable of learning to perform real-world sequential decision-making tasks in a robust and efficient way. The reinforcement Learning (RL) paradigm is arguably the most promising area of research to enable AI agents with such capability. In this chapter, we introduce our three research questions: *optimism*, *imagination*, and *harmony* in single-agent and multi-agent RL. We introduce our proposed methods and discuss how our empirical results validate our hypotheses.

Recently, a large number of RL algorithms have been developed on simple toy environments [11, 88, 102]. While those environments enable rapid bench-marking of new methods, they cannot fully represent the challenges of using RL to address real-world problems. For instance, real-world RL problems may require an agent to learn not to capitulate as long as a task is not successfully completed, i.e. to be **optimistic**. Moreover, completing tasks in visually complex environments requires an RL agent to have an explicit representation of a successful goal state, i.e. to **imagine** what that state may look like before reaching it. Such imagination enables an agent to constantly compare the new observations to that imagination of the goal state which increases its robustness. Finally, when the environment includes other potentially learning agents, an RL agent is required to avoid selfish behaviour and instead maintain the overall performance **harmony** across all the present agents. Those real-world challenges are often missing from simple simulation environments.

Inspired by such real-world RL challenges, in this thesis we propose multiple novel methods to tackle two of the most general and challenging real-world RL problems, i.e. **visual navigation** [125] and **autonomous driving** [21]. In visual navigation, an agent is placed in a complex 3D environment, e.g. a house or an office, and is tasked with finding a target object by navigating towards it and stopping within its predefined proximity. In autonomous driving, the agent is required to drive to a given destination, i.e. directly controlling the steering and acceleration, in a robust, safe, and efficient way, while interacting with other agents.

In the following sections, we introduce three different RL methods we are proposing in this thesis. Each section provides more method-specific background information that motivates the proposed approach. The purpose of this section is to highlight the research question every method is trying to answer and the significance of the proposed solution.

1.1 Part I: Single-Agent Reinforcement Learning and Visual Navigation

In Chapters 4 and 5 of this thesis, we discuss two separate methods to address specific limitations of previous state-of-the-art RL algorithms when used for the visual navigation problem. Our proposed methods are evaluated on a commonly used benchmark, e.g. AI2Thor [52]. Further discussion on how the simulation environment works and background problem-specific information is provided in 2.1.

1.1.1 Problem 1: Optimistic Reinforcement Learning

Actor-critic [67] RL algorithms have proven to be effective in learning navigation policies in different simulation environments and tasks [22, 23, 71, 115]. The role of the critic in such algorithms is to stabilise the policy learning by providing better feedback compared to the original policy gradient methods [87]. As such, learning to approximate the critic (i.e. the state-value function) accurately is essential to help with the convergence of the policy to a more optimal solution.

Most of the previous state-of-the-art navigation frameworks, however, share the same architecture and parameters between the actor and the critic neural networks (except for the last layer to map the representations to action and state-value) [22, 115]. This, as we show in our empirical studies, often leads to pessimistic value estimates hence leading to a sub-optimally converged policy, which often fails to successfully complete the navigation task. In order to address that limitation, we hypothesise that the critic should be able to represent more global information about the surrounding 3D environment.

In the visual navigation task [4], specifically object-goal navigation, the spatio-temporal relationship between the observed objects in the scene is key to finding the target object. Imagine observing the world from the limited Field-of-View (FoV) (e.g. 90 degrees) viewpoints equal to what an agent can observe at each navigation time-step. In that case, observing a kitchen stove at one time step might be a good indication that a kettle may be observed on either side of it in future time steps. Learning such object relationships in space and time provides a strong learning signal to the policy. As we show in our experiments, this signal guides the policy to learn to avoid early capitulation when the prospect of finding the target object in the future time steps is highly likely.

Recently, Graph Neural Networks (GNNs) have proven to be effective in learning effective representation where the relationship between entities matters [17, 120, 124]. We use a variant of GNN, e.g. [120] to allow the critic to build a more global representation of the scene and the observed objects during a navigation episode. Using GNN provides the critic with a more holistic view of the scene observed through limited FoV images. We utilise such a rich representation to reduce the approximation error for the state-value function. This leads to *optimistic behaviour*, as shown in our empirical results, by avoiding underestimation of the state value at critical time steps during a navigation episode. Further details of this method and the achieved experimental results are provided in Chapter 4.

1.1.2 Problem 2: Imaginative Reinforcement Learning

The ability to predict the future consequences of an action, i.e. *imagination*, is, arguably, one of the most challenging capabilities to endow artificial agents with. It is our imagination, as humans, that helps us make more robust and efficient decisions by avoiding actions that may not lead to our intended goal. Inspired by such human capacity, many different Model-Based RL (MBRL) algorithms have recently been developed [33, 36, 44, 45]. MBRL methods, generally, alternate between learning a forward model and planning using such learned dynamics models. As such, by explicitly modelling the outcome of possible actions at each time step, those approaches have shown to outperform model-free counterparts in simple environments [34, 36]. Despite significant progress, current state-of-the-art MBRL algorithms are limited by two fundamental challenges: learning an accurate forward model that can predict long into the future (i.e. until the end of an episode), and developing a planning algorithm which is robust to accumulative forward model prediction errors. Therefore, their application has been mainly limited to simple environments [91, 102], where the training and evaluation environments are the same.

In visual navigation, unlike conventional RL benchmarks, the environment is 3D and visually realistic [52, 89]. More importantly, the agent is evaluated in new previously unobserved scenes. Therefore, using existing MBRL algorithms which rely on accurate per-step forward predictions becomes infeasible. To address those issues, we instead augment model-free RL algorithms with a forward model that allows the agent to predict the latent future state, i.e. to *imagine*. To further reduce the complexity of the forward modelling, we train our model to only imagine successful trajectory outcomes. That way, our method enables agents to imagine only what a successful future looks like in the latent space. Such simplification balances the best of both worlds; it enables the agent to predict desirable (sub-)goal states while avoiding inaccurate per-step future predictions. Our method improves the efficiency and robustness of visual navigation without the need for costly per-step prediction or planning algorithms which are robust to inaccurate future predictions.

1.2 Part II: Multi-Agent Reinforcement Learning and Autonomous Driving

In the second part of this thesis, we move from single-agent RL algorithms to the more complicated problem of multi-agent RL (MARL). Inspired by the great success of RL (especially deep RL) in single-agent environments, MARL provides further opportunities to learn the policies of multiple agents sharing the same environment. A Myriad of real-world problems, including autonomous driving, are inherently multi-agent; that is, multiple agents concurrently learn new policies or adapt theirs to that of other interacting agents. Similar to part I of this thesis, we are interested in tackling a real-world problem using MARL. We choose to work at the intersection of MARL and autonomous driving for a range of reasons, including:

- There has been minimal work in this relatively new area of research which provides an opportunity for a great impact [80].
- Multi-agent driving can be cast as a mixed-motive MARL problem [63] which is both highly challenging, especially as the number of agents increases, and interesting.
- Last but not least, extending visual navigation to a multi-agent problem adds limited value. Multi-agent navigation can be sensibly defined as a cooperative MARL problem [108]. Owing to the availability of benchmarking environments [88], cooperative MARL has recently received significant attention [61, 77, 82, 97, 107]. Autonomous driving, on the other hand, despite sharing similar challenges as visual navigation, presents more complexities as it involves a combination of cooperative and competitive multi-agent interactions [94].

1.2.1 Harmonious Agents

Many real-world multi-agent problems involve self-interested agents interacting with others aiming to maximise their own individual reward function, i.e. mixed-motive problems [30, 63]. In such problems, independent reward optimisation leads to greedy selfish policies, often showing more aggressive behaviour [54, 80, 118]. This problem is further exacerbated if simultaneous access to limited indivisible resources is desirable by multiple agents [43].

Autonomous driving is a good example of such a problem where agents are self-interested (i.e. try to reach their destination in the shortest possible time) while sharing limited resources (e.g. road capacity) with other road users. Humans can learn to efficiently interact with others while adhering to both rules and social norms. An example of such behaviour is highlighted when part of a road is blocked and drivers take turns to pass, or while merging into a highway where drivers on the highway let merging vehicles do so smoothly. Autonomous vehicles, however, are still limited in learning those behaviours and instead choose to become more conservative to ensure safety.

We propose to alleviate the effect of selfish individual reward maximisation, among autonomous driving agents, by introducing a novel multi-agent intrinsic reward. Our intrinsic reward function is inspired by empowerment [50]. Intuitively, the agents learn to balance between maximising their individual reward and *empowering* other agents using our new optimisation objective. Empowerment is only calculated within every agent's neighbourhood. This allows our method to scale to a large number of agents. By reducing selfish behaviour through empowerment, we increase the overall harmony among the population of agents, measures based on their expected return.

1.3 Original Contributions and Thesis Outline

In summary, in this thesis, we propose multiple novel single-agent and multi-agent RL methods, for real-world complex problems, e.g. visual navigation and autonomous driving. Each one of our proposed methods is focused on addressing a specifically investigated limitation of the previous state-of-the-art methods when applied to real-world problems. We evaluate our methods on commonly used benchmarks for visual navigation and autonomous driving and show that they outperform baselines and previous state-of-the-art methods. Through empirical studies, we show the validity of our main hypotheses and provide further insights into the limitations of the proposed methods. We categorise our main original contributions into the following three main categories:

- **Optimistic RL:** In Chapter 4, we show that one of the main problems with the current state-of-the-art RL methods is the lack of optimism in behaviour. Agents trained using baseline methods fail to continue in longer trajectories and often capitulate early. We define optimism as the expected error between the learned state-value function, in actor-critic RL, and the ground-truth state values; value approximations using our method have both lower expected error and are generally higher in value, hence the *optimism*. We propose to train separate policy and value function approximators to alleviate the issue. Moreover, we endow our value approximation network with more global state observation, using a variation of GNN [120], to reduce the value approximation error. Our empirical studies show the efficacy of our method, which outperforms several state-of-the-art methods in a commonly used visual navigation benchmark, AI2Thor [52].
- **Imaginative RL:** In Chapter 5, take our first step towards enabling *imagination* in visual navigation agents based on model-free RL. While there has been some work on model-based RL (MBRL) [34], most of the previous state-of-the-art RL-based visual navigation frameworks are model-free. Model-free RL is attractive in terms of simplicity, ease of implementation and training, as it trains a policy that directly maps observations to actions. Such a policy does not explicitly consider the potential outcome of different actions, hence is limited in generalisation to complex environments, e.g. visual navigation. While can potentially address such limitations of model-free RL, developing MBRL algorithms for complex 3D environments is non-trivial. This is mainly due to two reasons; firstly, learning an accurate forward model that can generalise to unseen complex 3D scenes is extremely challenging. Secondly, developing a planning algorithm which is robust to the inaccuracies of the forward modelling is non-trivial. In order to alleviate those problems, we propose to augment the model-free RL with a forward model that learns to predict (i.e. to imagine) only the successful future states. By limiting the imagination to a single future state, which we call *sub-goal* state, we effectively

simplify the forward modelling problem. We then integrate our forward model into actor-critic RL and develop an imaginative RL agent which we show outperforms previous state-of-the-art methods.

- **Harmonious Agents:** The success of single-agent RL has recently motivated its extension to multi-agent problems [59, 98, 121]. In a multi-agent system of decision-making, multiple agents learn and adapt their policies, simultaneously, while interacting with one another. The constant learning and adaptation of all agents renders the environment non-stationary from the perspective of every single one of them [79]. Therefore, extending RL to multi-agent systems is inherently more challenging compared to single-agent learning. Moreover, in many real-world problems, agents optimise their own individual reward function, which may partially, if not fully, contrast with that of other agents [41, 54]. A real-world example of such problems is learning the autonomous driving policies of multiple agents (e.g. a fleet of autonomous vehicles). It is desirable for all autonomous vehicles to reach their given destination in the safest and most efficient possible way. As we show in Chapter 6, trivial independent learning of such policies leads to aggressive behaviours against other agents (e.g. not giving way at an intersection). Evaluating such aggressive policies across the population of agents reveals high performance disparity among the agents measured using their expected return. In other words, the agents with a more aggressive policy achieve higher efficiency at the cost of that of more conservative agents. We propose to alleviate such a lack of harmony among the agents by drawing inspiration from recent work on *empowerment*. Intuitively, we train agents to not only maximise their individual expected return but to *empower* their neighbours through our mutual information objective. That way, we improve the overall performance harmony across the agents. Furthermore, we develop an increasingly challenging environment where agents may break their rationality assumptions. Such scenarios may happen when some agents behave incautiously due to reasons such as long waiting times. We show that our agents are more robust to such irrational behaviour.

The rest of the thesis is organised as follows. In Chapter 2, we provide the preliminary information required to understand our proposed methods, which are discussed in later chapters. We discuss the specific mathematical frameworks used to precisely define each problem, notations, and the detail of all the simulation environments we use or develop. In Chapter 3, we provide a review of the most relevant literature on our three main proposed methods and discuss their advantages and differences. In Chapters 4, 5, and 6 we discuss the detail of our proposed methods and provide our empirical results and ablation studies. Finally, in Chapter 7, we conclude by summarising the thesis and discussing future work directions.

Chapter 2

Preliminaries

In the previous chapter, we introduced our three main research hypotheses. In this chapter, we provide further background information on the problems at hand, the benchmark simulators which have been used, and the preliminaries required to better understand our proposed methods. Moreover, we discuss the details of the mathematical frameworks utilised to model both our single-agent and multi-agent RL problems, i.e. Partially Observable Markov Decision Processes (POMDPs) and Partially Observable Stochastic Games (POSGs). We then provide the standard notations used throughout the related chapters of this thesis for each problem.

2.1 Part I: Single-Agent Problems

2.1.1 Partially-Observable Markov Decision Processes

Markov Decision Process (MDP) is a mathematical framework that allows the modelling of sequential decision-making problems addressed using RL. In the visual navigation problem, discussed in Sec. 2.1.3, the agent can only receive a partial observation of the environment at each time-step. We define the problem as a Partially-Observable Markov Decision Process (POMDP) denoted by the tuple $\{O, S, A, \mathcal{G}, \mathcal{P}, r, \gamma\}$ [98]. Here, O is the space of visual observations, S is the state space as encoded and observed internally by the agent, A is the action space, $\mathcal{G}$ is the set of target objects given by the environment, $\mathcal{P} := p(\mathbf{s}_t|\mathbf{s}_{t-1}, a_{t-1})$ is the transition function or environment dynamics model (unknown) for state $\mathbf{s}_t \in S$, and r is the reward function and γ is the reward discount factor.

2.1.2 Actor-Critic RL

In order to solve a POMDP, different RL algorithms have been introduced such as Deep Q-learning [58] and Policy Gradients (PG) [99]. Actor-critic [67] algorithms have mainly been introduced to alleviate the high variance in gradient estimates of PG algorithms. A2C, A3C [67], and PPO [93] are among the most commonly used actor-critic algorithms due to their simplicity in implementation and training. In this thesis we use variations of A3C (in Chapters 4 and 5) and PPO in Chapter 6.

The main difference between actor-critic and PG algorithms is learning a state-value function in addition to the policy, which is then utilised to improve policy learning. The critic, a.k.a as the advantage function, is defined as below:

$$A_\theta(o_t, a_t) = r_t + \gamma V_\theta(o_{t+1}) - V_\theta(o_t), \tag{2.1}$$

$$V_\theta(o_t) = \mathbb{E}_{\tau\sim\pi}[\sum_{i=t}^{T} \gamma^{t-i} r_i \mid o_t]. \tag{2.2}$$

The state-value function in Deep RL is approximated using a neural network (parameterised by θ) using the observed values from the collected trajectories, using supervised learning:

$$\mathcal{J}_V(o_t, \boldsymbol{\theta}) = \frac{1}{2}(V_\theta(o_t) - R)^2, \tag{2.3}$$

where R is the target return values, which is the the discounted sum of the total reward obtained during an episode: $R = \sum_{i=0}^{T} \gamma^i r_i$, where T is the episode horizon. The policy is then updated using the approximated state-values using the objective below:

$$\begin{aligned} \mathcal{J}_\pi(a_t \,|\, o_t, \boldsymbol{\theta}) = -\log \pi(a_t \,|\, o_t, \mathbf{g}_\tau; \boldsymbol{\theta})(r_t + \\ \gamma V_\theta(\mathbf{s}_{t+1}) - V_\theta(\mathbf{s}_t)) + \beta_H H_t(\pi), \end{aligned} \tag{2.4}$$

where $H_t(\pi)$ is the estimated entropy of the policy at each timestep and β_H is the coefficient for entropy exploration. In A3C, it is conventional to use entropy to encourage exploration in the beginning of the training.

2.1.3 Visual Navigation as a POMDP and AI2Thor Simulator

In visual navigation, the objective for the agent starting from a random initial location is to take a sequence of actions (collectively, a trajectory) to reach a given target object using only observed ego-centric RGB visual inputs. A navigation episode ends if either the agent takes the "STOP" action or the maximum number of permitted actions is exhausted. A trajectory is considered successful if the agent stops within a defined circular proximity of any instance of the target object. Visual navigation is commonly categorised into point-goal and object-goal navigation problems [15].

Object-goal visual navigation is the task where an agent needs to navigate towards an instance of a given target object in a previously unseen environment. This is an inherently challenging problem since the agent needs to find the shortest possible sequence of actions to first find a target object (i.e. to intelligently explore) and then navigate towards it (i.e. to plan and act accordingly while avoiding obstacles) in a visually complex 3D environment. The task is further complicated when at each time step the agent only receives limited field-of-view visual inputs only. This means the agent will need to learn simultaneously to (1) build a good state representation that enables the agent to localise itself as well as the target (if observed at some point) and (2) efficiently use that state representation to (implicitly) plan and navigate towards the target while avoiding the obstacles.

In a navigation episode, from every starting point, there are typically multiple action sequences (i.e. trajectories) that could lead to success, let alone the sequences that might fail. Learning to select the right action at each time step to create a trajectory that leads to the specified target object is the primary challenge.

The sequential nature of acting in the environment allows visual navigation to be defined as a POMDP. At each timestep, the agent receives an observation from the environment (i.e. RGB image), which is sampled from the state using an unknown underlying observation function. The agent then acts based on its trained policy to receive a reward and the next observation. An episode ends either by reaching the goal state or exhausting the maximum episode length. The completed episodes (a.k.a trajectories) are then used to update the policy parameters. This process is iterated until convergence.

Training RL agents to iteratively train a navigation policy in the real world is infeasible. That is mainly due to the sample inefficiency of the current RL algorithms. In order to address that problem different photo-realistic simulation environments have been introduced [52, 89]. In this thesis, we choose to work on the AI2Thor [52] environment which contains four different rooms in a household environment, e.g. bathroom, bedroom, living room, and kitchen. Examples of the scenes in this simulator are shown in Figure 2.1. At the beginning of each episode, the agent is placed in a randomly selected location within a randomly selected room type and provided with a randomly selected feasible object as the target.

AI2Thor was photo-realistic which made any learned navigation policy more easily transferrable to the real world. Compared to its existing counterparts [89], it was also more suitable for training RL policies given the known sample inefficiency issues, due to relatively smaller room sizes. Moreover, continuous state space (of possible navigation locations) makes it more real-world applicable. Finally, the number and diversity of scenes and customisability of the object organisations added to the attractions of this simulator for further challenging the trained policies. The latter features are specifically missing from more recently introduced simulators, e.g. [89].

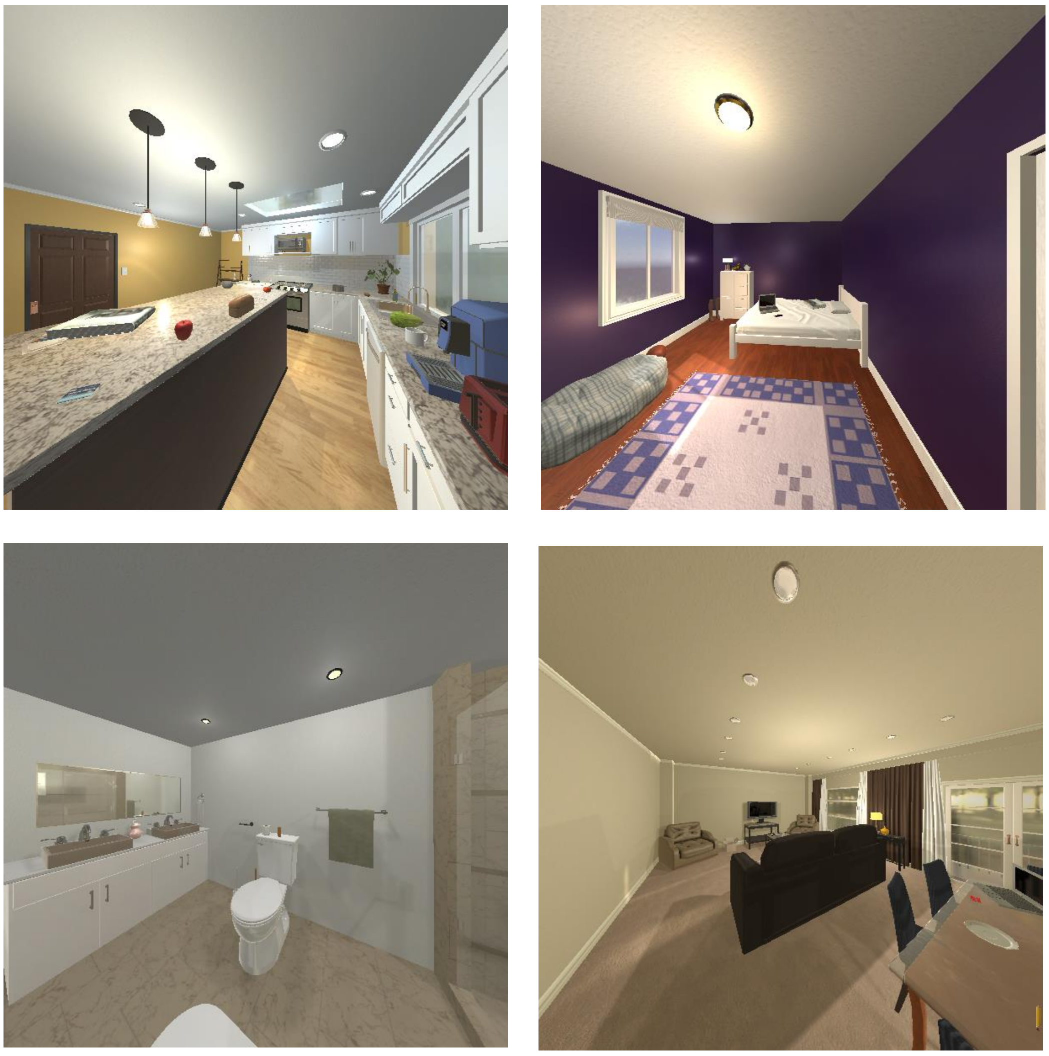


Figure 2.1: Example of four different room types in AI2Thor environment: ***top-left:*** *kitchen,* ***top-right:*** *bedroom,* ***bottom-left:*** *bathroom,* ***bottom-right:*** *living room.*

2.2 Part II: Multi-Agent Problem

While single-agent RL continues to advance it is limited by assuming the agent interacts with a stationary environment. In other words, the framework of POMDP we introduced in the previous section, does not consider the existence of other agents in the environment affecting the future observed outcome of another agent's policy. Motivated by those limitations, we decide to go beyond single-agent RL and introduce environment non-stationarity [79] arising from the existence of other learning agents.

