TECHNISCHE UNIVERSITÄT DRESDEN

"Friedrich List" Faculty of Transport and Traffic Sciences
Institute of Transport and Economics
Chair of Econometrics and Statistics, esp. in the Transport Sector

Master's Thesis

COMMUNICATION APPROACHES IN MULTI-AGENT REINFORCEMENT LEARNING

Vladyslav Nechai
Study program: Computational Modeling and Simulation

Supervised by:
Dipl.-Wi.-Ing. Martin Waltz

Reviewer:
Prof. Dr. Rer. Pol. Ostap Okhrin
Second reviewer:
Prof. Dr. Rer. Pol. Habil. Georg Hirte
Submitted on 15 March 2024

CONFIRMATION

I confirm that I independently prepared the thesis and that I used only the references and auxiliary means indicated in the thesis.

Dresden, 15 March 2024 Vladyslav Nechai

ABSTRACT

In decentralised multi-agent reinforcement learning communication can be used as a measure to increase coordination among the agents. At the same time, the essence of message exchange and its contribution to successful goal achievement can only be established with the domain knowledge of a given environment. This thesis focuses on understanding the impact of communication on a decentralised multi-agent system. To achieve this, communication is employed and studied in the context of Urban Air Mobility, in particular - to the vertiport terminal area control problem. A proposed in this work experimental framework, that promotes different information exchange protocols, allows to investigate if and how the agents leverage their communication capabilities. Acquired simulation results show that in the terminal area of a vertiport the aircrafts, controlled in a decentralised way, are capable of proper self-organisation, similar to the structured technique formulated in [Bertram and Wei(2020)]. A study of their communication mechanisms indicates that through different protocols the agents learn to signal their intent to enter a vertiport regardless of environment settings.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

CDF	Cumulative density function
Comm-MARL	Multi-agent reinforcement learning with communication
CTDE	Centralised training with decentralised execution
Dec-POMDP	Decentralised partially observable Markov decision processes
eVTOL	Electric vertical take-off and landing
LOS	Loss of separation
MARL	Multi-agent reinforcement learning
MDP	Markov Decision Process
NC	Non-communicative
PPO	Proximal Policy Optimization
RL	Reinforcement learning
UAM	Urban air mobility
VTAC	Vertiport terminal area controller

1 INTRODUCTION

In many real world situations multiple parties interact with each other to achieve their common and individual goals. Multi-agent reinforcement learning (MARL) is an area of machine learning that concerns such types of interactions between learning systems. The application domain for it is diverse: from controlling groups of artificial agents in video games [Vinyals et al.(2019)], to solving competitive predator-prey scenarios [Singh et al.(2018)], to navigating unmanned aerial vehicles for agricultural field imaging coverage [Marwah et al.(2023)]. In many occasions, the nature of interactions is cooperative, where participants act jointly with the purpose of attaining increased mutual benefits. In this regard, communication can play an important role in enabling better coordination among collaborators, as information exchange can provide a context, critical for optimal decision making.

Given the successful use of MARL in the domain of unmanned aerial vehicles [Xia et al.(2022)], potential adaptations to similar scenarios can be considered. Urban air mobility (UAM) is a concept of a future transportation system for passengers and cargo, facilitated by vertical take-off and landing electric aircrafts [Cohen et al.(2021)]. Taking into account the novelty of this idea, research is conducted around the aircraft [Silva et al.(2018)] and airspace design [Bauranov and Rakas(2021)], which includes the take-off and landing spots called vertiports. Provided a low space availability within an urban environment for the purposes of eVTOL infrastructure, it is critical to optimally manage the limited capacity of a vertiport. Therefore a need for some scheduling mechanism arises, that manages conflicts between the aircrafts.