Recent works have extended the problem of visual navigation to cooperative multi-agent navigation [108] and object relocation [26]. Such problems, however, are limited by the number of possible interacting agents as well as the pre-defined cooperative role of involved agents. Therefore, we choose to work on the related problem of multi-agent RL for autonomous driving. In autonomous driving, the number of interacting agents is significantly higher which increases the challenges of the environment's non-stationarity. Moreover, the interactions are ad-hoc and agents may constantly change their cooperative or competitive roles. Nonetheless, autonomous driving and visual navigation policies share the similarity of learning policies to successfully relocate an agent from a random starting point to a given target destination.

2.2.1 Partially-Observable Stochastic Games

Depending on the role of the interacting agents, different frameworks can be utilised to define the multi-agent problem. Here we use the most general framework of Partially-Observable Stochastic Games (POSGs). POSG consists of the tuple $< \mathcal{N}, \mathcal{S}, \{A_i\}, \{O_i\}, \{Z_i\}, \mathcal{P}, \{r_i\}, \{\gamma_i\} >$, where $\mathcal{N}$ is the set of agents present in the environment, $\mathcal{S}$ is the state space, A_i is action space of agent $i \in \mathcal{N}$, O_i is the observation space of agent i based on its observation function $Z_i : \mathcal{S} \times \mathcal{O}_i \rightarrow [0, 1]$, $\mathcal{P}(s^{(t+1)}|s^{(t)}, \boldsymbol{a}^{(t)})$ is the environment transition probability, $r_i : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is the reward function for agent i and γ_i is its discount factor. In POSG, every agent maximises its own objective, $J_i = \mathbb{E}[R_i|\pi_i, \pi_{\neg i}]$, where $R_i = \sum_t \gamma_i^t r_i^{(t)}$ is the discounted sum of its future reward, according to its own reward function, and π_i is its policy. We use sub-script indices for agent index and super-script in parentheses, where required, to indicate the time index, and $\neg i$ means all the agents except agent i.

2.2.2 MetaDrive Simulator and Our Extended Version

Several different autonomous driving simulators have recently been introduced. In this thesis, we focus on learning the decentralised policies of multiple agents when interacting with others in high-density complex driving scenarios. Therefore, instead of working on photo-realistic simulation environments, e.g. CARLA [21], we choose to work on the MetaDrive [55] simulator to develop our method. MetaDrive is a lightweight and efficient multi-agent learning platform which can scale up to large number of agents. In MetaDrive, several complex driving scenarios were introduced originally, as shown in Figure 2.2.

Our extension of the MetaDrive environment includes the following driving scenarios from the original implementation, each highlighting specific multi-agent interaction challenges:

(a) Bottleneck: In this scenario the agents are spawned on one end of the road (left or right) and their goal is to reach the other end of the road, travelling through the shortest path and as fast as possible. The number of lanes at the bottleneck is reduced from four lanes to a single lane on each side of the road; that is when agents can either be aggressive and selfish or considerate and pro-social. We extend this to N-Bottleneck version to investigate the effect of past performance on future behaviour.

(c) Roundabout: The main challenge in this scenario is to merge into the roundabout efficiently. Here an agent inside the roundabout will open more space for the agents waiting to enter at a small cost of its own reward but empowering other agents to showcase social behaviour.

Moreover, we add the following new scenario motivated by the real-life deriving challenges:

(b) T-Intersection: The agents are spawned randomly and follow their set goals similar to (a). Agents driving through the vertical road aiming to reach the right end of the main horizontal road are at a disadvantage due to the challenge of crossing the horizontal road. Without cooperation from the agents driving on the horizontal road, those disadvantaged agents will be waiting for extended periods (hence will receive a lower reward).

2.2.2.1 Self-Aware and Irrational Agents

Most of the current works on autonomous driving focus on learning driving policies designed to interact with rational agents which have stationary policies [3, 80, 103, 106]. Such assumptions, however, in real-world scenarios may not always stand correct. For instance, it is known that drivers' behaviours can be affected by their mental state [85, 86]; angry drivers behave less rationally which may endanger other road users' safety. We intend to introduce a mechanism to simulate such irrational behaviours into MetaDrive.

In order to simulate irrational agents, we draw inspiration from social driving behaviour studies [85]. We make several changes to the environment to simulate such behaviour:

- We make every agent self-aware. That is, every agent's observation includes an additional variable that indicates its performance thus far into an episode, which we call $Self-Awareness$ variable. For performance measures, we either use the environment return (i.e. accumulated discounted reward), or the number of relative overtakes. The latter provides an indication of the relative performance of an agent compared to its observed neighbours.
- We define the emotion variable for every agent as a truncated exponential function of the temporal difference in $Self-Awareness$ variable. That means the agent's emotion level (which can be interpreted as anger) increases as the observed improvement in $Self-Awareness$ drops. This essentially simulates the emotional process of starting to lose temper as the rate of gaining more reward drops, e.g. the agent is not getting closer to the destination.
- Finally, we design and implement a mechanism to simulate increasingly irrational behaviour as a function of the emotion variable for every agent. We achieve that by using several different approaches to increase the level of noise in agent's observation, e.g. missing a nearby object or obstacle.

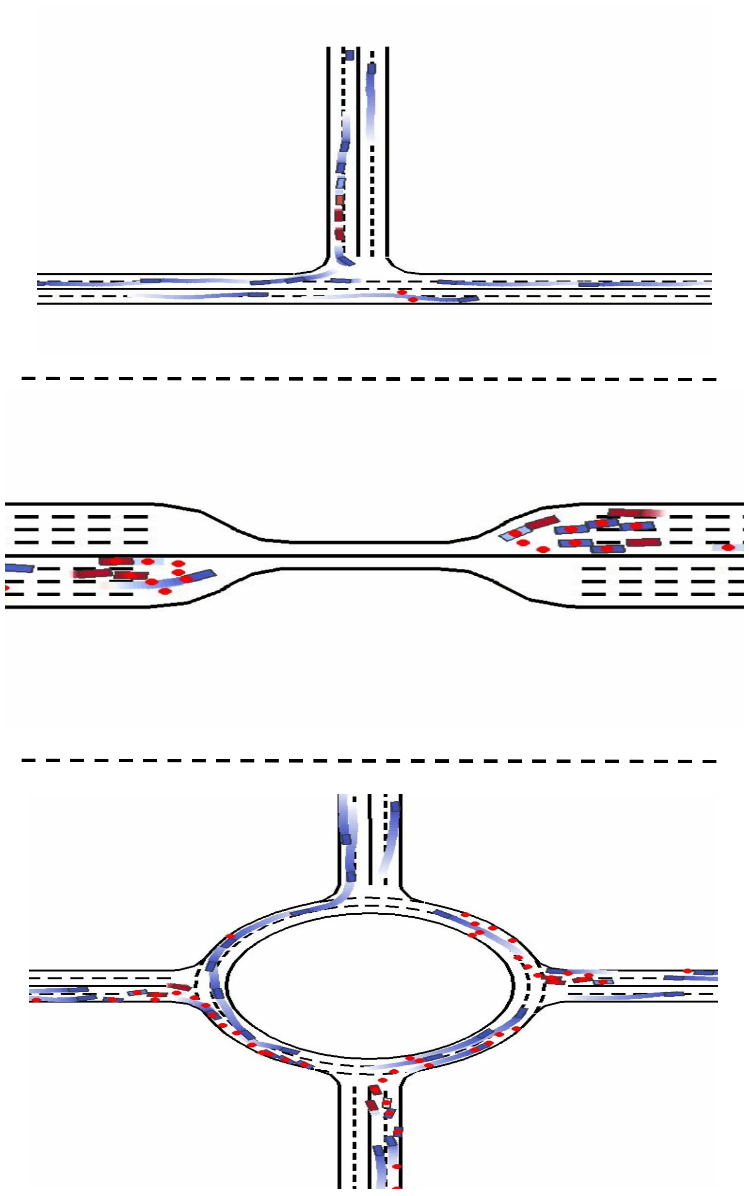


Figure 2.2: Current driving scenarios in MetaDrive simulator. Blue rectangles show learning vehicles, blue gradients vehicle trajectories, and red dots crash points.

Chapter 3

Literature Review

Research on single-agent and multi-agent RL has gained significant attention, recently. Some of the recent works focus on developing general-purpose RL algorithms in simple simulation environments, while others focus on tackling specific problems, e.g. visual navigation or autonomous driving. In this chapter, we first categorise existing RL methods into single-agent and multi-agent methods, which we discuss in separate sections. We then provide a review of the relevant literature from two different perspectives: general-purpose RL algorithms which are relevant in terms of the method only and RL frameworks developed for our specific problems at hand. The aim is to provide both an overview of the research in RL as well as an in-depth analysis of the most relevant works to our main contributions to this thesis.

We organise this chapter as follows: in Sec. 3.1 we focus on the literature related to the methods discussed in Chapters 4 and 5. We discuss methods focussing on both the RL method as well as the visual navigation problem. In Sec. 3.2 we focus on MARL as relevant to our method discussed in Chapter 6. Similarly, this includes relevant works both to the method and the autonomous driving problem.

3.1 Part I: Single-Agent RL and Visual Navigation

3.1.1 Reinforcement Learning Methods

Model-Based RL

ForeSI uses a forward model to generate future state representations, thus is closely related to model-based RL [36, 44, 45, 91, 123]. Weber et al. [111] train a recurrent model of the environment dynamics that can be used for planning by unrolling an episode in imagination. Alternating between learning a model of the environment dynamics and using such model to train a policy has recently been a major theme of work [34, 36]. In such methods, the policy is trained purely on imagined episodes. This training paradigm reduces the need for collecting extra trajectories as the agent can imagine not just what it has observed, but also interpolate in its learned space to imagine the unseen. In most of such methods, the environment dynamics model is a recurrent reconstruction-based model inspired by [33] which models every single state transition. Such methods have only recently surpassed the performance of model-free baselines, and only on simpler toy environments [45]; their generalisation to realistic visual navigation in a 3D environment has not yet been explored. For us, it means that our method is the first step towards enabling model-based RL in a complex real-world environment, i.e. visual navigation. In contrast to those methods, we enable longer-term forward modelling without performing step-wise future prediction which boosts the navigation performance in unseen environments while reducing the computation cost.

Experience Replay

Modifying the distribution of episodes in a replay buffer, known as hindsight experience replay, has been explored in other applications [6, 84, 101]. The core idea of those methods is to modify the distribution of goal states in the replay buffer (i.e. hindsight) to train a more generalisable policy able to reach arbitrary goals provided during test time. Such hindsight manipulation of the observations can become very costly in complex environments. That is mainly due to the computational cost of learning generative models of the environment that can accurately model important detail. Differently from them, here we introduce a method that is trained in hindsight and tested in foresight to generate successful (sub-)goal states for visual navigation.

Goal-Conditioned RL

Our method is also closely related to goal-conditioned RL where the policy is trained conditioned on a given goal state. In those methods, since the start and goal state are known a-priori one could use combinatorial optimisation on the states encoded using a Variational Auto-Encoder (VAE) [48] to find the best set of reachable sub-goals along a past trajectory [74]. The selected sub-goals can then be used to optimise the policy using Q-learning [68], an off-policy RL algorithm. Alternatively, the goal states can be imagined using a leared model of the environment to train the policy to reach arbitrary goals [72]. Following similar ideas, recently, Kanu et al. [46] use a VAE to learn to generate goals during test time, too. While those methods are related to ours, there are two fundamental differences: (1) unlike ours, the application of those methods has been limited to simpler environments where there is little to no difference between training and test environments; (2) unlike those most of those methods we use the selected sub-goals during the training to learn to generate them in unseen test environments, for the success.

3.1.2 Visual Navigation Problem

Visual Navigation

Research on visual navigation has attracted increasing attention in recent years mainly due to the availability of photo and sensor-realistic 3D simulators [20, 52, 89, 112, 117]. Various methods [14, 16, 24, 65, 115, 119, 125] have been proposed investigating different aspects of the problem. Some of such methods are based on Imitation Learning (IL). IL allows initialising a better policy (than random weights) [23, 24] when expert demonstrations (i.e. state-action pairs) are available; however, those demonstrations are expensive to collect in real-world problems. Additionally, policies trained using IL only typically suffer from lack of generalisation. Therefore, model-free RL algorithms are at the heart of most of the previous approaches with better performance [14, 16, 115, 119, 125]. Wortsman et al. [115] propose to use RL along with meta-learning [27] to enable test-time adaptation of the policy to unseen navigation environments. In another line of work, Du et al. [23, 24] focus on improving the perception of the navigation agent. They achieve better perception by developing a graph neural network that uses an off-the-shelf object detector [83] to exploit object co-occurrence relationship. Such object relationships are key to determine the potential location of an unseen object. For instance, observing a kettle can inform the agent a toaster might be nearby. In their more recent work, Du et al. [24] propose to further improve the perception by developing a Transformer-based model [104] to learn the object relations more efficiently. In contrast to those works, the main focus of our work is to improve the core RL algorithm. We achieve that by investigating the role of critic and future imagination in RL-based navigation policies. We show our method works well along with multiple different perception modules [23, 66, 115, 119], whether it be a GNN or a Transformer-based architecture.

Incorporating External Knowledge

Adding some form of prior to machine learning models has fast-tracked the progress in many different computer vision tasks [32, 90, 95]. The prior can be in the form of pre-trained weights of the network for object detection or segmentation tasks [90] or in the from of Graph Neural Networks (GNNs) representing common-sense knowledge [32].

In the RL framework, GNNs have been used to improve navigation policies by representing topological environment maps for more accurate localisation [65] or helping with more efficient exploration [19]. Generating and incorporating scene graphs [32] is also closely related to our problem. However, here we construct our knowledge graph externally and refine the node relationships without explicitly detecting the objects in the observation. This also separates our approach from some closely related prior work [22, 71] where an off-the-shelf object detector is used. Such methods, while improving the performance, have been explored before and can be plugged into other frameworks for further improvements, including ours. Additionally, increasing the complexity of the policy architecture increases the sample complexity of RL algorithms. As a results, such complex models are generally trained with IL [22] which is different from our objective. In this thesis, we are interested in developing novel RL-based methods to address the shortcomings of RL for real-world problems. Training complex policy architecture with RL using sparse reward remains an open problem.

Similar work to ours is proposed by Vijay et. al. [105] where the prior knowledge is injected into RL for navigation. The authors propose to learn various edge features encoded as one-hot vectors. The main difference, however, is that they address the navigation in a 2D fully-observable environment while in our case we deal with a near-real-world task of navigation in 3D environment. Similarly, the use of a knowledge graph to learn to find unseen objects by learning the correlations to the seen targets during the training has also been explored [119]. One of the deficiencies of such method is that it effectively increases the state space size. This is specifically important in RL framework where increasing the model's complexity can impede improving the performance, as we show in this thesis.

3.1.3 Value Function Estimation in Actor-Critic RL

One major drawback of model-free RL methods is the high sample complexity. This is specifically worsened in our 3D environment set-up where the agent has to learn the visual associations and ground the language instruction while learning the optimal policy using only sparse rewards. In order to mitigate this in actor-critic RL, some of the parameters of the policy are shared with the value estimation module which provides a stronger learning signal for the shared parameters. An accurate and stable estimate of the value, however, is necessary to reach this objective [92, 100].

Separating the parameters of the advantage and value functions to increase the accuracy of action-value function estimation has previously been investigated [109]. Inspired by that, we propose a novel value estimation module for visual navigation. This module can estimate the value of each state more accurately incorporating the object relationships encoded using a graph neural network.

3.2 Part II: Multi-Agent RL and Autonomous Driving

3.2.1 Multi-Agent Reinforcement Learning

Reinforcement Learning (RL) has recently gained increasing attention, mainly due to its success in solving games [91] and other toy problems [36, 45, 68]. In many real-world problems, such as autonomous driving [21, 94], however, an environment involves other learning agents. Accordingly, Multi-Agent RL (MARL) has evolved as a solution to learn a decentralised policy for multiple agents [60, 88]. Different frameworks have been developed which categorise different methods into cooperative [76], competitive [9], and mixed-motive environments [63]. A majority of the existing work are based on the Decentralised Partially Observable Markov Decision Processes (Dec-POMDP) framework [76] in cooperative environments [28, 82, 97]. Addressing the problem of credit assignment in those works has been a major focus. Another established framework is Partially Obseravle Stochastic Games (POSGs), e.g. [37]. In a POSG, agents may have individual reward functions different from those of other agents. This is more interesting, and challenging, since addressing the alignment between those objectives becomes a major problem [80, 118].

3.2.2 Multi-Agent Reinforcement Learning and Autonomous Driving

Autonomous driving has been mostly studied under the environment stationarity assumption where single-agent learning or planning is used against other fixed policies [2, 3]. More recently, multi-agent learning for autonomous driving has been studied. In some of the previous works [7, 94] autonomous driving is defined as a cooperative MARL problem. This is in contrast to the previous works which define a centralised policy to better control traffic flow and reduce congestion [18]. Differently from those works, in this thesis we consider the problem from the perspective of mixed-motive MARL. In our setting, agents may demonstrate different cooperative or competitive behaviour over the period of a single episode. The agents are, also, self-interested, and a cooperation mechanism is required to avoid aggressive behaviour [80].

3.2.3 Cooperation Among Self-Interested Agents

Multi-agent coordination has been well studied in cooperative settings where agents are maximising a shared reward, i.e. working in Dec-POMDP [76] framework. Among those a main line of work has focused on addressing the multi-agent credit assignment problem using value function decomposition [82, 97]. Such decomposition helps every agent to better understand personal contribution to the team reward, hence improving the cooperation. Promoting cooperation in mixed-motive settings, however, is more challenging and has been less studied. Agent modelling has been one area to help alleviate that problem [1]. Lowe et al. [60] propose to integrate a model of other agents' policies into policy gradient using and uses centralised value learning to improve cooperation in mixed cooperative-competitive settings. Note those methods, however, assumes fixed roles during an episode whereas in our setting agents do not take fixed roles.

Mixed-motive cooperation has also been studied in Sequential Social Dilemmas (SSDs) [54]. An active are of research in SSDs is incentivisation, where agents are directly incentivised to perform certain, i.e. cooperative, behaviour. For instance, recent works propose to train agents that are able to reward other agents to influence their actions, as part of their policy [41, 118]. Differently, some other works propose to increase, i.e. maximise, the diversity of behaviours among agents for learning more generalisable multi-agent interaction policies [64]. Looking at the policies of multiple agents from the perspective of fairness has also been recently explored. For instance, Jiang and Lu [43] introduce Fair-Efficient (FE) reward and hierarchical policy to promote diverse cooperative behaviour and increase performance uniformity, hence fairness among agents. In many real-world problems, however, including our autonomous driving setting, agents face inter-temporal dilemmas. Moreover, there is no clear distinction between cooperation and deception rather there is a continuous range of cooperativeness.

Closest to our work, more recently, Peng et al. [80] utilise average neighbourhood reward among agents to improve coordination. Any such reward shaping approach is known to be prone to reward hacking [87]. Additionally, in ad-hoc interactions, assuming an agent's overall return is influenced by any ad-hoc interaction during an episode does not always hold. Other works show that road rules emerge as a result of noisy observation [78]; the authors focus on learning more granular social behaviour in more complex driving scenarios involving a larger number of agents.

Chapter 4

Optimistic Reinforcement Learning

In the previous chapters, we introduced our research hypotheses, provided background information and defined the mathematical frameworks used throughout this thesis. In this chapter, we discuss our first contribution in which we investigate the role of critic in actor-critic RL algorithms used to learn visual navigation policies. In most of the previous RL-based methods, actor and critic share the same network parameters to help with learning richer state representation more efficiently. In our method, we show that a stronger critic helps with learning a more robust policy that is more robust to early capitulation.

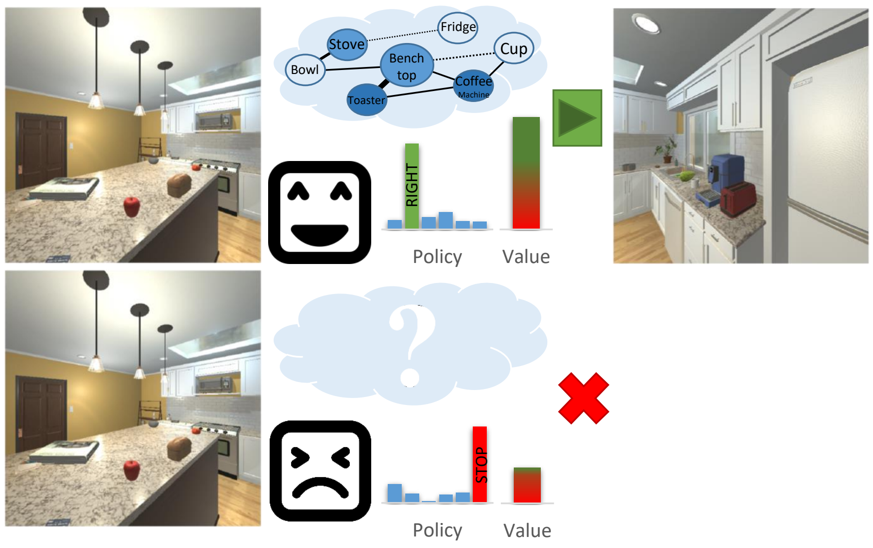


Figure 4.1: Our optimistic agent estimates the state value more accurately with the help of a graph representing the relationship between object locations. The more accurate value estimation helps with learning a more optimal policy that is robust to early capitulation (e.g. stopping before reaching the target object as a result of under-estimating the state value).

4.1 Background

Human beings are capable of finding a given object in an unexplored environment, *e.g.* a room in a new house, given just a short natural language command, *e.g.* name of the object. We primarily rely on our prior knowledge of the way different objects are co-located in a specific room in addition to the new sensory input information we receive. For example, we know in any bathroom, a soap bottle should be located near the basin, thus observing one (assuming a limited field of view and/or longer distance) helps us find the other more easily (*e.g.* by providing clues). Our belief, however, needs to be adjusted upon receiving new observations in a previously unseen environment. For example, the soap bottle might be misplaced or it might be in a new unseen shape or colour. Finally, we do not normally give up on searching as soon as we fail to find the target in the first attempt. In other words, we optimistically continue our search as long as we receive indications from the environment that the target is reachable. It is desirable for every navigation agent to be able to utilise prior knowledge, be prepared for adapting the beliefs to a new unseen environment and be robust to early capitulation.

To endow the navigation agents with the first capability, recent works [56, 62, 119] use variations of Graph Neural Networks (GNNs) to encode the object-object relationships, utilising external pre-training, in the form of a knowledge graph. Such methods, however, are generally based on Graph Convolutional Network (GCN) [12] hence simple in the way the relationship between different graph nodes are modelled. As such, as we show in our experiments, they cannot scale to more complex models. Naive scaling to more complex neural graph modelling leads to increasing the complexity of the state-space which makes learning the optimal policy more challenging. Avoiding such extra state-space complexities is crucial to avoid, especially in our sparse reward RL setting, since it exacerbates the challenges around credit assignment and impedes performance improvement. On the other hand, to enable the agents with the second skill (e.g. adaptation), recent work [115] develop a meta-learning approach to allow the agent to quickly adapt to a new environment, using a few gradient-based weight updates [27]. In that method, the authors show relative improvement in generalisation to unseen test environments. However, their method does not explicitly model the relationship between different objects. Therefore, adapting such prior knowledge to new environments remains unexplored. Such adaptation, as will be shown in our ablation studies, is non-trivial in RL framework. One explanation is related to the complexity of the policy's neural network model (e.g. the number of parameters). As we increase the model's complexity, efficient training becomes more challenging with conventional RL objective functions in sparse reward settings.

Finally, the last mentioned desirable capability for a navigation agent to have is a realistic (read it as optimistic) estimation of it's expected future reward. Revisiting the bathroom example again, an optimistic agent will continue to search for a soap bottle that is not found in its immediate expected location, e.g. near the basin. An unrealistic agent, however, will give up as soon as it fails to find the soap bottle near the basin without further search. In actor-critic RL a more realistic estimation of the value function will lead to a more optimal policy that is more robust to early capitulation.

To that end, we propose our Graph-based Value Estimation (GVE) module to efficiently benefit from the prior knowledge for more accurate value function estimation. The prior knowledge presents the agent with general rules about the semantic relationships of the objects and their relative location in an indoor environment. Instead of using the graph as the input to the state representation encoder, which uncontrollably increases the effective size of the state space, we use the graph to estimate the potential return from the current state. This way the graph is used to better guide the policy training rather than being directly used for action taking. We present, in our experiments, that simply conditioning the policy on the prior knowledge does not lead to a more optimal policy with better performance.

Intuitively, observing a correlated object to the target that the agent is searching for has a strong indication of the final reward that can be gained, following the current policy. For instance, when searching for a book in a room, observing a desk or shelves from the distance has an indication of finding the book, hence the current state has a higher value. The association of these observations through unstructured trial and error in RL without incorporating the prior knowledge is not simply achievable; specifically, when objects are small the extracted visual features might not contain much information to guide the agent towards them.

Our GVE results in a more optimally trained critic that can associate the visual observations with its prior knowledge to better estimate the state value. This is essential in our framework where we use a variant of actor-critic RL algorithm (*e.g.* Asynchronous Advantage Actor-Critic; A3C [67], where the critic's role is to identify the actions leading to a specific reward using the temporal difference of the state values. We empirically show that GVE improves the value estimation accuracy. It is shown that the high accuracy of value estimation is essential for an optimal policy training [113], which is further confirmed in the performance of our proposed method.

Overall, we propose a simple, principled and modular approach that can be employed in conjunction with other approaches for visual navigation that also use an actor-critic RL framework. Our main contributions are:

- We introduce a simple, yet effective method that can endow an actor-critic navigation agent by improved integration of prior knowledge for navigation.
- We propose the GVE module which reduces the value estimation error in A3C RL algorithm. We empirically show the higher accuracy of our estimates which leads to a more optimal policy.
- Finally, we provide the new state-of-the-art performance results on the object-target

visual navigation task in AI2THOR environment. This is achieved in the more challenging scenario of navigation using the target object name only.

Below we discuss the details of our proposed method and our empirical studies. We aim to address the following research questions:

Why optimisim: is the lack of optimism reducing the performance of RL-based navigation policies?

Achieving Optimism: how can we induce more optimism in RL-based navigation policies?

For further discussions on the mathematical framework used to define the problem, i.e. POMDP, and the notations refer to Chapter 2.

4.2 Proposed Method

In this section, we first describe the task in detail and then discuss our proposed method.

We employ A3C [67] as one of the most effective actor-critic RL algorithms to train our policy. We focus on actor-critic methods because they are popular in visual navigation due to their efficiency, ease of use and robustness. In addition, these methods bridge between gradient-based and value iteration approaches inheriting their benefits. We hypothesise that our proposed method can be applied to other actor-critic methods with minimal modifications. In our approach, the model is parameterised with $= \{\boldsymbol{\theta}, \boldsymbol{\theta}_\pi, \boldsymbol{\theta}_v\}$; $\boldsymbol{\theta}$ is the set of parameters of the backbone conditional state embedding network, $\boldsymbol{\theta}_\pi$ is the set of parameters for the policy sub-network, *a.k.a* actor, and $\boldsymbol{\theta}_v$ is the set of value sub-network's parameters, *a.k.a* critic. We use a CNN to encode the observations and Glove [42] to encode the word embeddings for the target. More details on the network architecture are presented in methods and implementation details.

Conventionally, the actor and critic share the network parameters. However, they are *not* tasked with the same objective: while the actor needs a minimum representation to capture the environment to take appropriate actions at each time-step, the critic needs a sufficiently holistic representation to estimate how likely it could be for the agent to achieve its goal. The critic does not need every detail of the scene and requires a

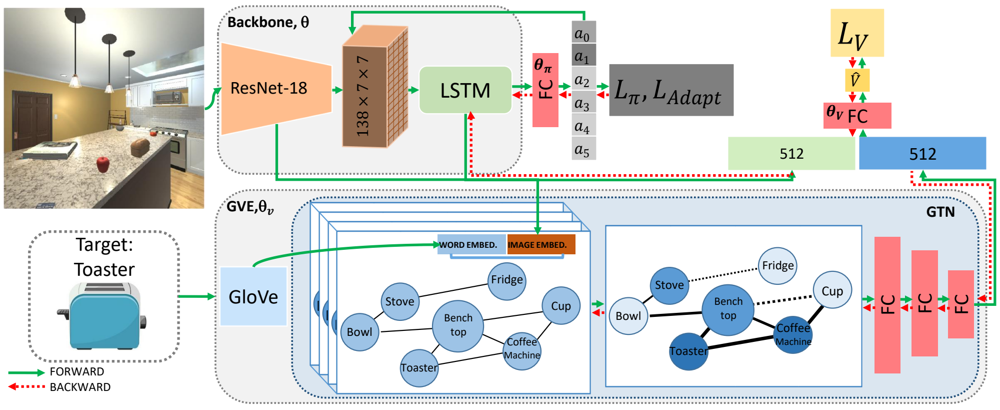


Figure 4.2: Overview of our method. Our GVE module augments the A3C backbone and is trained using the supervised loss L_V at each time step of the trajectory.

global representation as opposed to that of the policy. Through the feedback the critic provides, the actor improves which is an implicit guide for the policy. With this intuition, in our proposed approach we augment the critic's sub-network with our Graph-based Value Function Estimation (GVE) module that considers and updates a knowledge graph of the object relations. GVE leads to a more accurate value estimation which then reduces the variance of the gradient samples [113]. It is shown in [113] that a more accurate estimation of the value functions leads to a lower variance of the gradients which encourages more stable training and consequently finding a better policy.

For the adaptation, inspired by [115], we adopt Model Agnostic Meta-Learning (MAML) [27] to continuously adapt the learnt knowledge during test time. To do so, in our RL setup, we divide the training trajectories into meta-train D and meta-validation D' domains. D includes sub-trajectories of defined intervals and D' covers the complete trajectory. Then, L_{Adapt}, a loss function parameterised by $\boldsymbol{\phi}$ is learnt to compensate for the domain shift during test. We optimise the adaptation loss and the policy loss L_π according to the following update rules:

$$\begin{aligned}\boldsymbol{\theta}^i_{\text{total}} &\leftarrow \boldsymbol{\theta}_{\text{total}} - \alpha \nabla_{\boldsymbol{\theta}_{\text{total}}} L^{\tau\sim D}_{Adapt}(x_t, \pi(x_t); \boldsymbol{\phi}),\\ \boldsymbol{\theta}_{\text{total}} &\leftarrow \boldsymbol{\theta}^i_{\text{total}} - \beta_1 \nabla_{\boldsymbol{\theta}^i_{\text{total}}} L^{\tau\sim D'}_{\pi}(x_t, \pi(x_t)),\\ \boldsymbol{\phi} &\leftarrow \boldsymbol{\phi} - \beta_2 \nabla_{\phi} L^{\tau\sim D'}_{\pi}(x_t, \pi(x_t)).\end{aligned} \tag{4.1}$$

We define $\boldsymbol{\theta}_{\text{total}} = \{\boldsymbol{\theta}, \boldsymbol{\theta}_\pi\}$ in the above update rules for readability, α, β_1 and β_2 are the learning rate hyper-parameters and θ^i_{total} is the set of intermediate weights after the adaptation before the final update. In practice, we can have multiple intermediate updates i before the final update.

4.2.1 Learning Object Relationships

Our GVE module relies on a prior knowledge graph $G(V, E)$ which contains the information about the co-location of the objects in a scene. Intuitively, if the joint probability of observing object i and j together in a single view is high in an environment, then observing one should provide a strong cue for finding the other. Based on this intuition, inspired by the work of [119], we define an object co-location graph. The set of nodes V includes all the objects visible in the environment (*e.g.* whether used for navigation target or not) represented by their features. Each node feature, $\mathbf{v}_i \in \mathbb{R}^d$, encodes the concatenation of the RGB observation features extracted using a CNN and the semantic vector embedding of the object, extracted using a word embedding model. The edges show the co-location of the objects; that is, e_{ij} becomes incresingly closer to one during the training if and only if the objects i and j often appear in the same egocentric view of the agent, simultaneously.

We have multiple separate graphs encoding the objects visible in different room types. This is one of the novelties of our method which is, particularly, important since similar objects might appear in different rooms while their co-locating objects might be different. This will allow for having different edge values for our graph in each room. For instance, the object "bowl" might appear both in the "kitchen" and the "living room". It might be observed near the "cabinets" in the "kitchen" but close to "sofa" in the "living room". Therefore, our adjacency matrix, $A \in \mathbb{R}^{n\times n\times C}$, is a three-dimensional tensor where each channel C encodes the knowledge specific to a room type. We perform this by initialising the edges between objects that do not appear in each specific room type with zero. The graph separation enables the agent to mainly attend to one of the graph channels in each scene and avoid distraction. This way, more scene-specific knowledge can be encoded.

Inspired by [119], we also initialise the graph's binary edge weights using the co-occurrence of the objects in images acquired from the Visual Genome dataset [53].

However, we learn more accurate edge weights along with node features during the training using the attention mechanism, as proposed in Graph Transformer Network [120].

4.2.2 Graph-Based State-Value Learning

It is proven that policy gradient algorithms in RL have a high variance of gradient estimation and different solutions have been proposed to address this [31, 116]. The issue is mainly due to the difficulty of credit assignment which is exacerbated where the reward is sparse. In actor-critic methods, the variance is reduced by using the boot-strapped estimates of the state-value function as a baseline. It is also known that a more accurate estimate of the value function leads to a more optimally converged policy [113].

Accurate estimation of the state value in visual navigation is significantly challenging, particularly when searching for a target object. This is mainly due to the lack of global context at each time step where the agent can only observe a limited view of the scene. The context is conventionally encoded using the recurrent neural networks which are prone to forgetting and challenging to train. We propose our Graph-based Value Estimation (GVE) module to address this problem. Using GVE we encode the global context of object locations into our model so the agent has a better understanding of its position in the scene with respect to other objects. This will help the agent more realistically estimate the state value. Intuitively, there's a correlation between the objects present in each time-step's viusal input and the target object to navigate to. For example, when navigating to a smaller object like "book" in a room, the agent may look for larger indicator objects like desk or shelf that guide it towards the target. This means that the agent should be *optimistic* about finding the target object by observing a correlated object based on its prior knowledge. In other words, a state where a correlated object is visible should still have a high estimated value, close to the value of a state where the target is visible.

In order to encode our prior knowledge graph for value estimation, we propose to use Graph Transformer Networks (GTN) [120]. Using GTN our agent is able to learn new edges and update the edge weights. This way, depending on the scene, we are able to extract features from the graph that help with more efficient navigation. In order to update the original edge weights we first learn a soft weighted average of the edges across multiple channels of the input adjacency matrix:

$$H_i^l = \text{softmax}(\boldsymbol{W_i^l})A, \tag{4.2}$$

where $W_i^l \in \mathbb{R}^{1\times 1\times C}$ is a weight matrix over the channels. H_i will be the concatenation of multiple H_i^l matrices to learn multiple new mappings of the original adjacency matrix. In practice we can learn M different H_i and the final new adjacency matrix is then defined as the matrix multiplication of those which can encode the common paths:

$$A_{NEW} = H_1 H_2 ... H_M. \tag{4.3}$$

Finally, in order to encode the node features using the new adjacency matrix A_{NEW}, we use graph convolutional layers defined as:

$$Q = \sigma(\tilde{E}_i^{-1}\tilde{A}_{NEW}\boldsymbol{W_N}N), \tag{4.4}$$

where Q is the final embedding of the graph, $N \in \mathbb{R}^{n\times d}$ is the input node features, $\boldsymbol{W_N}$ is the weights for node embedding, $\tilde{A}_{NEW} = A_{NEW} + I$ is the augmented learned adjacency matrix with self-connections using identity matrix I and $\tilde{E}^{-1}$ is the inverse degree matrix for $\tilde{A}_{NEW}$. In practice, we can learn multiple convolutional layers embedding different node features using the new adjacency matrix. Therefore, the output graph representation vector Q is the result of both node and edge operations dynamically learnt during training.

In order to incorporate the graph's encoding into our GVE, we partially separate the parameters of the critic sub-network, $\boldsymbol{\theta}_v$, from the actor sub-network $\boldsymbol{\theta}_\pi$. Therefore, in our method, we estimate the state-value function according to:

$$\hat{V}(x_t) = \boldsymbol{W_1}Q + \boldsymbol{W_2}F(x_t, Z; \boldsymbol{\theta}); \tag{4.5}$$

$\boldsymbol{W_1}$ and $\boldsymbol{W_2}$ are aggregation parameters for the final value estimation which are also trained along the other parameters of the network and $\boldsymbol{\theta}$ is the set of parameters for F, the state encoding backbone network.

As can be seen in equation 4.5, our method decomposes the expected future reward into two components: one, estimated mainly conditioned on the state as encoded by the encoding network parameterised with $\boldsymbol{\theta}$; and the other component, Q, estimated by the graph encoding network which represents a correlation between the target object and the objects that are visible in the current state x_t. Therefore, the agent can include the expectation of the objects it might observe in the future states in order to estimate the value of current state more accurately. In other words, the agent has a more global understanding of the environment.

4.3 Experiments

As discussed in Chapter 2, we choose AI2Thor [52] environment for our experiments. This simulator consists of photo-realistic indoor environments (e.g. houses) categorised into four different room types: kitchen, bedroom, bathroom and living room. In order for fair comparison, we follow the same experimental setup as our main baseline [115]. In this setup, 20 different scenes of each room type are used for training; 5 scenes for each as validation and 5 for test.

We follow the recent convention in object-goal visual navigation task to measure the performance of our method based on Success Rate (SR) and Success weighted by Path Length (SPL).

4.3.1 Implementation Details

Our actor-critic network comprises of a LSTM with 512 hidden states and two fully-connected layers one for actor and the other for the critic. The actor outputs a 6-dimensional distribution $\pi(x_t)$ over actions using a Softmax function while the critic predicts a single scalar value. The critic sub-network also receives the GTN encoding as a vector of size 512 besides the 512-dimensional LSTM encoding. The concatenation of these two feature vectors is mapped to the value using a fully connected layer.

We use Glove [42] to generate 300-dimensional semantic word embedding of the objects and a pre-trained ResNet-18 to extract the features from the 300×300 image inputs. We concatenate these two to feed to our state encoder, LSTM. The overview of the

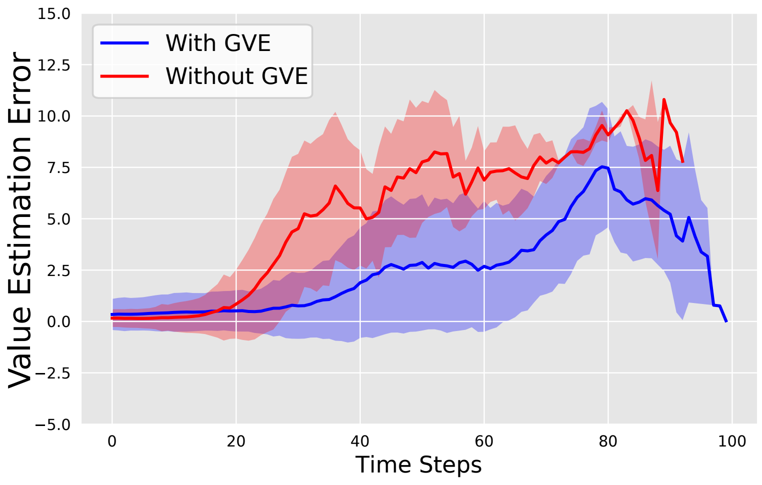


Figure 4.3: Value estimation error on test set. Our model has more accurate estimates of the state value function (lower error) which leads to better performance.

architecture can be seen in figure 4.2

Our prior knowledge graph is pre-trained on Visual Genome dataset [53]. The edge weights are set to one where the objects co-occur in a scene at least three times, and zero otherwise. There are 89 nodes in each channel of the graph and 5 channels in total; 4 layers dedicated to the objects in each room type and one self-connections layer for regularisation. We train a two layer adjacency matrix using GTN. The input to the graph, as node features, is a 1024-dimensional vector. This vector is the concatenation of 512-dimensional observation features, extracted using ResNet-18, with 512-dimensional Glove [42] embedding of the object name. The Glove embedding is mapped from 300 to 512 using a fully connected layer. The GTN features are mapped into a 512-dimensional vector using a fully connected layer.

Method	**SPL**	**Success**	**SPL >5**	**Success >5**
Random	3.64	8.01	0.1	0.28
A3C	14.68 ±1.8	33.04 ±3.5	11.69 ±1.9	21.44 ±3.0
A3C+Graph [119]	15.47 ±1.1	35.13 ±1.3	11.37 ±1.6	22.25 ±2.7
A3C+MAML(SAVN) [115]	16.15 ±0.5	40.86 ±1.2	13.91 ±0.5	28.70 ±1.5
A3C+Graph+GVE-ours	16.02	38.22	13.23	27.47
A3C+Graph+MAML-SS-ours (w/o. GVE)	15.13	38.8	13.68	29.64
A3C+Graph+MAML-Action-ours (w/o. GVE)	13.88	43.3	12.93	33.53
A3C+Graph+MAML+GVE-ours	**17.27** ±0.3	**43.8** ±1.1	**15.39** ±0.2	**33.68** ±0.9

Table 4.1: Quantitative comparison of our results with the baselines. Our approach improves all the baselines in all the four evaluation metrics, conventionally used in the previous state-of-the-art methods.

In order to train our model, we use Pytorch framework. We use SGD optimizer for meta-train weight updates and Adam [47] for the meta-test. As for the reward, we use a constant value of 5 for reaching the target and a step penalty of -0.01 for each single step. The maximum number of steps is capped at 50 during training and 200 during testing. We train all our methods until convergence with the maximum seven million episodes, whichever occurs first.

4.3.2 Baseline and SOTA Comparison

In order to better show the contribution our method we first compare it with a few different state-of-the-art (SOTA) and baselines, shown in Table 4.1. Firstly, we compare with the previous SOTA introduced by Wortsman et. al. [115], abbreviated as **A3C+MAML**. It uses MAML [27] to learn a loss function approximated by an instance of Temporal Convolutional Networks [8], over training episodes. Secondly, a related work [119] uses a fixed knowledge graph structure to encode object relationships as part of the state space, abbreviated as **A3C+Graph**.

In our approach, we benefit from the graph information more efficiently using our proposed GVE module. In addition to that, our graph structure and embedding architecture are also different, enabling a more accurate estimation of the value function.

Furthermore, to highlight the challenging navigation scenarios that our method is tackling, we compare our results with some trivial baselines. One simple solution is a **Random** agent for which the policy is to uniformly sample an action at all times. Another trivial baseline is shown as **A3C** in Table 4.1. This method is the result of dismantling our GVE and adaptation modules from the overall framework. Therefore, it acts as the simplest RL-based agent that shares the same backbone model with our proposed method.

4.3.3 Results and Ablation Studies

In this section we seek to answer a few principal questions with regards to our proposed method that shed more light on its strengths as well as weaknesses.