Some current approaches to this problem revolve around heuristic methods. A First-In-First-Out algorithm is analysed in terms of capacity and throughput by [Guerreiro et al.(2020)]. Their work revealed that under this procedure some inefficiencies can occur depending on the vertiport configurations. Another method called Insertion and local search is outlined in [Pradeep and Wei(2018)]. It works in conjunction with mixed-integer linear programming and time-advance algorithm to optimise the expected time of arrival in the context of a vertiport with a single landing pad. These methods concern the scheduling problem from the vertiport's point of view and do not involve direct vehicle control. Alternative solutions based on single-agent reinforcement learning are also proposed: in [Bertram and Wei(2020)] aircrafts learn to navigate the terminal area of a vertiport by following a pre-determined path and gradually progressing towards the centre. This system

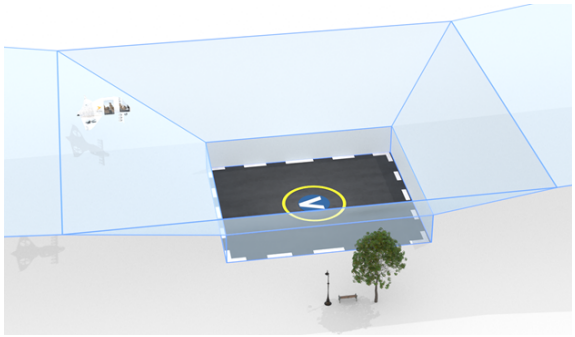

Figure 1.1: Design of a vertiport, where take-off and landing can be performed in both directions, proposed by [EASA(2022)]

relies on commands to proceed sent by a vertiport capacity manager.

A common thread that unites previously discussed approaches is the existence of some centralised entity that regulates the order of operations in the proximity of the landing pad. At the same time, the eVTOL systems are also expected to possess some level of autonomy and ability to navigate high-density traffic, especially in the areas close to the vertiports. Hence, research on the capabilities of self-organising systems, that do not rely on external commands, is motivated. Provided that independent aircrafts should cooperate to maximise the total utility of all parties involved, MARL-based methods can be suggested as a solution to this problem.

Directly utilising a decentralised MARL formulation is a possible and straightforward way to implement such a system. Still, it is reasonable to assume that some form of inter-agent communication can assist in conflict resolution. The benefits of communicative approach might include increased cohesion in collective decision making and better utilisation of the vertiport resources. However, it can also require a substantially increased learning complexity and it might not be clear if a successful goal achievement can be attributed to the information sharing. Therefore, the effect of communication on a decentralised multi-agent system needs to be analysed.

In this light, the research is carried out to study the nature of emergent communication in MARL, in particular in application to the vertiport terminal area control. The key contribution consists of three factors. First, the applicability of decentralised and communication-based MARL methods to the vertiport arrival scheduling is examined. Second, distinct approaches to priority management, that promote unique types of shared information and allow to establish the impact of communication, are proposed. Finally, self-organisation of the agents in the terminal area of a vertiport is studied, and, with the help of suggested approaches, the essence behind the developed communication protocols is investigated.

The content is structured as follows. In chapter 2 a necessary background in multi-agent reinforcement learning is introduced, approaches to implementing communicative systems within it are reviewed, and applications of reinforcement learning to the vertiport's terminal area management are summarised. In chapter 3 the details of a vertiport environment implementation are outlined and priority management schemes, that facilitate inter-agent information exchange, are described. In chapter 4 simulation results are presented, self-organising capabilities of the MARL-based system are evaluated, and the impact and nature of communication in a said system is analysed. In chapter 5 conclusions are drawn and future research directions are suggested.

2 BACKGROUND

2.1 REINFORCEMENT LEARNING BASICS

Reinforcement Learning is the paradigm of machine learning that can be described as learning-by-doing. The learner, commonly called an agent, is placed in different situations (environment) and is supposed to take actions in these situations. As the environment is often dynamic, it can be described as a sequence of its snapshots, called states. Usually there are multiple possible options for the action in any given state, and each of them is associated with a different quantifiable utility (reward), either immediate or delayed. The learning part comes when an agent has to figure out which actions bring the most reward over the sequence of decisions. In reinforcement learning the agent does not get any explicit instructions about it, so it learns by trial-and-error.