- *How does GVE improve value estimation error?*

In order to respond to this question, we analyse the estimated values over the trajectories

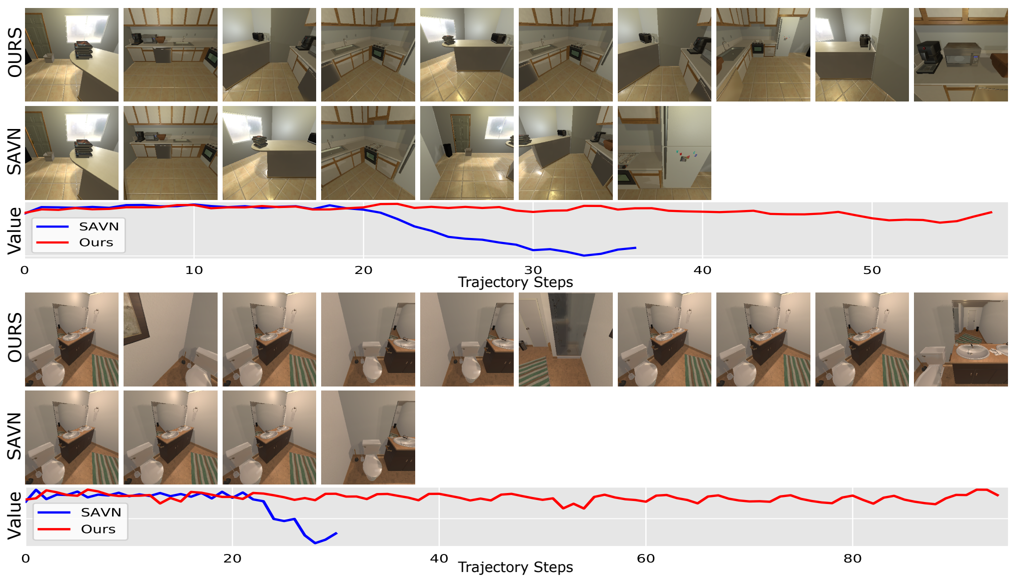


Figure 4.4: Qualitative comparison of two sample trajectories between our method and the baseline. ***Top****: target object: "bowl" in the kitchen. Our agent observes the bench-top and continuously predicts a high value (being optimistic) until finding the target. The baseline agent, however, has a less accurate understanding of the scene and gives up after a while not detecting the target.* ***Bottom****: target object: "soap bottle" in the bathroom. Our agent observes the basin and constantly predicts a high state value even though the soap bottle is small to be detected in the visual features. The baseline gives up after a few failed actions of moving forward.*

using our GVE in comparison to the baseline method. The estimation error over 1000 trajectories collected in the test scenes (unseen during training) is provided in Figure 4.3. The figure shows average and standard deviation of L2 distance per time step between the ground-truth value and the predictions. The red line shows the error for the baseline while the blue line shows the improvement as a results of our GVE. Our method effectively addresses the forgetting problem associated with LSTMs by estimating the values for longer trajectories more accurately.

- *Is GVE the optimal way to incorporate prior knowledge?*

In order to show the effectiveness of our graph-based value estimation, we compare our final method with two variants. First variant, termed as **A3C+Graph+MAML-SS** is to simply add our graph as part of the state space observation to the backbone network. As can be seen in Table 4.1 this method improves SR but not the SPL compared to **A3C+Graph**; however, compared to **A3C+MAML** in degrades the performance. We conjugate that the reason is by adding the complex graph to the backbone network, the

Table 4.2: Ours-LG *is the variant of our model where the image features are removed from the graph node features. We can see the graph is highly reliant on the observations to learn the relationships.*

Method	SPL	Success
Ours-LG	12.84	42.4
Ours-GVE	17.27	43.8

Table 4.3: Ours-RandomGraph *is the variant of our model where the edge weights are initialised randomly. The results on this table show the contribution of prior knowledge in our framework.*

Method	SPL	Success
Ours-RandomGraph	15.32	40.5
Ours-GVE	17.27	43.8

size of the state space is increased by orders of magnitude which impedes learning an effective state encoding under sparse reward constraint.

In another variant, termed as **A3C+Graph+MAML-Action**, we directly incorporate the graph into the policy for better decision making. Therefore, the policy will take actions conditioned upon the graph encoding. This approach can be observed as a weighted ensemble of policies representing different distributions. This leads to an improvement on SR but a significant degradation of SPL. We hypothesise that this can help the policy on better detecting the target object and thus on-time stopping which increases the SR; however, it cannot help with the optimality of the final policy. This further proves the effectiveness of our proposed GVE.

- What is the contribution of prior knowledge for value estimation?

In order to show the significant contribution of prior knowledge in our current framework, we train our final model with randomly initialised edge weights in the graph. As can be seen in 4.3, adding the graph with random edge weights does not improve the results compared to our baseline methods. This further confirms that our GVE module is reliant on prior knowledge for more accurate value estimation.

- *What is the quantitative improvement with respect to the baselines and previous state-of-the-art methods?*

As can be seen in Table 4.1, our final model **A3C+Graph+MAML+GVE** improves the previous SOTA **A3C+MAML** by almost 3% on SR and more than 1% on SPL. This

Table 4.4: Detailed comparison with the baseline method; SPL/Success rate are reported per room type. We can see that our method is general enough that improves the performance in 3/4 of the room types, with marginal performance on 1/4.

Method	**Bathroom**	**Bedroom**	**Kitchen**	**Living**
A3C+MAML	28.49/69.6	**8.65/29.2**	17.8/43.6	7.71/21.6
Ours-GVE	**31.03/75.6**	8.06/27.6	**17.93/45.6**	**9.41/25.2**

shows that our GVE helps the agent to find more targets in a smaller number of steps, as a result of having a more optimal policy. Our method is, particularly, more effective on longer trajectories where it improves the baseline by almost 5% on SR and almost 2% on SPL. This, again, confirms the effectiveness of our GVE which can provide enough global context disentangled from the length of the trajectory.

- *Is providing visual features to the graph nodes helpful?*

In order to show the integrity of the current design, we show the effect of removing observation (egocentric image) features from the node feature matrix. Thus, the graph will reduce to a fixed correlation among the objects, given in language embedding only (hence the name Language Graph, LG). It can also be observed as a sub-network for the value estimation to store value decomposition information without considering the observational correlations. As can be seen in Table 4.2, the performance of the model **Ours-LG** is significantly lower, particularly in SPL. This shows that the graph's contribution is not due to simply adding more trainable parameters.

- *How is our more optimal policy performing qualitatively?*

Finally, we provide qualitative comparison of the performance of our method compared to previous SOTA , *e.g.* **A3C+MAML** [115]. As is shown in Figure 4.4 the agent is navigating towards an instance of "microwave" in a kitchen scene. Our agent is able to find the target while constantly predicting a high value for the states due to observing the related objects like the "benchtop". The baseline, however, is after a few steps to predict low state values and *gives up* without reaching to the target.

For a more detailed comparison, we also provide performance results per room type, in Table 4.4. Additional trajectory visualisations can be found in the supplementary material.

4.4 Conclusion: Method Summary

In this chapter, we present a simple yet effective method to improve the performance of the actor-critic RL algorithm for visual navigation. Our method improves the state value function estimation accuracy by incorporating object co-location information in the form of a neural graph. Actor-critic RL is heavily reliant on the accuracy of the state value estimation to converge to the optimal policy. This is mainly because accurate value estimation helps with the correct identification of the actions along the trajectory that contribute the most to the final reward (i.e. credit assignment). Through extensive empirical studies, we show that our agent is realistically more optimistic (*e.g.* accurately predicts higher state values). This leads to successful navigation towards the target object when the baseline agents usually give up.

Chapter 5

Imaginative Reinforcement Learning

Imagination is, arguably, one of the most unique aspects of human intelligence. Humans can *imagine* the outcomes of different actions to help make the best possible decision. Such future prediction is often done in an abstract, i.e. latent, space, to help concentrate on the most seminal aspects of the future. In this chapter, we propose a novel method to enable imagination for visual navigation. Similar to the previous chapter, we target actor-critic RL algorithm. Our method endows an agent with the explicit prediction ability of a latent state it is required to reach before successful completion of the task, i.e. a (sub-)goal state. Such ability improves the robustness and efficiency of navigation without the need for complex planning algorithms or accurate stepwise future prediction.

5.1 Previous Methods and Research Question

Object-goal visual navigation is the task where an agent needs to navigate towards an instance of a given target object in a previously unseen environment. This is an inherently challenging problem since the agent needs to find the shortest possible sequence of actions aiming to first find a target object (i.e. to explore intelligently) and then navigate towards it (i.e. to plan and act accordingly while avoiding obstacles) in a visually complex 3D environment [20, 52, 89, 112, 117]. The task is fundamentally challenging since the agent does not normally have access to the state and only receives (visual) partial observations of the environment at each time step. Imagine yourself as the agent shown in Figure 5.1 entering a new previously unseen kitchen and trying to find a toaster. You will need to look around, while potentially moving to increase your observation area, localise yourself within the new scene, and navigate towards the toaster once you have localised it. In deep RL terminology, this means the agent will need to learn simultaneously to (1) build a good state representation that enables the agent to localise itself as well as the target (if observed at some point) and (2) efficiently use that state representation to (implicitly) plan and navigate towards the target while avoiding the obstacles. In a navigation episode, from every starting point, there are typically multiple action sequences (i.e. trajectories) that could lead to success, let alone the sequences that might fail. Learning to select the right action at each time step to create a trajectory that leads to the specified target object is the primary challenge.

To tackle this problem, model-free RL algorithms are typically used to train a single policy to perform all the aforementioned tasks implicitly by mapping the input visual observations into the actions [13, 23, 24, 69, 115]. Explicitly incorporating the transition in the environment to predict the potential outcome of the actions, hailed model-based RL, is also developed on other tasks and environments [33, 34, 36, 45, 122]. Those methods, however, are generally harder to train, especially in a 3D-rich environment, since every state transition in the environment has to be accurately modelled. Therefore, their successful application has been limited to much simpler benchmarks such as Atari [36, 45, 91], Deepmind Control Suite [34, 35] or robotic arm tasks [73, 81, 123]; unlike those problems, in visual navigation, the testing environment is unseen, visually much more complex and at times significantly different from the observed training environments.

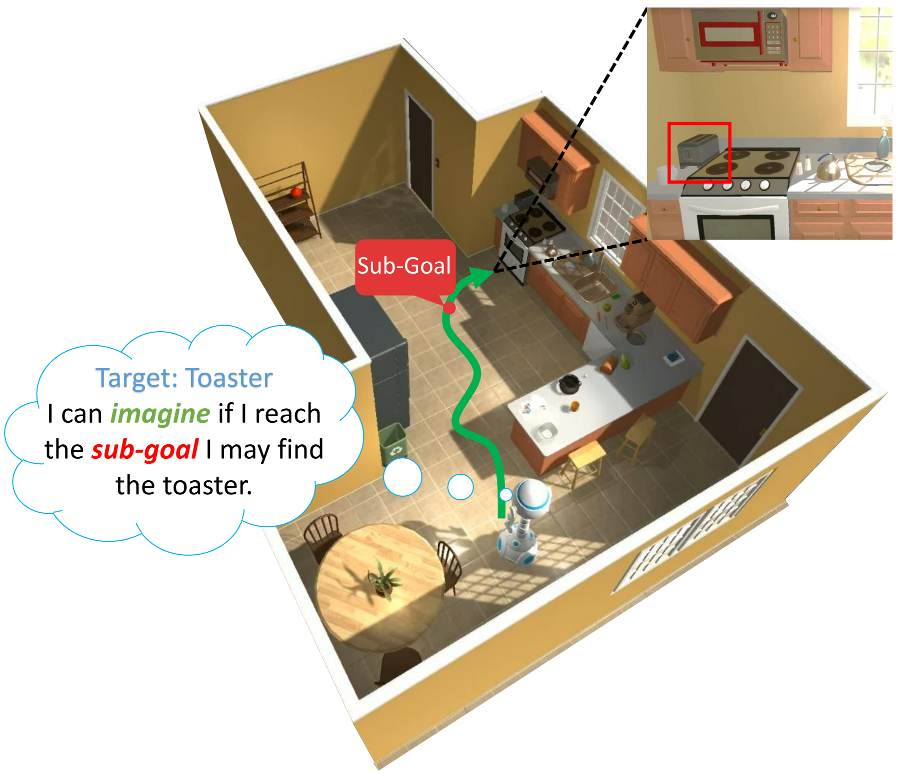

Sub-Goal
Target: Toaster
I can *imagine* if I reach the *sub-goal* I may find the toaster.

Figure 5.1: Enabling an agent to imagine, i.e., to predict, states on the path to success improves its ability to carry out complex tasks, particularly in unseen environments. As opposed to the conventional approaches, our ForeSI agent takes actions not only based on the current state, but also a prediction of a successful future, to achieve its goal.

To mitigate the above mentioned issues we propose our *Foresight Sub-goal Imaginator* (ForeSI) agent.

Intuitively, as shown in Figure 5.1, ForeSI enables the agent to have a foresight of a future sub-goal state through which it is more likely to successfully achieve its goal. By explicitly incorporating the sub-goal information into the policy, the agent can take better actions even when the target is not in the field of view of it.

In particular, ForeSI helps with the navigation in two main ways: firstly, it provides the agent with an imagined (i.e. predicted) representation of a sub-goal state that will help with successful task completion, i.e. stopping at the right location; our empirical results in the absence of an object detector support this hypothesis; secondly, it helps the agent to constantly remember the sub-goal state to navigate to even if that state is temporarily out of the field-of-view of it; i.e. it improves the navigation efficiency. In summary, in this work, our primary contributions are as follows:

1. We propose a method to enable visual navigation agents to generate foresight of the states on the path to success, i.e. sub-goal states, conditioned on the first state and the target object.
2. We propose an attention mechanism for state-value estimation that helps to identify the sub-goal state used to train our sub-goal generation module.
3. We propose an algorithm to efficiently integrate our method into model-free RL which re-utilises the past trajectories in hindsight via incorporating a replay buffer; thus ForeSI does not require extra data collection.
4. We show the effectiveness of our method in improving visual navigation performance when added to multiple different baselines with different perception methods and environment setups on AI2THOR [52].

5.1.1 Overview: Imaginative Visual Navigation Framework

Having a single policy with which to achieve various visual navigation tasks based solely on a target object's name is not trivial.

The aim of our *Foresight Sub-goal Imaginator* (ForeSI) agent (see Figure 5.2) is to enable predicting (i.e., imagining) a sub-goal state through which the agent has maximal

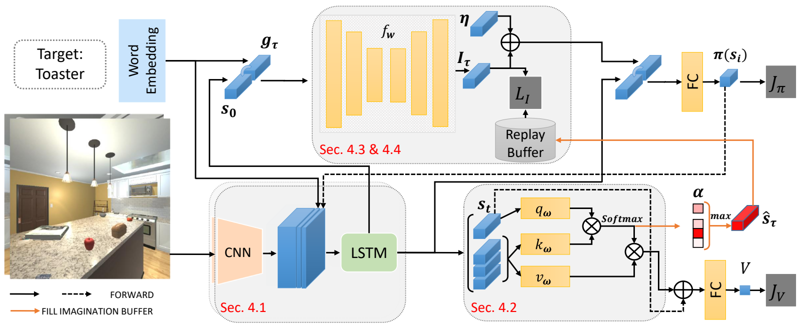


Figure 5.2: We develop an attention mechanism to identify the sub-goal state that minimises the critic error (in hindsight, at the end of an episode) and trains our sub-goal generator to predict that state (in foresight, at the beginning of the episode). Our training algorithm efficiently integrates ForeSI into actor-critic RL while avoiding extra data collection.

chances of success. Incorporating such a sub-goal in the policy enables the agent to take better actions since it has an indication of what states are important to visit in the future and eliminates a need for unnecessary exploration at test time. There are two main challenges to overcome: (1) to determine the sub-goal state to be learned by reviewing the observed trajectory in hindsight, and (2) to imagine or generate that sub-goal given only the initial state and target object in an unseen environment. Note that (1) happens in hindsight at the end of an episode while (2) happens in foresight at the very beginning of an episode.

In determining the sub-goal state to be learned, we note that the sub-goal most valuably imagined is that with the highest impact towards receiving the maximum reward. In other words, the optimal sub-goal is the one with which we could learn the best value function to estimate the expected reward. Intuitively, when the agent reaches the optimal sub-goal, finding the target and receiving the maximum reward should be easy (see Section 5.2.1).

Once we find the optimal sub-goal, we can train a model to predict or *imagine* an instance for an unseen environment. To that end, we collect a replay buffer of the *successful* trajectories to train our sub-goal generation module. Intuitively, when an agent navigates through particular states to achieve its goal, imagining the sub-goal from a related successful trajectory helps the agent to identify how to plan and take actions in unseen environments (see Section 5.2.2). This imagination is conditioned on the initial state and the target object and is integrated into a parallel execution of the policy in Asynchronous Advantage Actor-Critic (A3C) [67] for best performance. This allows the agent to learn to both imagine and navigate (see Section 5.2.3).

5.1.2 Notations

Formally, a trajectory τ of length $T{+}1$ consists of a tuple $(\mathrm{s}_0, a_0, r_0; \mathrm{s}_1, a_1, r_1; ..., \mathrm{s}_T, a_T, r_T)$ that is generated by taking action a_t at time t and observing the next state according to the dynamics of the environment $\mathrm{s}_{t+1} \sim p(\mathrm{s}_{t+1} \,|\, \mathrm{s}_t, a_t)$. A reward $r_t = r(\mathrm{s}_t, a_t, \mathrm{s}_{t+1})$ is received from the environment at each time step, which is conventionally [23, 24, 115, 119, 125] defined as a large positive number for a successful trajectory and a negative step penalty otherwise, to encourage success in lowest possible number of actions.

A common approach is to use actor-critic RL methods for learning the optimal policy [23, 115, 119, 125] for navigation $\pi_\theta(a_t \,|\, \mathrm{s}_t, \mathrm{g}_\tau, \boldsymbol{\theta})$ to choose action a_t at time t conditioned on an embedding of the target object's name g_τ (e.g. using GloVe embedding [42]) for a given environment (i.e. a room in AI2Thor) E. We use $\boldsymbol{\theta}$ to denote the set of all the parameters of the RL policy. Learning involves minimising $\mathcal{J}_\pi(a_t \,|\, \mathrm{s}_t, \boldsymbol{\theta})$ the negative of expected advantage function while minimising the difference of the estimated value function and the true return $\mathcal{J}_V(\mathrm{s}_t, \boldsymbol{\theta})$, where we have:

$$\begin{aligned}\mathcal{J}_\pi(a_t \,|\, \mathrm{s}_t, \boldsymbol{\theta}) &= -\log \pi(a_t \,|\, \mathrm{s}_t, \mathrm{g}_\tau; \boldsymbol{\theta})(r_t +, \\ &\qquad \gamma V_\theta(\mathrm{s}_{t+1}) - V_\theta(\mathrm{s}_t)) + \beta_H H_t(\pi) \qquad (5.1)\\ \mathcal{J}_V(\mathrm{s}_t, \boldsymbol{\theta}) &= \frac{1}{2}(V_\theta(\mathrm{s}_t) - R)^2, \qquad (5.2)\end{aligned}$$

and $R = \mathbb{E}_{\tau\sim\pi}[\sum_{i=t}^{T} \gamma^{t-i} r_i \,|\, \mathrm{s}_t]$. Here, H_t is the entropy of the policy that acts as a regulariser with β_H as its hyper-parameter.

For a detailed discussion on the mathematical framework (i.e. POMDP) and object-goal visual navigation refer to Chapter 2.

5.2 Proposed Method

In this section, we provide the details of our novel imaginative RL-based visual navigation framework.

5.2.1 Hindsight: Sub-Goal Identification

In the first step, we consider learning to identify the sub-goal state through which a navigation episode is successful. To that end, the agent has to consider its current state and the sub-goals it has navigated through with their corresponding visual, target object and dynamics representation. Intuitively, for the agent to obtain a high reward in a straight path through a hallway, for instance, it should have chosen the best action while at the previous corner to be successful. Since the only aspect of the RL that considers the future reward is the value function, we modify Eq. (5.1) using a residual function of the past states as:

$$V_\theta(\mathrm{s}_t) \approx V_\theta(\mathrm{s}_t^\star), \qquad \mathrm{s}_t^\star = \sum_{j=0}^{t} \alpha_j v_\omega(\mathrm{s}_j) + \mathrm{s}_t\,. \tag{5.3}$$

Here, $v_\omega(\mathrm{s}_j)$ is a linear function of the input and α_j is the jth dimension of $\boldsymbol{\alpha}$, which is defined as follows:

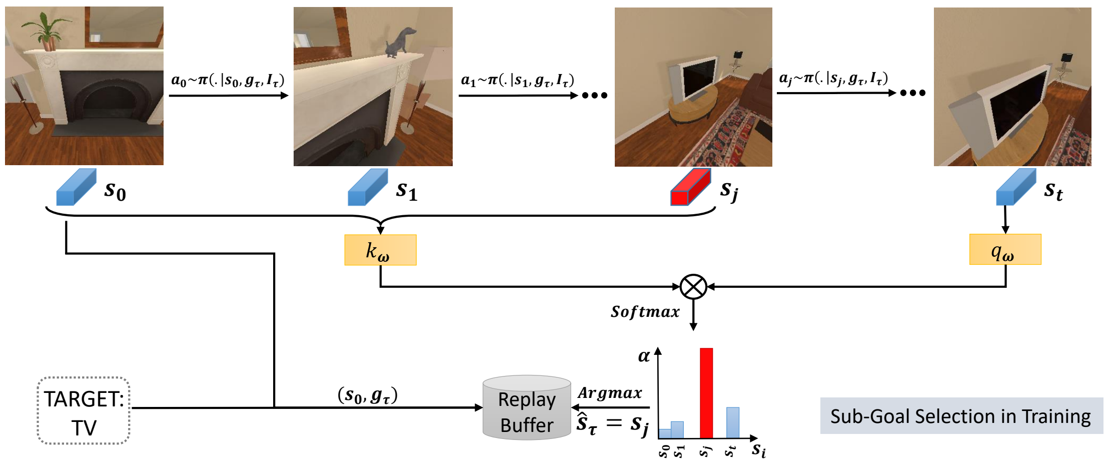
$$\boldsymbol{\alpha} = \text{softmax}\left(\frac{q_\omega(\mathrm{s}_t)k_\omega([\mathrm{s}_0 : \mathrm{s}_t])^\top}{\sqrt{t+1}}\right). \tag{5.4}$$

Here, q_ω and k_ω are linear functions analogous to the query and key in an attention mechanism [104] with $\mathrm{s}_{0:t}$ the concatenation of the states up to time t. We denote all of our sub-goal selection parameters by the set $\boldsymbol{\omega}$. Moreover, α_j is the correlation between state j and the current state t, and its magnitude specifies the likelihood that state j is an important sub-goal to reach.