A formalisation of such process is called Markov Decision Process (MDP). According to [Sutton and Barto(2018)], here an environment can be described by its state transition function $P : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \mapsto [0, 1]$:

$$P(s'|s, a) = \Pr\{s_t = s'|s_{t-1} = s, a_{t-1} = a\}, \tag{2.1}$$

where $s', s \in \mathcal{S}, a \in \mathcal{A}$ are states and action. This function expresses the probability of the next state given the current state and the action taken in it. Furthermore, a function $R(s, a) : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ indicates the reward the agent receives for the action a taken in the state s. From the agent's side, a policy $\pi(a|s) : \mathcal{S} \times \mathcal{A} \mapsto [0, 1]$ dictates the probability of taking an action a in the state s:

$$\pi(a|s) = \Pr\{a_t = a|s_t = s\}. \tag{2.2}$$

In this regard, the goal of the agent is to learn an optimal policy π^* such that it maximises the total expected discounted return of an episode $G_{t_0} = \sum_{t=t_0}^{t_e} \gamma^t R(s_t, a_t)$ with respect to actions. Here γ is a discount factor, that controls the importance of future rewards.

Due to the connection between the expected return and the state-action pairs traversed over the course of the episode, it is convenient to define a state-value function $V_\pi(s) : \mathcal{S} \mapsto \mathbb{R}$. It represents

the expected future return given the current state if the agent follows the policy π:

$$\begin{aligned} V_\pi(s) &= \mathbb{E}_\pi[G_t|s_t = s] \\ &= \mathbb{E}_\pi[r_t + \gamma G_{t+1}|s_t = s] \\ &= \sum_a \pi(a|s) \sum_{s'} P(s'|s,a) \left[R(s,a) + \gamma V_\pi(s')\right]. \end{aligned} \tag{2.3}$$

An action-value function $Q_\pi(s,a) : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ can be defined in the same way, with the exception of additional conditioning on the action:

$$\begin{aligned} Q_\pi(s,a) &= \mathbb{E}_\pi[G_t|s_t = s, a_t = a] \\ &= \sum_{s'} P(s'|s,a) \left[R(s,a) + \gamma \sum_{a'} \pi(a'|s') Q_\pi(s',a')\right]. \end{aligned} \tag{2.4}$$

State-value and action-value functions are directly connected through the policy:

$$V_\pi(s) = \sum_a \pi(a|s) Q_\pi(s,a).$$

A useful property of value functions is that they define a partial order over the set of policies. This implies the existence of some policy π^* that is not weakly dominated by any other policy π, i.e. in any given state s the expected future return $V_{\pi^*}(s)$ can not be increased. Such a policy is called optimal.

2.2 ALGORITHMS

As [Sutton and Barto(2018)] suggest, the main goal in reinforcement learning is to find an optimal policy π^* that maximises the expected return $G_{t_0} = \sum_{t=t_0}^{t_e} \gamma^t R(s_t, a_t)$, where $a_t \sim \pi^*(a_t|s_t)$. There are different ways to solve this problem, which can be roughly grouped into two approaches: action-value methods and policy-based methods. An example of an action-value method is Q-Learning [Watkins and Dayan(1992)]. In these methods the policy is not learnt explicitly. It is instead derived from the action-value function, i.e. $\pi(a|s) = \max_a \mathbb{E}_\pi[G_t|s_t = s, a_t = a] = \max_a Q(s,a)$ in the case of Q-Learning. In this case an optimal policy will be achieved once the estimates for expected returns are close enough to their true values.

On the contrary, methods that directly map a state (or observation) to the action without necessarily relying on the estimates of expected returns are called policy-based. Here policy $\pi(a|s,\theta)$ is some function parameterised by the set of values $\theta \in \mathbb{R}^p$. For the environments with high-dimensional state-spaces these functions are generally artificial neural networks, which are universal function approximation mechanisms. In order to obtain an optimal policy it is necessary to optimise the parameters θ based on some performance measure $J(\theta)$, which is typically done by the gradient descent scheme:

$$\theta_{i+1} = \theta_i - \alpha \nabla J(\theta_i).$$

Generally $J(\theta)$ is chosen to be some fitness function that reflects the performance in terms of acquired rewards and thus needs to be maximised.