Effectively, we use the attention mechanism described above to identify the sub-goal state that minimises the state value function estimation error.

Using the key-query product above we choose the state in the past that is most related to the current state. Interestingly, if the state is novel (i.e., uncorrelated with the past states) we allow the agent to assign a high attention weight to its most recent state.

We select the state with the maximum attention weight as the sub-goal state at the end of an episode, i.e., $\hat{\mathrm{s}}_\tau = \mathrm{s}_{t^*}$ where $t^* = \arg\max_t \alpha_t$ (note $\boldsymbol{\alpha}$ has length $T+1$). We



$a_0 \sim \pi(.|s_0, g_\tau, I_\tau)$
$a_1 \sim \pi(.|s_1, g_\tau, I_\tau)$
$a_j \sim \pi(.|s_j, g_\tau, I_\tau)$
s_0
s_1
s_j
s_t
k_ω
q_ω
Softmax
α
TARGET: TV
(s_0, g_τ)
Replay Buffer
Argmax
$\hat{s}_\tau = s_j$
s_i
Sub-Goal Selection in Training

Figure 5.3: Overview of our sub-goal identification method showing how our agent selects a sub-goal state to fill the replay buffer during the training; the replay buffer is then used to train our sub-goal generation module.

subsequently task ForeSI to learn to generate this sub-goal. Using this method we ensure that the imagination will guide the policy towards a state that has the highest correlation to a successful goal state. Additionally, allowing for the sub-goal state to be different from the previously achieved goal state helps avoid repeating the potentially sub-optimal trajectories. The sub-goal identification process is visualised in more details in Figure 5.3. Note that in our sub-goal selection mechanism we add position embedding to each state representation s_i to maintain the order of the observed states in the sub-goal selection, inspired by [104].

5.2.2 Foresight: Sub-Goal Generation

For learning to imagine, or generate, the selected sub-goal state $\hat{\mathrm{s}}_\tau$ we consider a replay buffer. The replay buffer, denoted by M is filled with tuples of $(\mathrm{s}_0, \mathrm{g}_\tau, \hat{\mathrm{s}}_\tau)$, an initial state, an embedding representation of the target object g_τ for trajectory τ and the sub-goal. We then devise the following objective to train our sub-goal generation function $f_{\mathbf{w}}$:

$$\min_{\mathbf{w}} \quad \mathbb{E}_{(\mathrm{s}_0, \mathrm{g}_\tau, \hat{\mathrm{s}}_\tau)\sim M} \left| \hat{\mathrm{s}}_t \; - \; f_{\mathbf{w}}([\mathrm{s}_0 : \mathrm{g}_\tau]) \right|, \tag{5.5}$$

where $\mathbf{w}$ is the set of parameters of our sub-goal generation module and $[:]$ denotes the concatenation of vectors. For $f_{\mathbf{w}}$ we use a multi-layer perceptron and a bottleneck with the intuition that the structure of the sub-goal distributions lies in a lower dimensional space. Intuitively, the imagination module does not need to generate every single dimension of the state accurately, since there might be unnecessary information about the other objects or the background scene in the representation. It might also include dynamic objects that constantly change location and/ or appearance across different environments. Using a smooth version of an L1 loss, we avoid penalising such inexact predictions too harshly.

We use a shared recurrent state encoding for both the policy and the sub-goal generation module. Therefore, the generated sub-goal state representation not only has information about the sub-goal's visual appearance but also encodes the history of the past observations and actions before that state. Hence, we essentially generate a representation of the whole trajectory that leads the agent to a successful goal state. Furthermore, sharing the state encoder that is trained along with the policy helps generate sub-goal states that are directly useful for the action selection. This is as compared to methods such as [33] where the agent needs to first perform a random policy search to collect a dataset from the environment, and then use only the visual features for generating future states.

Since computing the expectation in Eq. (5.5) is impractical due to the buffer size and the constant change in the distribution of states, we only consider the latest collected tuples to update the model's parameters. That is, we update the model when $|M| = M_{\max}$ and empty the buffer.

5.2.3 Integration into Actor-Critic RL

To use ForeSI we condition the policy on the imagined sub-goal state. This way the policy learns to take actions based on the current state and the imagined one to achieve its goal.

One potential issue with this approach is that it could bias the policy towards exploiting the known imagined sub-goal states and avoiding essential exploration, hence degrading the performance. We address this issue by adding Gaussian noise to the imagined states. Therefore, in practice the generated sub-goal is as follows:

$$\begin{aligned} a_t &\sim \pi_\theta(a_t \,|\, \mathrm{s}_t, \mathbf{g}_\tau, \mathbf{I}_\tau), \quad \text{and} \\ \mathbf{I}_\tau &= f_{\mathrm{w}}([\mathrm{s}_0 : \mathbf{g}_\tau]) + \boldsymbol{\eta}, \quad \boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I}). \end{aligned} \tag{5.6}$$

Here, $\mathbf{0}$ is a vector of all zeros, $\boldsymbol{I}$ is the identity matrix, and σ^2 is the variance of the noise. The additive noise is decayed during training to allow more exploration when the agent is more unsuccessful (i.e., the imagination is not robust enough) and exploit otherwise. We choose the noise level by adjusting the variance, that is,

$$\sigma^2 = \max(\sigma^2_{\max} - \rho, 0), \tag{5.7}$$

where $\sigma^2_{\max}$ is a pre-defined maximum variance threshold, and ρ is a moving average of the success rate over the past episodes. This simple heuristic ensures the noise level is proportionate to the success rate, for instance, if $\rho = 0.9$ and the success rate is around 90% we completely remove the added noise.

Finally, we integrate the sub-goal selection and the imagination into the on-policy Asynchronous Advantage Actor-Critic (A3C) [67] algorithm which allows for efficient and parallel training of multiple agents. We revise the training objectives to integrate ForeSI into A3C:

$$\mathcal{J}^\star_\pi(a_t \,|\, \mathrm{s}_t, \boldsymbol{\theta}) = -\log \pi(a_t \,|\, \mathrm{s}_t, \mathbf{g}_\tau, \mathbf{I}_\tau; \boldsymbol{\theta})(r_t + \gamma V_\theta(\mathrm{s}^\star_{t+1}) - V_\theta(\mathrm{s}^\star_t)) + \beta_H H_t(\pi), \tag{5.8}$$

$$\mathcal{J}^\star_V(\mathrm{s}_t, \boldsymbol{\theta}) = \frac{1}{2}(V_\theta(\mathrm{s}^\star_t) - R)^2. \tag{5.9}$$

We summarise the training of our approach in Algorithm 1. As may be observed, in the spirit of A3C, each agent stores its own replay buffer and computes its noise level hyper-parameter proportionate to its moving average success rate.

Algorithm 1: Training One ForeSI Agent

```
Randomly initialise $\boldsymbol{\theta}, \mathbf{w}, \boldsymbol{\omega}$
Initialise replay buffer $M = \varnothing$
while episode < MAX_EPISODE do
    $(\mathrm{s}_0, g_\mathcal{T}) \sim E_{\mathrm{RND}}$                    ▷ $E_{\mathrm{RND}}$ is a random environment
    $\mathbf{I}_\mathcal{T} = f_\mathbf{w}([\mathrm{s}_0 : \mathbf{g}_\mathcal{T}] + \boldsymbol{\eta},\ \boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I})$          ▷ Eq. (5.6)
    while $a_t \neq$ STOP $\&$ $t \leq T$ do
        $a_t \sim \pi_\theta(a \,|\, \mathrm{s}_t, \mathbf{g}_\mathcal{T}, \mathbf{I}_\mathcal{T})$
    end
    Compute $\boldsymbol{\alpha}$ using eq. (5.4) for the trajectory
    Update $\boldsymbol{\theta}$ and $\boldsymbol{\omega}$ via eq. (5.1) and (5.9)
    if trajectory is successful in the environment then
        $\hat{\mathrm{s}}_\mathcal{T} = \mathrm{s}_{t^*},\ t^* = \arg\max_t \alpha_t$
        $M = M \cup \{\mathrm{s}_0, \mathbf{g}_\mathcal{T}, \hat{\mathrm{s}}_\mathcal{T}\}$                     ▷ Update Buffer
    Update the average success rate $\rho$
    if $|M| = M_{\max}$ then
        for epoch $\leq$ Epoch$_{\max}$ do
            for $\{\mathrm{s}_0, \mathbf{g}_\mathcal{T}, \hat{\mathrm{s}}_\mathcal{T}\} \in M$ do
                $\boldsymbol{\omega} \leftarrow \boldsymbol{\omega} - \beta \nabla_\mathbf{w} |\hat{\mathrm{s}}_t - f_\mathbf{w}([\mathrm{s}_0 : \mathbf{g}_\mathcal{T}])|$
            end
        end
        $M = \varnothing$
    Update $\sigma^2$                                                    ▷ Eq. (5.7)
end
```

5.3 Experiments

In this section, we present implementation details of our method as well as extensive experiments to show how ForeSI improves the visual navigation performance of multiple significantly different baselines.

We use the AI2Thor [52] environment to benchmark our results. The simulator consists of photo-realistic indoor environments (i.e., houses) categorised into four different room types: kitchen, bedroom, bathroom and living room. We run our experiments on two distinct setups of this simulator for fair comparison against previous state-of-the-art approaches. In both of those setups 20 different scene layouts of each room type are used for training; 5 scenes for validation and 5 for the test. We provide the final results on the test set based on the best performing model on the validation set. For a fair comparison, we follow the exact object configuration and target object list according to the baseline

Method	**SPL**	**SR**	**SPL>5**	**SR>5**
		First Setup		
A3C [115]	14.68	33.04	11.69	21.44
A3C+MAML [115]	16.15 ± 0.5	40.86 ± 1.2	13.91 ± 0.5	28.70 ± 1.5
A3C+ForeSI	**15.23** ± 0.4	**36.80** ± 1.1	**13.14** ± 0.3	**27.55** ± 1.4
A3C+MAML+ForeSI	**16.75** ± 0.5	**45.5** ± 1.0	**15.8** ± 0.6	**34.7** ± 1.1
		Second Setup		
A3C+ORG [23]	37.5	65.3	36.1	54.8
A3C+ORG+ForeSI	**39.41** ± 0.3	**68.0** ± 0.6	**36.85** ± 0.4	**56.11** ± 0.8

Table 5.1: A quantitative comparison of our method against four representative baselines. The baseline methods are significantly different in perception, policy modelling and environment setup. ForeSI improves the performance of all of them including the previous state-of-the-arts [23, 115]. SPL>5 and SR>5 show the metrics for trajectories longer than 5 steps.

methods.

In the first setup, we follow the configuration as used in [115, 119]. In this setup the target object is selected from the following list: pillow, laptop, television, garbage can, box, bowl, toaster, microwave, fridge, coffee maker, garbage can, plant, lamp, book, alarm clock, sink, toilet paper, soap bottle and light switch. Here a trajectory is considered successful if the agent stops within **1 meter** circular proximity of the target object while the object is visible in the ego-centric view of the agent.

In the second setup, we follow the configuration recently introduced in [23]. The distribution of the target objects and the object location configuration in this setup is significantly different. The following objects are added to the list of the targets compared to the first setup: cellphone, chair, desk lamp, floor lamp, kettle, pan, plate, pot, remote control, stove burner; moreover, the following objects are removed from the list in the first setup: pillow, box, plant, lamp, toilet paper, soap bottle. The difference in the target objects can also affect the navigation performance due to differences in size, distant visibility etc. Lastly, here the successful trajectory criterion is relaxed to a **1.5 meters** circular distance around the target object.

Following the recent conventions in visual navigation tasks [5, 23, 115, 119] we evaluate the performance of our method using two main metrics: Success Rate (SR) and Success weighted by inverse Path Length (SPL), calculated as $\sum s_i \frac{L_i^g}{L_i}$, where L_i^g is the ground truth shortest path to the target, L_i is the length of the trajectory as taken by the agent and s_i is the binary success indicator.

5.3.1 Implementation Details

We use A3C [67] as the basis of our method. This actor-critic algorithm is a good fit for visual navigation tasks since the agents can explore in parallel and asynchronously, which is more computationally efficient. Our method can potentially be integrated into other actor-critic algorithms since it only relies on the critic to identify the sub-goals during the training.

Our backbone model comprises a single-layer LSTM state encoder with 512 hidden states. The input to the encoder at each time step is the visual features extracted from a pre-trained ResNet-18 [39] and the Glove embedding [42] of the target object, as visualised in Figure 5.2. Note the Glove embedding in the second setup is replaced with a one-hot vector embedding following the settings in [23]. The policy comprises of a single fully-connected layer that outputs the distribution over 6 actions, *{RotateLeft, RotateRight, MoveAhead, LookDown, LookUp, Stop}*, using a Softmax activation function. The state-value head is a two-layer MLP that maps the attended state representation to a single scalar value. We use a reward of 5 for task completion and a negative step penalty of -0.01 for each taken action. We implement the attention mechanism as three fully connected layers of size 512 that map the query, key and value.

Our ForeSI's f_{w} comprises 6 fully connected layers that receive the concatenation of the state representation and the target object embedding as input. We use a $\tanh$ non-linearity in all layers to mimic the behaviour of the LSTM state encoder.

We use a replay buffer of size 32 for each agent to train its sub-goal generation module separately. We, then, transfer the learnt weights to a shared model between the agents, asynchronously, and we empty the replay buffer. For the additive noise in sub-goal generation (see Section 5.2.3), we use $\sigma^2_{max} = 0.9$ and the last 100 episodes to compute the moving average for the success rate.

Method	Bathroom	Bedroom	Kitchen	Living
First Setup				
+MAML	**28.49**/69.6	8.65/**29.2**	17.8/43.6	7.71/21.6
+MAML+ForeSI	27.03/**73.6**	**8.81**/27.6	**21.55/54.0**	**9.61/26.8**
Second Setup				
+ORG	**49.87/83.89**	35.43/62.21	38.63/69.02	29.33/47.83
+ORG+ForeSI	47.44/80.4	**38.15/65.6**	**41.09/72.87**	**30.39/52.14**

Table 5.2: Detailed comparison against previous state-of-the-art methods on the two different setups; SPL/SR are reported per room type. Our method is general enough to improve performance in 3/4 of the room types, with a marginal performance impact on 1/4. Notably, we improve the SR of trajectories in the "kitchen" and "living room" by a large margin where the trajectories are generally longer and generating the sub-goal is more important.

5.3.2 Results

In Table 5.1 and Table 5.2 we compare our approach to that of state-of-the-art baselines in two separate simulator setups as discussed in Section 5.3. The first baseline, **A3C**, is only using the backbone model described in Section 5.3.1 trained using A3C RL algorithm. This is the main baseline that shows how a simple RL objective can perform without any extra components or modifications. **A3C+ForeSI** is the variant of A3C upgraded with our method. We see that empowering the agent with forward modelling improves the success rate by more than 3%. The improvement on longer-horizon tasks, i.e., longer trajectories, is even more significant, more than 6%. We conjecture that this is mainly because in longer trajectories, in the absence of ForeSI, the agent can more easily forget the objective by focusing more on short term tasks such as obstacle avoidance. Furthermore, in longer trajectories, it is, also, more likely to follow a path that does not lead to success from the very beginning when ForeSI is not used. Using ForeSI, however, helps the agent to constantly remember the (sub-)goal and, thus, take the actions that are more likely to achieve the goal.

Additionally, we, also, evaluate the performance of our approach when used along with a self-adaptive baseline, where we utilise meta-learning similar to **A3C+MAML** [115] for test-time policy adaptation. **A3C+MAML** shows a considerable performance boost over the simple **A3C** baseline as shown in Table 5.1. Despite that, when combined with our method in **A3C+MAML+ForeSI** we observe an additional absolute improvement of around 5% in success rate on both the short and long trajectories. This further demonstrates the modularity of our approach by showing that it improves both adaptive, and non-adaptive, baseline methods. Note that while the absolute improvement in both the short and long trajectories presents similar figures, the relative improvement is much larger for long trajectories. This, again, supports the previous hypothesis that our imagination can effectively help address both the LSTM forgetting problem and finding the correct path to the successful target. Furthermore, nearly 2% improvement compared to **A3C+MAML+ForeSI** on the SPL over the long trajectories implies that ForeSI helps avoid futile wandering around.

The third baseline that we compare our method against is **A3C+ORG** [23]. In that method the authors use an off-the-shelf object detector, i.e. FasterRCNN [83], and incorporate the detected object bounding boxes along with their confidence scores into the policy by building a neural graph, inspired by [119]. Their method builds a better state representation that renders finding target objects much easier for the agent. As long as the objects are detected by the object detector the agent can learn to directly navigate towards them taking a relatively short trajectory. The authors also modify the environment setup as discussed in Section 5.3. Therefore, we find that a suitable baseline to compare against to demonstrate the modularity of our approach. We build our approach using their method as the backbone which we call **A3C+ORG+ForeSI**. We observe that our method improves the success rate by over %2 compared to [23]. This shows that irrespective of the quality of the state representations empowering the agent with a forward model using our method can help improve the navigation performance. Note that we do not further compare our results with the addition of imitation learning into this problem, as the authors did in [23]. This is because, in this chapter, we are only concerned about integrating a forward model into model-free RL.

5.3.3 Ablation Studies

5.3.3.1 What to Imagine:

In our approach, we learn to generate the sub-goal state from the agent's own successful navigation episode; we could, however, consider alternative approaches that we compare here. Firstly, we seek to answer whether our ForeSI provides the agent with valuable information about its future sub-goal states.

In Table 5.3, we consider a random state generation by replacing the output of our

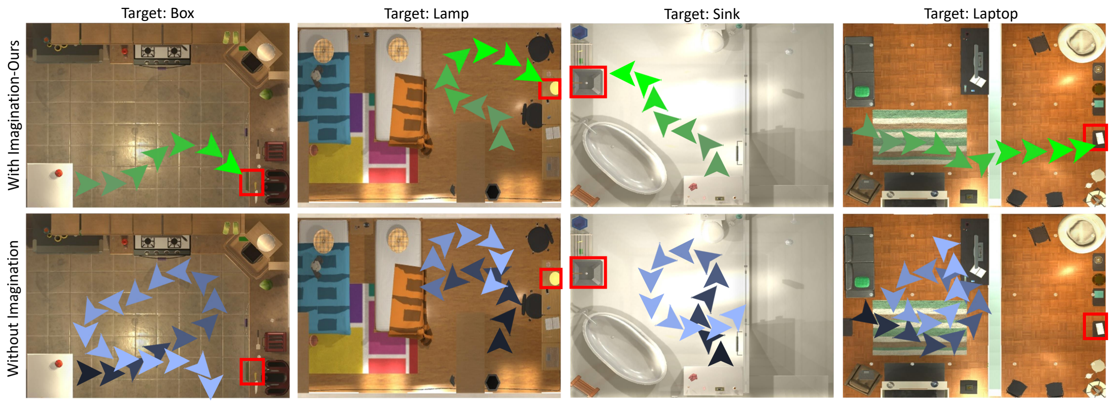


Figure 5.4: Qualitative comparison of navigation episodes of ForeSI against the baseline method [115] in four different room types. The examples represent cases where our method improves both the robustness and the efficiency of navigation.

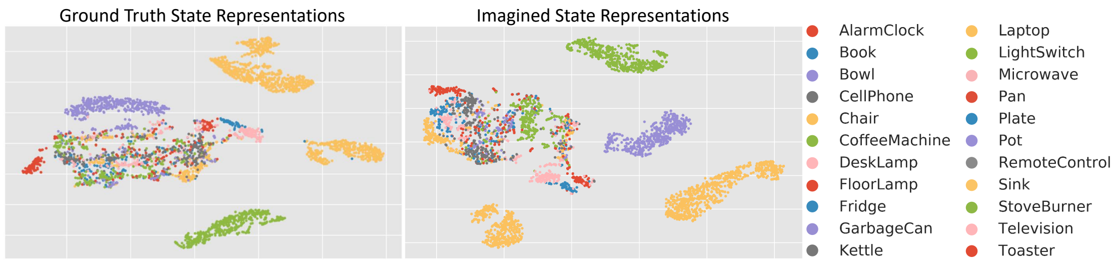


Figure 5.5: t-SNE comparison of the sub-goal states generated using our sub-goal method against the ground truth states. Our agent's predicted sub-goals closely follow the structure of the ground-truth and are clustered similarly.

sub-goal generation with $\tanh(\mathbf{s})$ where $\mathbf{s}$ is sampled from a Gaussian noise, i.e. $\mathbf{s} \sim \mathcal{N}(\mathbf{0}, 0.5\mathbf{I})$, shown in **Ours-RND**. The $\tanh$ non-linearity assures that the noise is similar in values to the states generated by our sub-goal generation, but are nonetheless rather random and meaningless. As observed, this leads to a deterioration in the success rate compared to the original baseline, **A3C+ORG**. We hypothesise that, because it is uninformative, the policy learns to dismiss the randomly generated state to some extent, but not completely, hence the slight performance degradation.

We further compare our approach with the case of predicting the weighted average of the states in Equation 5.3, rather than a single sub-goal. The intuition for this experiment is that knowing the attended states gives the agent valuable information about what important future states are expected to be traversed. As shown in **Ours-ATT** of Table 5.3 this leads to improved performance which indicates it is helpful to know about the sub-goal states ahead of the agent; however, since the sub-goal is not always identifiable (multiple sub-goals and trajectories could have the same attended states) the performance gain is rather insignificant.

5.3.3.2 Sub-Goal Generation Interval:

Rather than generating the sub-goal from the initial state, we can consider generating multiple sub-goals in fixed intervals every given number of steps. In that case, in each interval step, a different sub-goal is predicted. As shown in **Ours-INT** in Table 5.3, this leads to a performance degradation. We believe this is because the policy collapses to a deterministic one due to the constraints on predicting multiple sub-goals. Consequently, the agent stops exploring and converges to a sub-optimal solution.

Method	**SPL**	**SR**
A3C+ORG[23]	37.5	65.3
Ours-RND	37.57	64.8
Ours-INT	37.78	63.8
Ours-ATT	37.76	65.4
Ours-ForeSI	**38.66**	**67.6**

Table 5.3: Ablation study of different variants of our method. ForeSI avoids the unnecessary complications of multi-step future predictions. Our future state predictions are meaningful.

5.3.3.3 Explicitly Structured Sub-Goal Generation:

It can be seen in Figure 5.5 that ForeSI can accurately learn the structure in the state representations used by the agent's state encoder. To further investigate the quality of the generated states we train a Conditional Variational Auto-Encoder (C-VAE) [96] to learn to encode the state representations. Then, we use ForeSI to generate the latent state of the C-VAE rather than working directly on the agent's state representations. We compare the average loss achieved by our module trained directly on the agent's state representations with that of trained on the latent representations of the C-VAE. We observe that direct sub-goal generation achieves a lower average loss, 0.012 vs 0.018, while being significantly less complex.

5.3.4 Qualitative Comparison

In Figure 5.4 we present four sample navigation episodes where our method improves both the efficiency and robustness of the baseline [115]. We refer the reader to Table 5.1 for the overall test set results.

In Figure 5.6 we compare two sample trajectories between our method and the baseline method [115]. In Figure 5.6 (a) we observe that forward modelling helps our agent identify the target object while the baseline method disregards the observed target and stops at a random location after a few searching steps. While being eventually successful, here our agent misses the target once and returns to a similar state again. This might be due to the fact that our sub-goal selection method might not have observed the target in the very first few steps and thus can still be further improved. A similar scenario can be observed in Figure 5.6 (b), where the laptop is visible in the agent's observation but the agent has to take a few extra steps and return to the laptop again. This has been observed multiple other times in the test trajectories. Despite the improvement that our method presents in SPL 5.6 provides more insights into how we might further improve our sub-goal selection.

In Figure 5.7 we present two sample trajectories that show how our agent is able to take the shortest path to the target object. In contrary, the baseline method [115] either dismisses the target object and takes the wrong trajectory in Figure 5.7 (a), or cannot stop at the right location in Figure 5.7 (b) and fails. Those two samples show that our

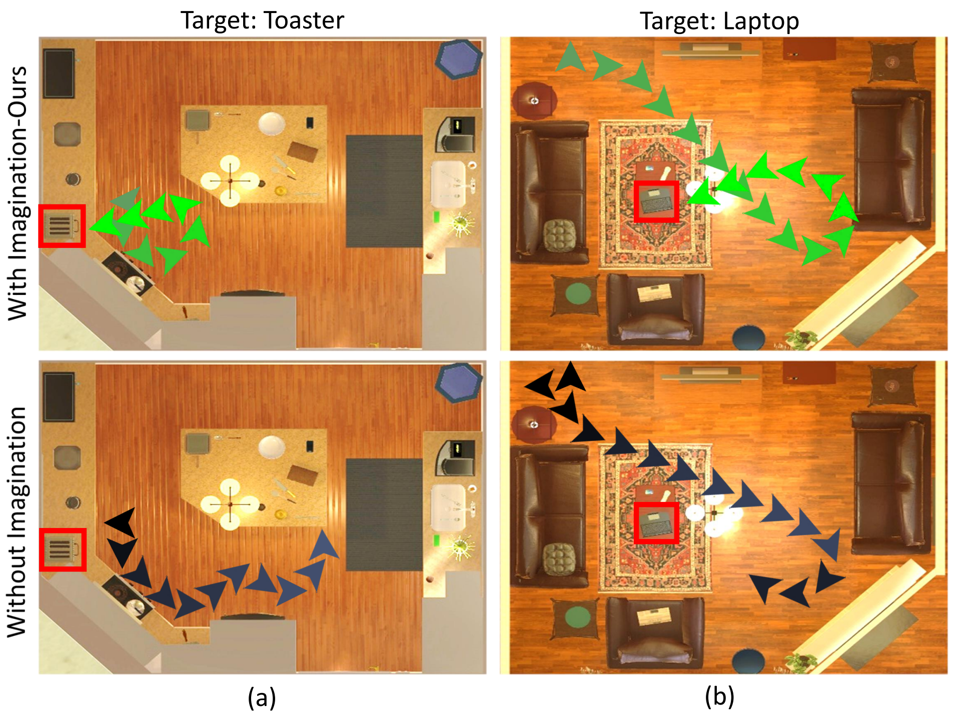


Figure 5.6: Although our agent achieves an overall higher success rate, the length of the trajectory could still be more optimal.

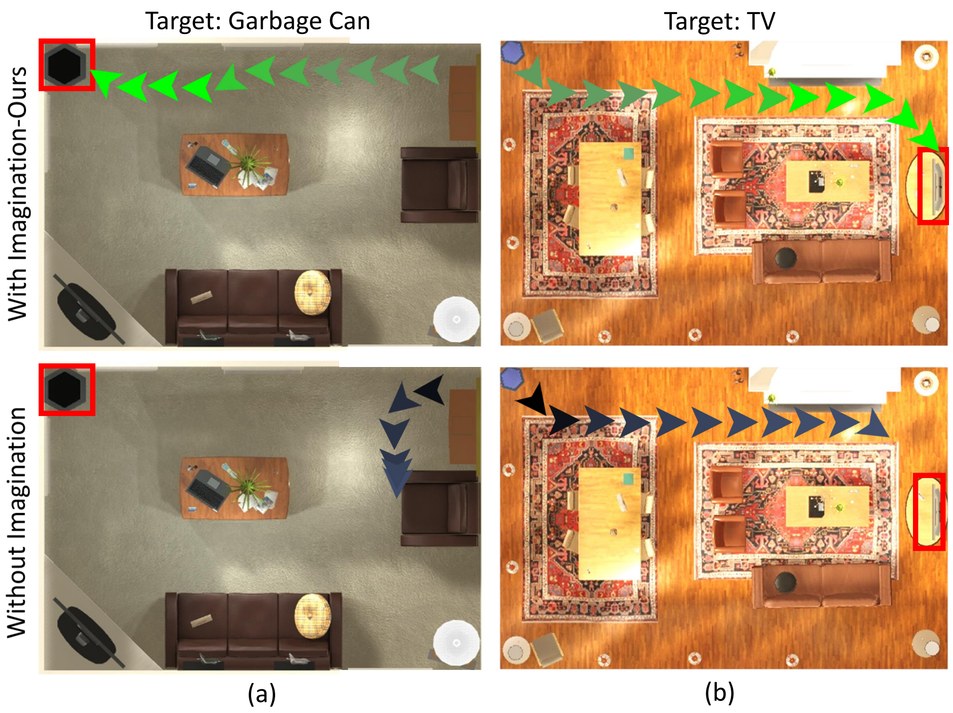


Figure 5.7: Our ForeSI agent has learnt to imagine the optimal sub-goal hence it takes a near-optimal trajectory while the baseline method fails.

method has the potential to address the two major problems with the current previous state-of-the-art methods.

In Figure 5.8 we present sample failure cases of our method where the baseline agent is able to complete the trajectories successfully. In Figure 5.8 (a), our agent fails to detect the "alarm clock" despite being visible in the first few state observations and instead moves towards the place where a bedside is usually located. This can be due to the fact that during the training the agent has mostly observed the "alarm clock" closer to the bed hence our agent imagines a state closer to the bed. In Figure 5.8 (b), although our agent takes the correct trajectory that leads to the target, "plant", it fails to stop close enough. On the contrary, the baseline method [115] stops within the 1-meter proximity and is successful.

In Figure 5.9 we present qualitative results of our sub-goal identification method as described in Section 5.2.1. In that figure, random successful trajectories from the unseen test set are visualised. The goal state is shown with a green frame and the selected

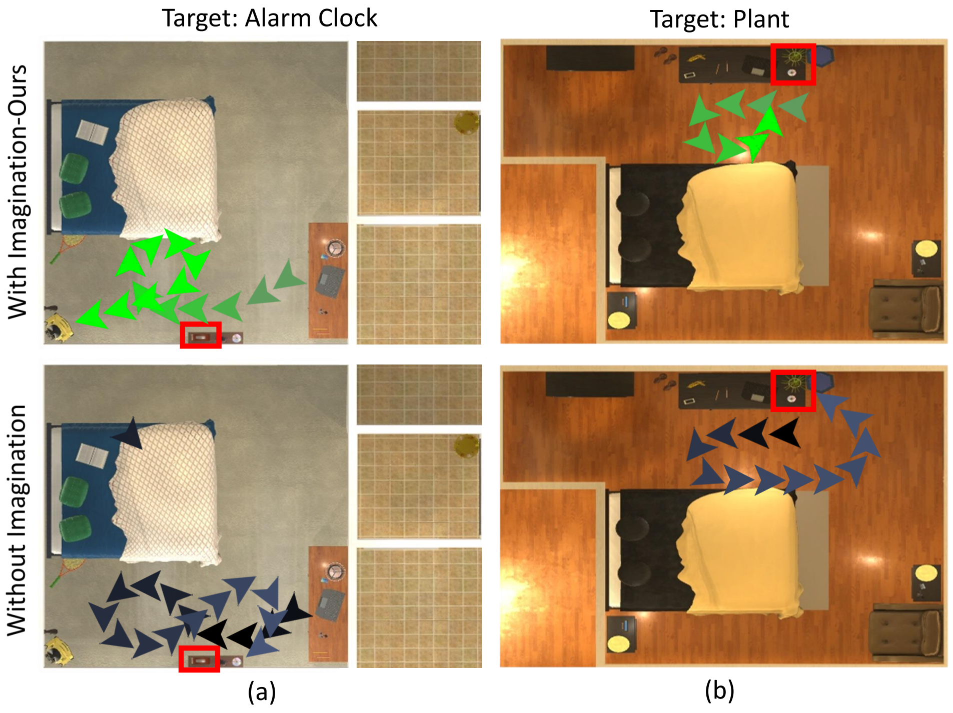


Figure 5.8: Failure cases of our method; ***(a)*** *shows a sample failure that might happen due to imagining the wrong state (here a bedside near the bed where the alarm clock normally is located) and* ***(b)*** *happens due to early stopping.*

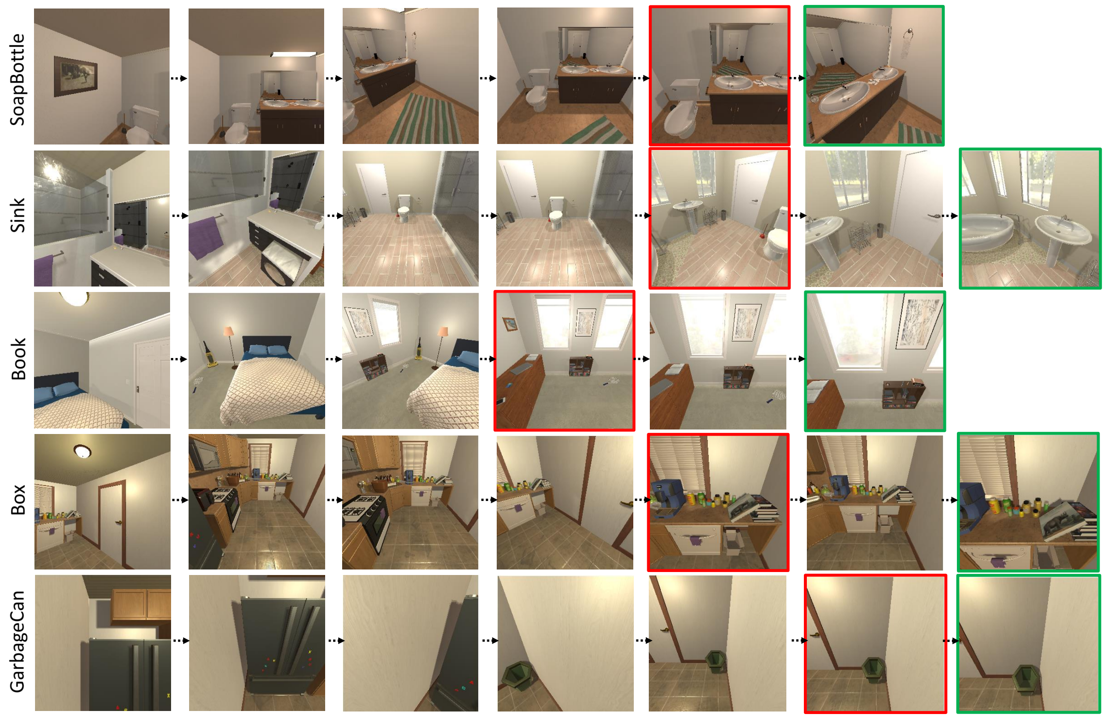


Figure 5.9: Sample trajectories from the unseen test set showing sub-goal selection results using our method. The ***red*** *frame shows the selected sub-goal state and the* ***green*** *frame the final successful goal state.*

sub-goal state is shown with a red frame. We can see that the sub-goal state highly correlates with the real goal state. In all those episodes our agent learns to identify the sub-goals that have a few common features: (1) the target object is clearly visible; (2) the goal state is reachable with very few steps; (3) from the sub-goal to the goal state the agent needs to take simple actions, usually just *MoveAhead*.

5.3.5 Method Limitations

While it has been proven effective, we consider this work as an initial step towards enabling forward modelling for visual navigation. One interesting future direction may be to investigate the role of sub-goal generation for better exploration. Moreover, we may also improve the sub-goal generation by better learning from failed episodes due to inaccurate sub-goals.

5.4 Conclusion: Method Summary

In this work, we showed that integrating a forward model into visual navigation using ForeSI improves both the robustness and efficiency of navigation. We achieved this by enabling our agent to imagine (i.e. to predict) a future sub-goal state that leads to a successful navigation episode. Our extensive experiments show how ForeSI can be integrated into a wide variety of methods to improve their navigation performance.

Chapter 6

Harmonious Multi-Agent Reinforcement Learning

In the previous chapters, we discussed our proposed methods to learn a more robust and efficient visual navigation policy using single-agent RL. Many real-world problems, however, involve multiple learning agents which imposes additional learning challenges. Motivated by that, in this chapter we develop a multi-agent reinforcement learning algorithm to tackle such problems. In our setting, agents maximise their own individual rewards, which may give rise to a conflict, if the objectives of the involved agents overlap. As a result, agents trained independently may show aggressive behaviour which increases the performance disparity across the population of agents. In this chapter, we propose a novel method to reduce the performance disparity and instead promote *harmonious* behaviour among the agents.

6.1 Background

Most of the previous works in Multi-Agent Reinforcement Learning (MARL) are based on the assumption that agents share the same reward function. In such scenarios, agents are inherently motivated to cooperate to achieve higher utility. In many real-world multi-agent problems, however, that assumption does not always stand true. That is, agents have individual reward functions *(i.e. they are self-interested)* which results in the conflict of interest, at least at some stage during their interactions with other agents, if not continuously. That challenge is further exacerbated when there are limited environmental resources to share among agents. To avoid aggressive selfish behaviour it is important in such problems to design mechanisms that establish cooperative social behaviour among those agents.

Autonomous driving is a good example of such a problem where agents (i.e. drivers) are self-interested (i.e. try to reach their destination in the shortest possible time) while sharing limited resources (e.g. road capacity) with other agents. Humans can efficiently interact with other road users while showing social behaviour. An example of such behaviour is highlighted when part of a road is blocked and drivers take turns to pass, or while merging into a highway where drivers on the highway let merging vehicles do so smoothly. Current autonomous vehicles, however, mostly circumvent learning such complex cooperative behaviours by learning safe, but rather conservative, policies. Developing driving policies that can demonstrate human-like complex social behaviours is non-trivial.

Previous works propose different mechanisms to promote cooperation among agents. Such methods can be categorised into mainly two ideas: designing better rewarding mechanisms to reward agents for their contribution to the success or failure of other agents (e.g. average neighbourhood reward), or encouraging action coordination by maximising the statistical dependence between agents' policies or between policies and the joint state. The former category of methods assumes the long-term effect of an agent's actions on other agents' performance, which does not always hold. This is specifically not the case when agents interact temporarily in an ad-hoc fashion. The second category of work is constrained by communication requirements and has been less explored among a large population of self-interested agents.

In this work, we propose to train agents that learn to *empower other agents* through *self-awareness*. We model the effect of an agent's current policy on the (near-) future performance of another agent. Through that explicit modelling, our agents learn policies that balance maximising their own reward and that of another agent by avoiding actions that adversely affect the future performance of others. By assuming local and short-term mutual interactions between agents our method is more suitable for an ad-hoc setting where the assumption of the long-term effect of actions does not hold all the time.

In summary, our contributions are as follows:

- We propose an objective based on mutual information, inspired by Empowerment [50] that enables neighbour empowerment, across agents.
- We introduce self-awareness into agents' state which enables neighbour empowerment and improves the interpretability of learned policies.
- We empirically show that our framework effectively improves social behaviour among self-interested agents interacting in an ad-hoc fashion. Compared to a range of baselines our method improves the safety and success rate of driving in complex autonomous driving scenarios.
- Finally, our method is more robust to irrational behaviour among agents, compared to the baseline.

6.1.1 Problem Statement: Self-Interested Agents

In our problem agents have their own individual reward function; that is agents are self-interested. We can use the framework of Partially-Observable Stochastic Games (POSGs) to formally define such a problem, as discussed in Sec. 2. In POSG, every agent maximises its own objective, $J_i = \mathbb{E}[R_i|\pi_i, \pi_{\neg i}]$, where $R_i = \sum_t \gamma_i^t r_i^{(t)}$ is the discounted sum of its future reward, according to its own reward function, and π_i is its policy. We use sub-script indices for agent index and super-script in parentheses, where required, to indicate the time index, and $\neg i$ means all the agents except agent i.

Moreover, in our framework, we assume agents can share extra information during training within their local neighbourhood, defined with a distance threshold. During the evaluation, however, agents take actions sampled from their individual policies that are conditioned on their local observation, only. This framework is known as Centralised Training, Decentralised Execution (CTDE) and has been used in various previous works [28].

6.2 Method

6.2.1 Neighbour Empowerment Through Self-Awareness

Empowerment, is an information-theoretic concept, introduced by [50], defined as the channel capacity between an agent's actions and its *own* future state, as below:

$$\varepsilon = \texttt{max}\, C(s^{(t+1)}; a^t) = \texttt{max}\; H(s^{(t+1)}) - H(s^{(t+1)}|a^{(t)}), \tag{6.1}$$

where $s^{(t+1)}$ is the future state and a^t is the agent's actions sampled from its policy. The above definition of Empowerment has previously been used in single-agent RL to improve the agent's exploration policy [70]. By maximising the Empowerment as an objective or intrinsic reward the agent learns to take actions leading to states with more control; this can, also, be seen in the entropic definition of the mutual information 6.1 as it reduces the entropy of the conditional state distribution while maximising the marginal entropy of the future state. For instance, in a robotic navigation scenario, an agent maximising Empowerment will more likely take actions leading to locations with high accessibility to other future locations (i.e. states).

In this work, inspired by single-agent Empowerment, we aim to train self-interested agents that act less selfishly, i.e. *they empower others*. Empowering other agents can intermittently be at odds with an agent's objective of maximising its own reward. This becomes even more challenging as the number of agents increases since the future state of agents is a function of the joint actions of all of them. Therefore, naively maximising Empowerment across agents in large populations can result in training instability and non-convergence. To address that problem, we introduce *Self-Awareness (SA)* variable into every agent's state. SA estimates an agent's discounted past performance, thus far into an episode. For instance, it could be an estimate of own collected reward over the past T timesteps or the sum of the temporal value differences over the same interval.

Encoding self-awareness into an agent's state introduces two main advantages. Firstly, it enables the agent to constantly be aware of its past performance which can be utilised for explainability in safety-critical applications, such as our autonomous driving setting. More importantly, it can be utilised to enable neighbour empowerment without causing training instability. Therefore, we propose a new Empowerment-inspired mutual information objective, which we call *CrossEmpowerment*; CrossEmpowerment is the channel capacity between one agent's action and another agent's SA variable:

$$\varepsilon = \texttt{max}\, C(a_i^t;\ SA_j^{(t+1)}),\ \ \forall\, i, j \in \mathcal{N}\ \texttt{and}\ \ i \neq j. \tag{6.2}$$

CrossEmpowerment maximises the dependence between an agent's current action and the future SA variable of *another* agent (e.g. a neighbouring agent). The increased dependency regulates the selfish behaviour of self-interested agents and encourages agents to care about the future performance of their neighbours. This can also been seen in the entropic definition of CrossEmpowerment which leads to reduced entropy of the conditional distribution of the future SA of one agent given another agent's action: $\texttt{I}(a_i^{(t)};\ SA_j^{(t+H)}) = H(SA_j^{(t+H)}) - H(SA_j^{(t+H)}|a_i^{(t)})$.

In practice, agents can communicate their SA locally within a pre-defined neighbourhood during training while imposing negligible communication costs. Moreover, in ad-hoc agent interactions, where the interactions are short, communicating the SA alleviates the challenge of learning to estimate those from short-term observations.

Our proposed SA variable has a general definition to work with (almost) any reward function, which may vary across problems. In its most general form, it is defined as the discounted sum of the past reward, since the beginning of the episode:

$$SA_i^{(T)} = \sum_{t=0}^{t=T} \gamma^{T-t} r_i^{(t)}. \tag{6.3}$$

The above definition can be seen as the backward value function, where the rewards collected more recently have a higher influence. Intuitively, this represents the temporal effect of receiving the reward, which diminishes through time as its memory disappears.

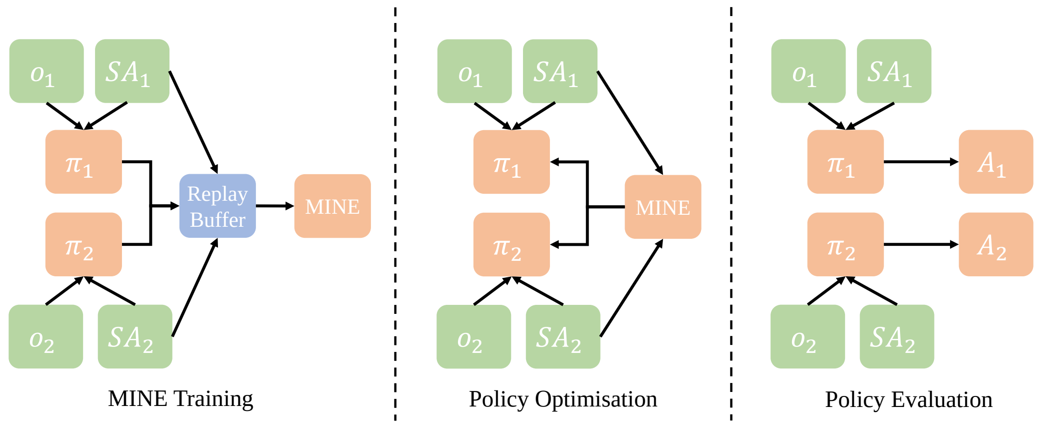


Figure 6.1: Overview of different stages of training and evaluation of our framework. During the training, we alternate between training mutual information estimator and optimising the policy.