2.2.1 Policy gradient

Policy gradient methods are based on an important fact called policy gradient theorem. It states that for $J(\theta)$ defined to be the expected return of the whole episode $\mathbb{E}_{\pi_\theta}[G_0]$, i.e. a state-value function of the initial state $V_{\pi_\theta}(s_0)$, the gradient w.r.t. θ can be calculated as follows:

$$\nabla J(\theta) = \nabla \mathbb{E}_{\pi_\theta}[G_0] \propto \sum_s \mu(s) \sum_a Q_{\pi_\theta}(s,a) \nabla \pi(a|s,\theta), \tag{2.5}$$

where $\propto$ means "proportional to", $\mu(s)$ is the probability of the state s under policy π_θ and $Q_\pi(s,a)$ is the action-value function. This formulation is quite convenient since it allows to compute the gradient without explicitly knowing the state transition function of the environment.

It is possible to substitute the true expectation of states s and actions a under policy π with their Monte-Carlo estimate from the unrolled episode $\{S_t, A_t, R_t | t = 0, \dots, t_e\}$ and derive a basic policy gradient method called REINFORCE [Williams(1992)]:

$$\nabla J(\theta) = \mathbb{E}_{\pi_\theta}\left[G_t \nabla \ln \pi(A_t|S_t,\theta)\right] \tag{2.6}$$

$$\theta_{i+1} \leftarrow \theta_i + \alpha G_t \nabla \ln \pi(A_t|S_t,\theta_i) \tag{2.7}$$

A generalisation of REINFORCE involves the introduction of the "baseline function". It aims at decreasing variance in the gradient updates but keeps the expectation of the updates unchanged. Furthermore, a state-value function can be used as a baseline, in which case it becomes possible to learn it alongside the policy. Following [Sutton and Barto(2018)], if the value function is parameterised by some set of weights $\mathbf{w}$, then the update rule for REINFORCE with baseline is:

$$\delta = G_t - V(S_t, \mathbf{w}_i) \tag{2.8}$$

$$\mathbf{w}_{i+1} \leftarrow \mathbf{w}_i + \alpha^{\mathbf{w}} \delta \nabla V(S_t, \mathbf{w}_i) \tag{2.9}$$

$$\theta_{i+1} \leftarrow \theta_i + \alpha^{\theta} \delta \nabla \ln \pi(A_t|S_t,\theta_i) \tag{2.10}$$

It is worth noting that such update for $\mathbf{w}$ is natural and follows from the definition of the state-value function, as it corresponds to minimising the L_2 distance between the return G_t and state-value $V(S_t, \mathbf{w}_i)$. It is true since $\alpha_1^{\mathbf{w}} \nabla (G_t - V(S_t,\mathbf{w}))^2 = \alpha_2^{\mathbf{w}} (G_t - V(S_t,\mathbf{w})) \cdot \nabla V(S_t,\mathbf{w})$.

2.2.2 Actor-critic

A common trick used in many reinforcement learning algorithms involves using some bootstrapped estimates for the expected return instead of the actual value. This technique revolves around the recurrence property of the value functions and is called temporal difference. As outlined by [Foerster(2018)], in policy gradient methods, when it is applied to the REINFORCE with baseline, the algorithm is known as actor-critic method:

$$\begin{aligned} \theta_{i+1} \leftarrow & \theta_i + \alpha \left(G_t - V(S_t, \mathbf{w_i})\right) \nabla \ln \pi(A_t|S_t,\theta_i) \\ = & \theta_i + \alpha \left(R_t + \gamma V(S_{t+1}, \mathbf{w_i}) - V(S_t, \mathbf{w_i})\right) \nabla \ln \pi(A_t|S_t,\theta_i). \end{aligned} \tag{2.11}$$

Here actor and critic refer to the learn policy and value functions respectively. The weights for the critic network can be updated in a similar way as in 2.9. In this case it would be equivalent to minimising the L_2 distance between R_t and $(\gamma V(S_{t+1}, \mathbf{w_i}) - V(S_t, \mathbf{w_i}))$. However, it is also possible to use a separate target network for the bootstrapped estimates, with its parameters being periodically updated to equal critic's $\mathbf{w}$.

2.2.3 Proximal Policy Optimization

While policy gradient methods have the benefit of simplicity, they have some practical downsides. In particular, they can have poor data efficiency and robustness. Performing multiple update steps on the same episode data with the vanilla policy gradient update can cause the policy to suddenly significantly diverge. This necessitates either smaller parameter updates, or sampling larger batches of data in order to produce more accurate estimates for the gradient. To address these issues, Proximal Policy Optimization (PPO) [Schulman et al.(2017)] was introduced.