Therefore, agents will be encouraged to constantly continue receiving rewards. In sparse reward settings, where the agent is rewarded according to its achievement at the end of the episode, this can be replaced with an estimate of the progress into the task (or a learned function approximating the progress). We leave sparse reward applications for future work.

6.2.2 Optimising CrossEmpowerment in MARL

As described in Sec. 6.2.1, we propose to add CrossEmpowerment into the multi-agent learning objective to improve cooperation among self-interested agents. As a result, the agents learn optimal policies that not only maximise their individual reward, but it does so while empowering other agents. Our CrossEmpowerment alleviates the selfish behaviour agents might demonstrate as a result of maximising their own individual reward while sharing the resources (e.g. the state space) with other agents.

Our CrossEmpowerment can be jointly optimised along with any policy-based reinforcement learning algorithm, and the new objective for every agent will be as follows:

$$\mathcal{J}_{\theta_i} = \max_{\theta}\mathbb{E}[\sum_{t=0}^{T}\gamma_i^t r_i^{(t)}] + \sum_{t=0}^{T}\sum_{j\notin i}\alpha_j \mathtt{I}(a_i^{(t)};\ SA_j^{(t+H)}|o_i^{(t)}). \tag{6.4}$$

In this work, we add our objective to the original Proximal Policy Optimisation (PPO) [93] objective defined as below. We follow individual policy learning, IPPO introduced by [114] which uses a centralised advantage function for more stable training:

$$\nabla_{\theta_i}\mathcal{J}_{\theta_i} = \mathbb{E}_{(a_i^{(t)}\sim\pi_{\theta_i},o_i^{(t)})}[\nabla_{\theta_i}\log\pi_{\theta_i}(a_i^{(t)}|o_i^{(t)}, SA^{(t)})\mathtt{U}(\boldsymbol{o}^{(t)}, a_i^{(t)}, \boldsymbol{a}_{\neg_i}{}^{(t)})], \tag{6.5}$$

$$+\sum_{j\notin i}\nabla_{\theta_i}\alpha_j \mathtt{I}(a_i^{(t)};\ SA_j^{(t+H)}|o_i^{(t)}). \tag{6.6}$$

where $SA_j^{(t)}$ is the neighbour agent's SA variable, as defined in Sec. 6.2.1, $\mathtt{U}(\boldsymbol{o}^{(t)}, a_i^{(t)}, \boldsymbol{a}_{\neg_i}{}^{(t)}) = r_i^{(t)}+\gamma_i V_i(\boldsymbol{o}^{(t)})-V_i(\boldsymbol{o}^{(t-1)})$ is the centralised advantage function, $\nabla_{\theta_i}\alpha_j \mathtt{I}(a_i^{(t)};\ SA_j^{(t+H)}|o_i^{(t)})$ is the weighted gradient of the mutual information. Following previous works, we also use the surrogate objective in PPO [93] and policy clipping to optimise the return maximisation term in our objective.

6.2.3 Neural Mutual Information Maximisation Using Reparameterisation

The exact optimisation of the mutual information in our objective introduced in Sec. 6.2.1 is intractable. Previous methods, introduce different mutual information lower-bounds that we can utilise here [29, 70]. The unbounded nature of mutual information can make designing the appropriate reward function (e.g. the normalisation factor) a difficult manual process. Moreover, it is preferred for the lower bound to be approximated with a neural network so we can utilise batch optimisation to increase efficiency when dealing with a large number of agents. Mutual Information Neural Estimation (MINE) [10] utilises the dual formulation of the KL-Divergence to train a neural network that approximates the mutual information lower bound. We use the re-parameterisation trick [49] to directly optimise an agent's policy based on the mutual information estimated by MINE. An overview of our framework is shown in Figure 6.1. We run ablation studies on the effect of this choice in our experiments in Sec. 6.3.

6.3 Experiments

In this section, we aim to answer the following empirical questions:

1. Does our method effectively promote social behaviour and alleviate the selfish behaviour of self-interested agents? How can we quantify such behaviour?
2. What are the strengths and weaknesses of our proposed method measured using the metrics defined below?
3. How do agents trained using our method performs when interacting with a specific type of irrational agents?

We compare our method to different baselines using a range of different performance metrics, besides the conventional episodic reward. This provides further insights into the dynamics of our proposed method. The metrics are as follows:

Success Rate: the ratio of the number of agents successfully completing their episodes (i.e. reaching their destination avoiding accidents etc.), over the total number of agents spawned in one episode.

Fairness (harmony): following previous methods [43], we define fairness as the coefficient of variation of reward among the agents in an episode: $\texttt{CV} = \sqrt{\frac{1}{n-1}\sum_n \frac{(R_i-\mu)^2}{\sigma^2}}$

Safety: following our baseline [80], we define safety as the average rate of failures in an episode; we consider the following failures in an episode: crashing into objects or other vehicles, driving on the wrong side of the road, or driving off the road (e.g. on the kerb).

6.3.1 Experiment Setup and Simulation Scenarios

We run our experiments in our extended version of the recently introduced MetaDrive [55] simulator. MetaDrive [55] is a lightweight and fast simulation environment suitable for evaluating MARL algorithms in different driving scenarios.

In this work, we select and create driving scenarios which have three distinctive features:

1- they require learning complex interactions among multiple agents, 2- such interactions require learning social cooperative behaviour beyond existing right-of-way rules, and 3- they highlight the conflict between acting selfishly or showing cooperative behaviour (at the cost of some personal reward). Figure 2.2 shows our experimental scenarios, which we define below:

(a) Bottleneck: In this scenario the agents are spawned on one end of the road (left or right) and their goal is to reach the other end of the road, travelling through the shortest path and as fast as possible. The number of lanes at the bottleneck is reduced from four lanes to a single lane on each side of the road; that is when agents can either be aggressive and selfish or considerate and pro-social.

(b) T-Intersection: The agents are spawned randomly and follow their set goals similar to (a). Agents driving through the vertical road aiming to reach the right end of the main horizontal road are at a disadvantage due to the challenge of crossing the horizontal road. Without cooperation from the agents driving on the horizontal road, those disadvantaged agents will be waiting for extended periods (hence will receive a lower reward).

(c) Roundabout: The main challenge in this scenario is to merge into the roundabout efficiently. Here an agent inside the roundabout will open more space for the agents waiting to enter at a small cost of its own reward but empowering other agents to showcase social behaviour.

We implement our method in RLLib [57] using PyTorch. We use a replay buffer of 1024 episodes to train our mutual information network. We uniformly sample mini-batches of 512 observation-action pairs to simulate the joint distribution and re-sample actions separately to simulate drawing samples from the marginal action distribution. We then use those mini-batches to train MINE, which consists of a two-layer Multi-Layer Perceptron (MLP) with LeakyReLU non-linearity in between layers. We train MINE for 5 epochs on every replay buffer. We then reuse the episodes from the same replay buffer to train our policy and value networks. The mutual information loss coefficient is set to 0.1. We train all our baselines and our method for 1 M episodes and every experiment is repeated 5 times with different random seeds.

Except suggested otherwise, all our methods are trained with the following reward

	Episode Return	**CV**	**Success Rate**	**Safety**	**Average Velocity**
IPPO	14264.63 ± 1614.11	0.52 ± 0.05	0.41 ± 0.02	0.26 ± 0.13	**31.67 ± 1.30**
CoPO	13732.91 ± 1132.24	0.49 ± 0.04	0.53 ± 0.01	0.24 ± 0.19	24.04 ± 2.11
Ours	**16234.93 ± 810.98**	**0.45 ± 0.03**	**0.59 ± 0.05**	**0.10 ± 0.05**	25.40 ± 3.59

Table 6.1: Comparison of our results against the baselines in T-Intersection scenario. IPPO is the simplest baseline where agents learn independently and CoPO agents maximise average neighbourhood rewards. Mean and variance over five random seeds are shown.

function:

$$r_i^{(t)} = r_{i,\,success}^{(t)} + \alpha r_{i,\,speed}^{(t)} + \beta r_{i,\,distance}^{(t)}, \tag{6.7}$$

where $r_{i,\,success}^{(t)}$ is the binary success reward with a fixed value of 10.0 if the episode ends successfully, and 0.0 otherwise; $r_{i,\,speed}^{(t)} = (current\_speed/max\_speed) * road\_indicator$ is the speed reward, where $road\_indicator$ indicates whether the vehicle is on the correct path to the target location, $r_{i,\,distance}^{(t)} = distance\_travelled * road\_indicator$ the travelled distance reward, and α and β are their corresponding coefficients, set to 0.1 and 1.0 respectively, in our experiments.

We utilise a simpler truncated version of the above reward function as SA added to individual agent state observations. We define SA as the normalised discounted sum of past speed reward (set $\beta = 0$). This is a proxy for the full reward which makes it more interpretable while focusing on the waiting time. Including the distance, the reward can cause non-convergence due to different agents travelling distances before they meet other agents at ad-hoc interaction points.

6.3.2 Baseline Methods and Quantitative Comparison

In Table 6.1 we compare our method against some baseline methods. Our two baselines represent a trivial solution and a strong recently introduced method, as discussed below:

IPPO [114]: This is the simplest baseline where every agent learns independently using the reward function introduced in Eq. 6.7. Regardless of how the reward coefficients are adjusted (which requires inefficient manual tuning for each task), agents trained using this method demonstrate aggressive selfish driving behaviour which results in less harmony among the population of agents.

CoPO [80]: This method proposes to integrate the average neighbourhood reward into the individual reward of each agent. Such addition encourages agents to learn policies which are more cooperative since agents are rewarded (or punished) for the success (or failure) of their neighbours.

As we can see in Tab. 6.1, our method improves the success rate, CV (lower is better, more harmonious) and safety compared to other baselines. This comes at a small cost of average velocity. We argue that this is due to agents learning to drive more cautiously giving way to other vehicles, that are otherwise stuck for longer periods.

Additionally, in Figure 6.2 we show results in other driving scenarios, including the roundabout and the bottleneck, described in Section 6.3.1. Each curve shows the mean and standard deviation of the results over five random seeds. As we can see, our method improves the total reward achieved by the population of agents in all three driving scenarios; the improvement is more significant in the bottleneck scenario which highlights the necessity of

6.3.3 Reward Shaping Study

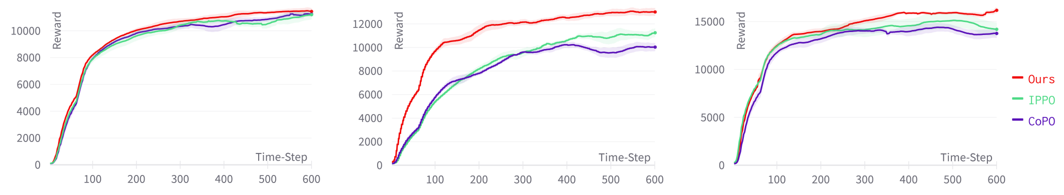
Here we aim to investigate whether a reward-shaping approach could help alleviate the aggressive behaviour among the agents. We observe the best overall results when setting $\beta = 10.0$ in Eq. 6.7 to allow agents for slower, i.e. more conservative, driving. The comparison results in the T-Intersection scenario are summarised in Table 6.2. Note the increase in the overall reward is due to the new reward function and does not mean better performance. We can see that using careful reward tuning can lead to a reduction in crash rate; however, the harmony among the agent is further lost, i.e. the CV value is increased. The agents become less aggressive (and slower) to reduce the crash rate.

6.3.4 Robustness Against Irrational Agent Behaviour

In this section, we design an experiment to provide further insights into the limitations of our proposed framework and those of the baseline methods. Our empirical studies thus far

	Episode Return	CV	Success Rate	Safety	Average Velocity
Ours, Reward Shaping	**20920.94** ± **1049.99**	0.54 ± 0.05	0.38 ± 0.03	0.18 ± 0.09	22.08 ± 3.01
Ours, *SA Variable*	16234.93 ± 810.98	**0.45** ± **0.03**	**0.59** ± **0.05**	**0.10** ± **0.05**	**25.40** ± **3.59**

Table 6.2: Ablation study: we tune the weightage of different reward components to try to alleviate selfish behaviour.



(a) Roundabout (b) Bottleneck (c) T-Intersection

Figure 6.2: Training curves in different driving scenarios.

show that our method effectively promotes social behaviour among self-interested agents. This is achieved under the rationality assumption of all the agents during all the episodes. In reality, however, agents might break such assumptions, intrigued by different internal or external factors, especially in mixed-autonomy environments. One of those factors, studied in the previous social behaviour of human drivers [85] is emotion and how it affects both human perception and decision making. To investigate the generalisation of our method here, we design an environment where we introduce noise into a subset of randomly selected agents' observations or actions. The noise is an inverse function of their SA variable. Intuitively, when agents continuously observe lower own performance they become emotional which then reduces their accurate perception ability and increases the likelihood of their irrational behaviour.

In Table 6.3 we compare our method against a baseline, IPPO, in a T-Intersection driving scenario in such a setting. *"Obs"* indicates adding noise to the observation of every agent while *"Act"* indicates noisy actions. In both cases, the magnitude of the noise is increased exponentially proportional to the individual observed performance of an agent. With this formulation, we aim to simulate scenarios where agents act increasingly stochastic as a function of their personal performance. We can see that in general the noisy actions more heavily affect the overall performance of the agents. We hypothesise that adding noise to the actions reduces the controllability of the vehicle more significantly. That is, the agent

	Safety	**Success Rate**
IPPO (Obs)	38.01 ± 0.11	29.28 ± 0.03
Ours (Obs)	14.25 ± 0.06	52.08 ± 0.09
IPPO (Act)	27.56 ± 0.12	23.27 ± 0.04
Ours (Act)	18.32 ± 0.03	49.34 ± 0.07

Table 6.3: Comparing our method against the baseline, IPPO, in our new setting, where irrational behaviour has been made possible, shows the relatively higher robustness of our method.

is unable to observe the effect of committing to an action predicted by its policy. Adding noise to the observation, however, may only result in missing obstacles at times.

6.4 Method Limitations

One of the main limitations of our method arises naturally due to having two, potentially conflicting, optimisation objectives. In any such method, including ours, tuning the contribution of the two objectives may require additional efforts for new environments. The use of a training curriculum to adaptively adjust those parameters may be effective which we leave for future work.

6.5 Conclusion: Method Summary

In this chapter, we introduced a mutual information framework inspired by Empowerment [50] to promote social behaviour among self-interested agents. We achieve that through encoding self-awareness into an agent's state that can then be efficiently shared among agents within a local neighbourhood. Through empirical studies, we show that our method results in a stronger cooperative behaviour among agents leading to a higher overall success rate, and more importantly more *harmonious* population performance.

Chapter 7

Conclusion

In this thesis, we present multiple original contributions to the field of reinforcement learning for real-world problems. Our proposed methods address two of, arguably, the most challenging and fundamental real-world sequential decision-making problems, i.e. visual navigation and autonomous driving. We start from single-agent RL, where an agent interacts with a stationary environment to learn an optimal policy. We, further, continue to multi-agent RL, where a large population of agents simultaneously learn to interact with other agents. In this chapter, we summarise our findings and contributions and discuss the implications of those contributions for any future work.

7.1 Summary

Despite the great success of RL on toy environments [36, 45] its capacity to solve real-world problems remains limited. Inspired by that, in this thesis, we propose tackling visual navigation and autonomous driving using novel RL-based methods. Such problems are selected to highlight some of the main challenges RL may potentially face in real-world. In visual navigation, for instance, the agent needs to learn goal-driven state representations from partial observations in a complex 3D environment. Those representations then need to be utilised by the policy to learn to search the environment while committing to finding the target object. Moreover, any learned policy is then evaluated in an unseen environment which further challenges the agent with generalisation ability. Differently, in autonomous driving, the agent is challenged by learning intricate interactions with other vehicles on the road in a continuous actions space.

In Chapter 4, we present a simple yet effective method to improve the performance of the actor-critic RL algorithm for visual navigation. Our method alleviates the state-value under-estimation problem by incorporating object co-location information as a neural graph. Actor-critic RL is heavily reliant on the accuracy of the state-value approximation for policy convergence. This is mainly because accurate value estimation helps with the correct identification of the actions along the trajectory that contribute the most to the final reward, i.e. credit assignment. Through extensive empirical studies, we show that our trained agent is realistically more optimistic (*e.g.* accurately predicts higher state values). This leads to successful navigation towards the target object when the baseline agents typically give up. We run multiple ablation studies and show the contribution of our design choices to the overall framework.

In Chapter 5, we show that integrating a forward model into visual navigation using ForeSI improves both the robustness and efficiency of navigation. We achieve this by enabling our agent to imagine (i.e. to predict) a future sub-goal state that leads to a successful navigation episode. Our forward model avoids costly step-wise future production and instead predicts only a latent state if the navigation episode were to be successful. Our extensive experiments show how ForeSI can be integrated into several previous state-of-the-art methods to boost their navigation performance.

In Chapter 6, we propose a mutual information framework inspired by Empowerment [50] to promote selfless social behaviour among self-interested agents. We achieve that through encoding *self-awareness* into our agents' observation which we assume can be observed by neighbouring agents. An agent observing a neighbour's self-awareness value learns to *empower* lower performing agents hence increasing harmony among the population. Through empirical studies, we show that our method results in a stronger cooperative behaviour among agents leading to a higher overall success rate.

7.2 Potential Future Work Directions

At the time of writing this thesis, research on real-world RL is still in its infancy while gaining increasing attention. More recently, the main challenges ahead of real-world RL have been articulated to encourage the research community to further focus on such problems [25]. In this thesis, our proposed methods tackle some of those challenges, e.g. partial-observability, generalisation to unseen test environments, continuous state and action spaces, and environment non-stationarity as a result of co-existence of other learning agents. Our proposed methods, while insightful and valuable, are still far from completely solving such challenges. Therefore, we expect multiple different ways those findings can be utilised in any potential future work. In this section, we review some of the most promising ones for the interested reader, e.g. fresh graduate students and current researchers.

Our proposed method in Chapter 4 takes the first step towards a deeper understanding of the role of the critic in actor-critic RL [67]. Recently, several other works have also considered similar research questions, although mostly in a toy environment [110]. In multi-agent RL community similar ideas have been explored that allow the critic to have access to more global (i.e. centralised) information [28]. In our work, different from those, we partially decompose the value function approximator to allow access to more global information. Instead of GNN, one may build a top-down map of the observed environment to enhance the input to the critic. Exploring the level of decomposition between the policy and the value networks could reveal more interesting avenues for future research. Another interesting research area would be to investigate the type of additional information the value network can have access to as well as its architecture. For instance, utilising learnable mapping approaches, e.g. [14], for value estimation would be one alternative. One could, also, investigate a different training regime where the critic lags (or leads) in the number of trained epochs compared with the policy network. Furthermore, validating the same findings on other actor-critic RL algorithms, e.g. PPO [93] would be an interesting future work direction. Moreover, injecting optimism through other means such as memory networks could also be a possible future direction. One could also quantify the optimism and utilise that to decide whether the current navigation trajectory may succeed or terminate early and avoid unnecessary exploration.

Regarding our seconds proposed method in Chapter 5, we, also, imagine a range of possible extensions. Imagination is a very complex ability in humans hence there are different ways artificial agents may be endowed with similar abilities. While research on world models continues [33, 51] such models are yet to prove accurate enough. Unlike our proposed method, one may adopt step-wise future prediction, as is commonly seen in model-based RL methods [34, 36]. In that case, future step imagination can be decomposed into two distinct tasks: modelling the transformation for the observed part of the world and predicting the unobserved parts. We could combine our method proposed in Chapter 4 to generate more realistic transformations of the past observations when predicting the future. For the parts of the future state that are yet to be observed, we could also utilise hierarchical prior representations of the world to help with more accurate predictions. For instance, the agent may first predict that turning right might lead to observing parts of a kitchen, in a household environment. It can then predict a more granular level of the items it may observe next, and so on. Learning when and how to re-imagine a sub-goal state would be another interesting potential future direction arising from our work. Such a method could be combined with an optimism measurement component to help with determining when the goal cannot be achieved. At that time, utilising the recent observations of the environment to generate a more accurate prediction of the sub-goal state could be another area of research. In many such problems, investigating the role of causality can potentially have a significant implication. For example, decomposing the latent variables that lead to different aspects of imagination, such as colour, shape, etc. could be very helpful. Such a method could take our proposed method to a new level, which currently relies upon correlations. Causality can also play a significant role in identifying the sub-goal states in hindsight. Our method, again, relies on correlations (i.e. using self-attention) to identify such important states, i.e. sub-goals. While it works in our setting, correlations do not always represent causal effects.

Finally, concerning our multi-agent RL method introduced in Chapter 6, we, also, discuss several potential future directions. Our method introduces an agent-level objective to encourage less selfish behaviour among agents. Our new objective, inspired by Empowerment, is integrated into actor-critic RL using linear scalarisation. The weights to combine the two objectives have been optimised empirically. Extending the linear scalarisation function to explicit multi-objective learning would be one immediate research direction to further balance the two objectives [38]. Differently, one may look at the same problem from the population level and introduce a new global objective which can promote such behaviour among the agents. That could further relate multi-agent RL to game theoretic concepts of social choice and mechanism design [75]. Furthermore, our method currently relies on the assumption of having access to the reward function of neighbouring agents. In practice, such assumption can be relaxed by learning the reward function of other agents from observations, e.g. using inverse RL [40]. Additionally, explicit selection of a specific neighbouring agent to be *empowered* could be another interesting area of research. For example, such a method could be more advantageous in scenarios where the agents need to coordinate to give way to an incoming emergency vehicle.

Overall, in this thesis, we address three main shortcomings of the current RL methods in real-world problems. We show that a visual navigation agent can utilise the relationship between different objects in an environment to more accurately estimate the state value, i.e. to remain *optimistic*. One should be aware of the problem of over-optimism and investigate in what scenarios remaining optimistic can help with achieving the task. Additionally, we show that *imagining* a goal state can help with the robust and efficient completion of a navigation episode. The content of the imagination to avoid potentially misleading input to the policy can be further examined. Finally, we show teaching agents to *empower* their neighbours can alleviate their selfish behaviour. The right balance between an agent's objective and that of helping others can change over time and space, striking which remains an open problem.

Bibliography

[1] Stefano V Albrecht and Peter Stone. ‘Autonomous agents modelling other agents: A comprehensive survey and open problems’. In: *Artificial Intelligence* 258 (2018), pp. 66–95.

[2] Stefano V Albrecht et al. ‘Interpretable goal-based prediction and planning for autonomous driving’. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 1043–1049.

[3] Alexander Amini et al. ‘Learning robust control policies for end-to-end autonomous driving from data-driven simulation’. In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 1143–1150.

[4] Peter Anderson et al. ‘On evaluation of embodied navigation agents’. In: *arXiv preprint arXiv:1807.06757* (2018).

[5] Peter Anderson et al. ‘Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments’. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 3674–3683.

[6] Marcin Andrychowicz et al. ‘Hindsight experience replay’. In: *Advances in neural information processing systems* 30 (2017), pp. 5048–5058.

[7] Giulio Bacchiani, Daniele Molinari and Marco Patander. ‘Microscopic traffic simulation by cooperative multi-agent deep reinforcement learning’. In: *arXiv preprint arXiv:1903.01365* (2019).

[8] Shaojie Bai, J. Zico Kolter and Vladlen Koltun. *An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling*. 2018. arXiv: 1803.01271 [cs.LG].

[9] Trapit Bansal et al. 'Emergent complexity via multi-agent competition'. In: *arXiv preprint arXiv:1710.03748* (2017).

[10] Mohamed Ishmael Belghazi et al. 'Mutual information neural estimation'. In: *International conference on machine learning* (ICML2018). PMLR. 2018, pp. 531–540. URL: https://arxiv.org/abs/1801.04062.

[11] Marc G Bellemare et al. 'The arcade learning environment: An evaluation platform for general agents'. In: *Journal of Artificial Intelligence Research* 47 (2013), pp. 253–279.

[12] Joan Bruna et al. 'Spectral networks and locally connected networks on graphs'. In: *arXiv preprint arXiv:1312.6203* (2013).

[13] Matthew Chang, Arjun Gupta and Saurabh Gupta. 'Semantic visual navigation by watching youtube videos'. In: *arXiv preprint arXiv:2006.10034* (2020).

[14] Devendra Singh Chaplot et al. 'Learning to explore using active neural slam'. In: *arXiv preprint arXiv:2004.05155* (2020).

[15] Devendra Singh Chaplot et al. 'Object goal navigation using goal-oriented semantic exploration'. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 4247–4258.

[16] Devendra Singh Chaplot et al. 'Object goal navigation using goal-oriented semantic exploration'. In: *Advances in Neural Information Processing Systems* 33 (2020).

[17] Kevin Chen et al. 'A Behavioral Approach to Visual Navigation with Graph Localization Networks'. In: *Robotics: Science and Systems XV* (2019). DOI: 10.15607/rss.2019.xv.010. URL: http://dx.doi.org/10.15607/RSS.2019.XV.010.

[18] Jiaxun Cui et al. 'Scalable multiagent driving policies for reducing traffic congestion'. In: *arXiv preprint arXiv:2103.00058* (2021).

[19] Hanjun Dai et al. *Learning Transferable Graph Exploration*. 2019. arXiv: 1910.12980 [cs.LG].

[20] Matt Deitke et al. ‘Robothor: An open simulation-to-real embodied ai platform’. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 3164–3174.

[21] Alexey Dosovitskiy et al. ‘CARLA: An open urban driving simulator’. In: *Conference on robot learning*. PMLR. 2017, pp. 1–16.

[22] Heming Du, Xin Yu and Liang Zheng. *Learning Object Relation Graph and Tentative Policy for Visual Navigation*. 2020. arXiv: 2007.11018 `[cs.CV]`.

[23] Heming Du, Xin Yu and Liang Zheng. *Learning Object Relation Graph and Tentative Policy for Visual Navigation*. 2020. arXiv: 2007.11018 `[cs.CV]`.

[24] Heming Du, Xin Yu and Liang Zheng. ‘VTNet: Visual Transformer Network for Object Goal Navigation’. In: *arXiv preprint arXiv:2105.09447* (2021).

[25] Gabriel Dulac-Arnold et al. ‘Challenges of real-world reinforcement learning: definitions, benchmarks and analysis’. In: *Machine Learning* 110.9 (2021), pp. 2419–2468.

[26] Kiana Ehsani et al. ‘ManipulaTHOR: A Framework for Visual Object Manipulation’. In: *arXiv preprint arXiv:2104.11213* (2021).

[27] Chelsea Finn, Pieter Abbeel and Sergey Levine. *Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks*. 2017. arXiv: 1703.03400 `[cs.LG]`.

[28] Jakob Foerster et al. ‘Counterfactual multi-agent policy gradients’. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 32. 1. 2018.

[29] Tim Franzmeyer, Mateusz Malinowski and João F Henriques. ‘Learning Altruistic Behaviours in Reinforcement Learning without External Rewards’. In: *arXiv preprint arXiv:2107.09598* (2021).

[30] Philip S Gallo Jr and Charles G McClintock. ‘Cooperative and competitive behavior in mixed-motive games’. In: *Journal of Conflict Resolution* 9.1 (1965), pp. 68–78.

[31] Evan Greensmith, Peter L Bartlett and Jonathan Baxter. ‘Variance reduction techniques for gradient estimates in reinforcement learning’. In: *Journal of Machine Learning Research* 5.Nov (2004), pp. 1471–1530.

[32] Jiuxiang Gu et al. ‘Scene Graph Generation With External Knowledge and Image Reconstruction’. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019). DOI: 10.1109/cvpr.2019.00207. URL: http://dx.doi.org/10.1109/CVPR.2019.00207.

[33] David Ha and Jürgen Schmidhuber. ‘World models’. In: *arXiv preprint arXiv:1803.10122* (2018).

[34] Danijar Hafner et al. *Dream to Control: Learning Behaviors by Latent Imagination*. 2020. arXiv: 1912.01603 [cs.LG].

[35] Danijar Hafner et al. ‘Learning latent dynamics for planning from pixels’. In: *International Conference on Machine Learning*. PMLR. 2019, pp. 2555–2565.

[36] Danijar Hafner et al. ‘Mastering atari with discrete world models’. In: *arXiv preprint arXiv:2010.02193* (2020).

[37] Eric A Hansen, Daniel S Bernstein and Shlomo Zilberstein. ‘Dynamic programming for partially observable stochastic games’. In: *AAAI*. Vol. 4. 2004, pp. 709–715.

[38] Conor F Hayes et al. ‘A practical guide to multi-objective reinforcement learning and planning’. In: *Autonomous Agents and Multi-Agent Systems* 36.1 (2022), pp. 1–59.

[39] Kaiming He et al. ‘Deep residual learning for image recognition’. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[40] Jonathan Ho and Stefano Ermon. ‘Generative adversarial imitation learning’. In: *Advances in neural information processing systems* 29 (2016).

[41] Natasha Jaques et al. ‘Social influence as intrinsic motivation for multi-agent deep reinforcement learning’. In: *International Conference on Machine Learning*. PMLR. 2019, pp. 3040–3049.

[42] RichardSocher JeffreyPennington and ChristopherD Manning. ‘Glove: Global vectors for word representation’. In: Citeseer.

[43] Jiechuan Jiang and Zongqing Lu. ‘Learning fairness in multi-agent systems’. In: *Advances in Neural Information Processing Systems* 32 (2019).

[44] Lukasz Kaiser et al. 'Model-based reinforcement learning for atari'. In: *arXiv preprint arXiv:1903.00374* (2019).

[45] Lukasz Kaiser et al. 'Model-based reinforcement learning for atari'. In: *arXiv preprint arXiv:1903.00374* (2019).

[46] John Kanu et al. *Following Instructions by Imagining and Reaching Visual Goals*. 2020. arXiv: 2001.09373 `[cs.LG]`.

[47] Diederik P Kingma and Jimmy Ba. 'Adam: A method for stochastic optimization'. In: *arXiv preprint arXiv:1412.6980* (2014).

[48] Diederik P Kingma and Max Welling. 'Auto-encoding variational bayes'. In: *arXiv preprint arXiv:1312.6114* (2013).

[49] Durk P Kingma, Tim Salimans and Max Welling. 'Variational dropout and the local reparameterization trick'. In: *Advances in neural information processing systems* 28 (2015).

[50] Alexander S Klyubin, Daniel Polani and Chrystopher L Nehaniv. 'Empowerment: A universal agent-centric measure of control'. In: *2005 ieee congress on evolutionary computation*. Vol. 1. IEEE. 2005, pp. 128–135.

[51] Jing Yu Koh et al. 'Pathdreamer: A world model for indoor navigation'. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 14738–14748.

[52] Eric Kolve et al. *AI2-THOR: An Interactive 3D Environment for Visual AI*. 2017. arXiv: 1712.05474 `[cs.CV]`.

[53] Ranjay Krishna et al. 'Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations'. In: *International Journal of Computer Vision* 123.1 (2017), 32–73. ISSN: 1573-1405. DOI: 10.1007/s11263-016-0981-7. URL: http://dx.doi.org/10.1007/s11263-016-0981-7.

[54] Joel Z Leibo et al. 'Multi-agent reinforcement learning in sequential social dilemmas'. In: *arXiv preprint arXiv:1702.03037* (2017).

[55] Quanyi Li et al. 'Metadrive: Composing diverse driving scenarios for generalizable reinforcement learning'. In: *arXiv preprint arXiv:2109.12674* (2021).

[56] Ruiyu Li et al. 'Situation Recognition with Graph Neural Networks'. In: *2017 IEEE International Conference on Computer Vision (ICCV)* (2017). DOI: 10.1109/iccv.2017.448. URL: http://dx.doi.org/10.1109/ICCV.2017.448.

[57] Eric Liang et al. 'RLlib: Abstractions for distributed reinforcement learning'. In: *International Conference on Machine Learning*. PMLR. 2018, pp. 3053–3062.

[58] Timothy P Lillicrap et al. 'Continuous control with deep reinforcement learning'. In: *arXiv preprint arXiv:1509.02971* (2015).

[59] Michael L Littman. 'Markov games as a framework for multi-agent reinforcement learning'. In: *Machine learning proceedings 1994*. Elsevier, 1994, pp. 157–163.

[60] Ryan Lowe et al. 'Multi-agent actor-critic for mixed cooperative-competitive environments'. In: *Advances in neural information processing systems* 30 (2017).

[61] Anuj Mahajan et al. 'Maven: Multi-agent variational exploration'. In: *Advances in Neural Information Processing Systems* 32 (2019).

[62] Kenneth Marino, Ruslan Salakhutdinov and Abhinav Gupta. 'The More You Know: Using Knowledge Graphs for Image Classification'. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017). DOI: 10.1109/cvpr.2017.10. URL: http://dx.doi.org/10.1109/CVPR.2017.10.

[63] Kevin R McKee et al. 'Social diversity and social preferences in mixed-motive reinforcement learning'. In: *arXiv preprint arXiv:2002.02325* (2020).

[64] Kevin R McKee et al. 'Social diversity and social preferences in mixed-motive reinforcement learning'. In: *arXiv preprint arXiv:2002.02325* (2020).

[65] Xiangyun Meng et al. *Scaling Local Control to Large-Scale Topological Navigation*. 2019. arXiv: 1909.12329 `[cs.RO]`.

[66] Piotr Mirowski et al. 'Learning to navigate in complex environments'. In: *arXiv preprint arXiv:1611.03673* (2016).

[67] Volodymyr Mnih et al. 'Asynchronous Methods for Deep Reinforcement Learning'. In: *Proceedings of The 33rd International Conference on Machine Learning*. Ed. by Maria Florina Balcan and Kilian Q. Weinberger. Vol. 48. Proceedings of Machine Learning Research. New York, New York, USA: PMLR, 2016, pp. 1928–1937. URL: http://proceedings.mlr.press/v48/mniha16.html.

[68] Volodymyr Mnih et al. 'Playing atari with deep reinforcement learning'. In: *arXiv preprint arXiv:1312.5602* (2013).

[69] Mahdi Kazemi Moghaddam et al. 'Optimistic Agent: Accurate Graph-Based Value Estimation for More Successful Visual Navigation'. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2021, pp. 3733–3742.

[70] Shakir Mohamed and Danilo Jimenez Rezende. 'Variational information maximisation for intrinsically motivated reinforcement learning'. In: *Advances in neural information processing systems* 28 (2015).

[71] Arsalan Mousavian et al. 'Visual Representations for Semantic Target Driven Navigation'. In: *2019 International Conference on Robotics and Automation (ICRA)* (2019). DOI: 10.1109/icra.2019.8793493. URL: http://dx.doi.org/10.1109/ICRA.2019.8793493.

[72] Ashvin V Nair et al. 'Visual reinforcement learning with imagined goals'. In: *Advances in Neural Information Processing Systems*. 2018, pp. 9191–9200.

[73] Suraj Nair and Chelsea Finn. 'Hierarchical foresight: Self-supervised learning of long-horizon tasks via visual subgoal generation'. In: *arXiv preprint arXiv:1909.05829* (2019).

[74] Soroush Nasiriany et al. 'Planning with goal-conditioned policies'. In: *Advances in Neural Information Processing Systems*. 2019, pp. 14843–14854.

[75] Noam Nisan and Amir Ronen. 'Algorithmic mechanism design'. In: *Proceedings of the thirty-first annual ACM symposium on Theory of computing*. 1999, pp. 129–140.

[76] Frans A Oliehoek and Christopher Amato. *A concise introduction to decentralized POMDPs*. Springer, 2016. URL: https://www.fransoliehoek.net/docs/OliehoekAmato16book.pdf.

[77] Afshin OroojlooyJadid and Davood Hajinezhad. 'A review of cooperative multi-agent deep reinforcement learning'. In: *arXiv preprint arXiv:1908.03963* (2019).

[78] Avik Pal et al. 'Emergent road rules in multi-agent driving environments'. In: *arXiv preprint arXiv:2011.10753* (2020). URL: https://arxiv.org/abs/2011.10753.

[79] Georgios Papoudakis et al. ‘Dealing with non-stationarity in multi-agent deep reinforcement learning’. In: *arXiv preprint arXiv:1906.04737* (2019).

[80] Zhenghao Peng et al. ‘Learning to Simulate Self-Driven Particles System with Coordinated Policy Optimization’. In: *Advances in Neural Information Processing Systems* 34 (2021).

[81] Aske Plaat, Walter Kosters and Mike Preuss. ‘Model-Based Deep Reinforcement Learning for High-Dimensional Problems, a Survey’. In: *arXiv preprint arXiv:2008.05598* (2020).

[82] Tabish Rashid et al. ‘Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning’. In: *International Conference on Machine Learning*. PMLR. 2018, pp. 4295–4304.

[83] Shaoqing Ren et al. ‘Faster r-cnn: Towards real-time object detection with region proposal networks’. In: *Advances in neural information processing systems*. 2015, pp. 91–99.

[84] Zhizhou Ren et al. ‘Exploration via Hindsight Goal Generation’. In: *Advances in Neural Information Processing Systems*. 2019, pp. 13485–13496.

[85] Ernst Roidl, Berit Frehse and Rainer Höger. ‘Emotional states of drivers and the impact on speed, acceleration and traffic violations—A simulator study’. In: *Accident Analysis & Prevention* 70 (2014), pp. 282–292.

[86] Ernst Roidl, Berit Frehse and Rainer Höger. ‘Emotional states of drivers and the impact on speed, acceleration and traffic violations—A simulator study’. In: *Accident Analysis & Prevention* 70 (2014), pp. 282–292.

[87] Stuart Russell. *Human compatible: Artificial intelligence and the problem of control*. Penguin, 2019.

[88] Mikayel Samvelyan et al. ‘The starcraft multi-agent challenge’. In: *arXiv preprint arXiv:1902.04043* (2019).

[89] Manolis Savva et al. *Habitat: A Platform for Embodied AI Research*. 2019. arXiv: 1904.01201 `[cs.CV]`.

[90] Alexander Sax et al. *Mid-Level Visual Representations Improve Generalization and Sample Efficiency for Learning Visuomotor Policies*. 2018. arXiv: 1812.11971 `[cs.CV]`.

[91] Julian Schrittwieser et al. 'Mastering atari, go, chess and shogi by planning with a learned model'. In: *Nature* 588.7839 (2020), pp. 604–609.

[92] John Schulman et al. *High-Dimensional Continuous Control Using Generalized Advantage Estimation*. 2015. arXiv: 1506.02438 `[cs.LG]`.

[93] John Schulman et al. 'Proximal policy optimization algorithms'. In: *arXiv preprint arXiv:1707.06347* (2017).

[94] Shai Shalev-Shwartz, Shaked Shammah and Amnon Shashua. 'Safe, multi-agent, reinforcement learning for autonomous driving'. In: *arXiv preprint arXiv:1610.03295* (2016).

[95] William Shen et al. 'Situational Fusion of Visual Representation for Visual Navigation'. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)* (2019). DOI: 10.1109/iccv.2019.00297. URL: http://dx.doi.org/10.1109/ICCV.2019.00297.

[96] Kihyuk Sohn, Honglak Lee and Xinchen Yan. 'Learning structured output representation using deep conditional generative models'. In: *Advances in neural information processing systems*. 2015, pp. 3483–3491.

[97] Kyunghwan Son et al. 'Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning'. In: *International Conference on Machine Learning*. PMLR. 2019, pp. 5887–5896.

[98] Matthijs TJ Spaan. 'Partially observable Markov decision processes'. In: *Reinforcement Learning*. Springer, 2012, pp. 387–414.

[99] Richard S Sutton et al. 'Policy gradient methods for reinforcement learning with function approximation'. In: *Advances in neural information processing systems* 12 (1999).

[100] Richard S Sutton et al. 'Policy gradient methods for reinforcement learning with function approximation'. In: *Advances in neural information processing systems*. 2000, pp. 1057–1063.

[101] Yunhao Tang and Alp Kucukelbir. *Hindsight Expectation Maximization for Goal-conditioned Reinforcement Learning*. 2020. arXiv: 2006.07549 [cs.LG].

[102] Yuval Tassa et al. 'Deepmind control suite'. In: *arXiv preprint arXiv:1801.00690* (2018).

[103] Dimitrios Troullinos et al. 'Collaborative multiagent decision making for lane-free autonomous driving'. In: *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*. 2021, pp. 1335–1343.

[104] Ashish Vaswani et al. 'Attention is all you need'. In: *Advances in neural information processing systems*. 2017, pp. 5998–6008.

[105] Varun Kumar Vijay et al. *Generalization to Novel Objects using Prior Relational Knowledge*. 2019. arXiv: 1906.11315 [cs.AI].

[106] Eugene Vinitsky et al. 'Benchmarks for reinforcement learning in mixed-autonomy traffic'. In: *Conference on robot learning*. PMLR. 2018, pp. 399–409.

[107] Oriol Vinyals et al. 'Grandmaster level in StarCraft II using multi-agent reinforcement learning'. In: *Nature* 575.7782 (2019), pp. 350–354.

[108] Haiyang Wang et al. 'Collaborative visual navigation'. In: *arXiv preprint arXiv:2107.01151* (2021).

[109] Ziyu Wang et al. *Dueling Network Architectures for Deep Reinforcement Learning*. 2015. arXiv: 1511.06581 [cs.LG].

[110] Ziyu Wang et al. 'Dueling network architectures for deep reinforcement learning'. In: *International conference on machine learning*. PMLR. 2016, pp. 1995–2003.

[111] Théophane Weber et al. *Imagination-Augmented Agents for Deep Reinforcement Learning*. 2018. arXiv: 1707.06203 [cs.LG].

[112] Luca Weihs et al. 'Allenact: A framework for embodied ai research'. In: *arXiv preprint arXiv:2008.12760* (2020).

[113] Ronald J Williams. 'Simple statistical gradient-following algorithms for connectionist reinforcement learning'. In: *Machine learning* 8.3-4 (1992), pp. 229–256.

[114] Christian Schroeder de Witt et al. 'Is independent learning all you need in the starcraft multi-agent challenge?' In: *arXiv preprint arXiv:2011.09533* (2020).

[115] Mitchell Wortsman et al. ‘Learning to learn how to learn: Self-adaptive visual navigation using meta-learning’. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 6750–6759.

[116] Cathy Wu et al. ‘Variance reduction for policy gradient with action-dependent factorized baselines’. In: *arXiv preprint arXiv:1803.07246* (2018).

[117] Fei Xia et al. ‘Gibson env: Real-world perception for embodied agents’. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 9068–9079.

[118] Jiachen Yang et al. ‘Learning to incentivize other learning agents’. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 15208–15219.

[119] Wei Yang et al. ‘Visual semantic navigation using scene priors’. In: *arXiv preprint arXiv:1810.06543* (2018).

[120] Seongjun Yun et al. *Graph Transformer Networks*. 2019. arXiv: `1911.06455 [cs.LG]`.

[121] Kaiqing Zhang, Zhuoran Yang and Tamer Başar. ‘Multi-agent reinforcement learning: A selective overview of theories and algorithms’. In: *Handbook of Reinforcement Learning and Control* (2021), pp. 321–384.

[122] Lunjun Zhang, Ge Yang and Bradly C Stadie. ‘World Model as a Graph: Learning Latent Landmarks for Planning’. In: *arXiv preprint arXiv:2011.12491* (2020).

[123] Marvin Zhang et al. ‘Solar: Deep structured representations for model-based reinforcement learning’. In: *International Conference on Machine Learning*. PMLR. 2019, pp. 7444–7453.

[124] Jie Zhou et al. ‘Graph neural networks: A review of methods and applications’. In: *arXiv preprint arXiv:1812.08434* (2018).

[125] Yuke Zhu et al. ‘Target-driven visual navigation in indoor scenes using deep reinforcement learning’. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)* (2017). DOI: 10.1109/icra.2017.7989381. URL: http://dx.doi.org/10.1109/ICRA.2017.7989381.