The main idea behind this algorithm is to limit the magnitude of policy updates and make several sequential gradient descent steps on the same batch of data. This is achieved by first introducing a "surrogate" objective $L(\theta)$:

$$L(\theta) = \mathbb{E}\left[\frac{\pi(A_t|S_t,\theta)}{\pi(A_t|S_t,\theta_{old})} D_t\right] = \mathbb{E}\left[r_t(\theta) D_t\right], \tag{2.12}$$

where D_t is the advantage at time stamp t (described further), and $r_t(\theta)$ is the probability ratio between the current policy π_θ and the policy in the beginning of the update step $\pi_{\theta_{old}}$. Second, this objective is clipped in some ϵ-interval $[1-\epsilon, 1+\epsilon]$ of the probability ratio $r_t(\theta)$, where ϵ typically assumes the values in the range $[0.1, 0.3]$. Depending on the sign of the advantage D_t it then becomes:

$$L^{CLIP}(\theta) = \mathbb{E}\big[\min\{r_t(\theta)D_t, \quad \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)D_t\}\big]. \tag{2.13}$$

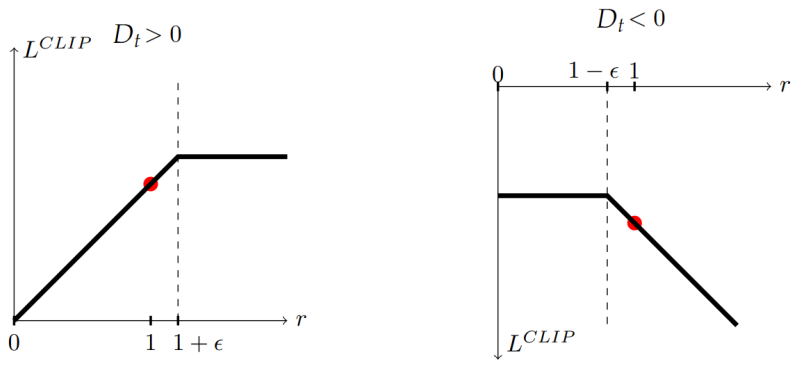

Figure 2.1: L^{CLIP} as the function of probability ratio r, depending on the sign of advantage D_t, [Schulman et al.(2017)]

Compared to the unclipped version, the objective L^{CLIP} does not provide any additional benefit for the large changes in the policy that move it towards larger values. At the same time, it leaves the changes that decrease the objective L^{CLIP} in their full scale. Intuitively, since it is a lower (pessimistic) bound of the unclipped objective, it puts more emphasis on eliminating samples with low objective value, making the algorithm more conservative in its policy updates.

In the surrogate objective 2.12 the advantage D_t is the measure of the utility of action A_t compared to the expected return from this state, i.e. its state-value function $V(S_t)$. It can be calculated over some time horizon $T - t$:

$$D_t = -V(S_t) + R_t + \gamma R_{t+1} + \cdots + \gamma^{T-t+1} R_{T-1} + \gamma^{T-t} V(S_T). \tag{2.14}$$

PPO uses a generalised version of such advantage, which is calculated as:

$$D_t = \delta + (\gamma\lambda)\delta_{t+1} + \cdots + (\gamma\lambda)^{T-t+1}\delta_{T-1}, \tag{2.15}$$

$$\text{where} \quad \delta_t = R_t + \gamma V(S_{t+1}) - V(S_t). \tag{2.16}$$

For training, an environment is unrolled for some number of timesteps T to create a batch of data. A surrogate loss is then constructed and optimized for some number of epochs K. It is possible to run multiple updates on the same data because a clipped surrogate loss function 2.13 limits the large policy updates, but keeps the small ones intact. This allows the algorithm to traverse faster through the regions of insignificant policy changes, thus improving sample efficiency.

2.3 MULTI-AGENT REINFORCEMENT LEARNING

Theoretical framework described previously addresses the setup with one agent in the environment. It can also be generalised to account for multiple decision makers in the system. In stochastic games [Shapley(1953)] N agents sequentially take actions, that influence the state of the environment. According to [Foerster(2018)], it can be viewed as agents forming a joint action $\mathbf{a_t} = \langle a_t^1, \ldots, a_t^N \rangle$ in a given state s_t, with the next state dictated by the state transition function $P(s'|s, \mathbf{a})$. In this case the joint action space $\boldsymbol{\mathcal{A}} \equiv \mathcal{A}_1 \times \cdots \times \mathcal{A}_N$ becomes a cartesian product of the action spaces of all decision makers in the environment. The reward function $R(s, \mathbf{a}) : \mathcal{S} \times \boldsymbol{\mathcal{A}} \mapsto \mathbb{R}^N$ now maps state s and joint action $\mathbf{a}$ to a vector $\mathbf{r} = \langle r^1, \ldots, r^N \rangle$ containing rewards for each agent.

2.3.1 Centralised and decentralised control

In a multi-agent setting policy can be similarly defined on the joint action space $\pi(\mathbf{a}|s) : \boldsymbol{\mathcal{A}} \times \mathcal{S} \mapsto [0, 1]$:

$$\pi(\mathbf{a}|s) = \Pr\{a_t^1 = a^1, \ldots, a_t^N = a^N | s_t = s\}. \tag{2.17}$$

Such definition implies a centralised unit that controls all agents in the environment at the same time. This might be suboptimal depending on the setup, e.g. in a competitive environment, where one agent achieves larger rewards at the expense of the other agents' losses in performance. Furthermore, regardless of the setting, joint action space grows exponentially with the amount of agents and can become intractable. In order to address this challenge it is possible to consider decentralised control, in which agents act independently. Here each agent $d \in \mathcal{D} = \{1, \ldots, N\}$ has their own policy $\pi^d(a^d|s)$, which is the same as in the single-agent formulation. [Foerster(2018)]

suggests that, provided the agents are independent, the probability of a joint action $\mathbf{a_t}$ can be factorised as

$$\Pr(\mathbf{a_t}|s_t) = \prod_d \pi^d(a_t^d|s_t). \tag{2.18}$$

2.3.2 Partial observability

Another important aspect in MARL is partial observability. It encapsulates the idea of agents not having access to the true state of the environment s_t. In decentralised partially observable Markov decision processes (Dec-POMDP) [Oliehoek et al.(2008)] they instead rely on some local representation of the environment $o^d \in \mathcal{O}^d$, called an observation, which is governed by the observation function $O(\mathbf{o}|s, \mathbf{a}) : \boldsymbol{\mathcal{O}} \times \mathcal{S} \times \boldsymbol{\mathcal{A}} \mapsto [0, 1]$. Here $\boldsymbol{\mathcal{O}} \equiv \mathcal{O}_1 \times \cdots \times \mathcal{O}_N$ is a joint observation space. Similarly to the state transition function 2.1, observation function O represents a probability of receiving a joint observation $\mathbf{o}$ given a joint action $\mathbf{a}$ taken in a state s. In a partially observable environment the policy becomes a mapping from the observation space into the action space of an individual agent $\pi^d(a^d|o^d) : \mathcal{O}^d \times \mathcal{A}^d \mapsto [0, 1]$.

2.3.3 Multi-agent learning algorithms

In order to learn an optimal policy in MARL, some additional details have to be considered, as compared to the single-agent setting. In the most straightforward case of centralised control in a cooperative scenario with a common goal, an optimal policy can be achieved with the same methods as in the single-agent environment, since the only thing that changes is action space dimensionality. On the other hand, in decentralised control with partial observability each agent is independent and can receive separate rewards. It is possible for them to optimise their own policies independently as well, in which case they maximise their rewards by considering other agents as external parts of the environment. The adaptations of algorithms like Q-Learning and Actor-Critic for this setup are called Independent Q-Learning and Independent Actor-Critic.

One training technique that can be used in cooperative settings is parameter sharing. Instead of having individual network parameters θ^d for each agent, a universal set of parameters $\theta = \theta^d \forall d \in \mathcal{D}$ can be learned. The benefit is twofold: first, the inference time can be reduced by performing calculations for all agents simultaneously in one batch. Second, since the sampled trajectories all belong to the same set of parameters, gradients can be estimated with less variance, thus improving sample efficiency of the learning process.

In general, the essence behind Dec-POMDP is that the policy of each agent has to be conditioned only on local observations. This means that global information like environment state s_t or joint observation $\mathbf{o}_t$ can be used during training, as long as the policy does not depend on it. This approach is called centralised training with decentralised execution (CTDE). Different CTDE approaches, like Multi-agent deep deterministic policy gradients (MADDPG) [Lowe et al.(2020)] and Counterfactual multi-agent policy gradient (COMA) [Foerster et al.(2017)] use an actor-critic architecture and with a centralised critic, that is conditioned on joint observation and action vectors $\mathbf{o}_t$ and $\mathbf{a}_t$. These methods were empirically shown to provide better performance and faster convergence over Independent Actor-Critic on different benchmarks.

The algorithms described above use a replay buffer that stores past episode data, in order to re-use it during the training phase. This makes these algorithms off-policy since they are learning on the transitions, that were not sampled by the current policy. Due to MARL environments involving a large degree of non-stationarity incurred by the agents, it might be beneficial to also consider on-policy algorithms, e.g. PPO.

Similarly to the vanilla Actor-Critic, PPO can also be adapted to the multi-agent scenario. Independent PPO (IPPO) and Multi-agent PPO (MAPPO) represent two different approaches - the first is fully decentralised, and the second operates in CTDE paradigm and has a centralised critic. Despite theoretical concerns of being less sample efficient, both IPPO and MAPPO have been shown to outperform off-policy methods on multi-agent benchmarks [Papoudakis et al.(2021)], [Yu et al.(2022)], [de Witt et al.(2020)].

2.4 COMMUNICATION IN MULTI-AGENT REINFORCEMENT LEARNING

In MARL with decentralised control and partial observability agents often get access to different parts of information about the environment. Since the agents are decentralised, they would normally be unable to share the information, which in some cases can be crucial for correct decision making. Allowing for information exchange between decentralised agents, i.e. communication, can be an efficient way to set up better coordination in a multi-agent system. It is possible to share information in a pre-determined way: in robot football the agents communicate their position and proximity to the ball at every time step to every member of the team [Stone and Veloso(1998)]. However, such a structured approach does not need to be the case. While agents are learning control in their environment, they are also able to independently learn different communication schemes that might impact their overall performance.

In order to set up communication in a multi-agent system, a number of conceptual questions need to be answered. *How* the agents are communicating, *who* they are sending the messages to, *what* information they are transmitting, etc. A convenient way to structure such design decisions was proposed in [Zhu et al.(2022)], which considers various technical aspects that characterise communication approaches.

2.4.1 Communicator type

A straightforward way of enabling information sharing between the agents in the system would be direct communication. In RIAL and DIAL [Foerster et al.(2016)] at time step $t-1$ each agent in addition to taking some action a^d_{t-1} also emits some message m^d_{t-1}, which all other agents have access to. Then, at the next time step t each agent can condition their policy π^d and communication network M^d on the joint message $\mathbf{m}_{t-1} = \langle m^1_{t-1}, \dots, m^N_{t-1} \rangle$. A new action and message are then $a^d_t \sim \pi^d(a^d_t | o^d_t, \mathbf{m}_{t-1}, x^d_{t-1})$ and $m^d_t = M^d(o^d_t, \mathbf{m}_{t-1}, x^d_{t-1})$ respectively, where x^d_{t-1} is all additional input, like hidden state of the recurrent network, agent index and previous action. In this approach an agent can convey some information to all other agents in the system directly.

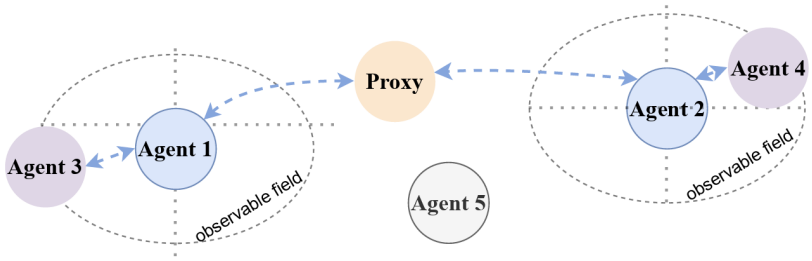

Figure 2.2: Different communication structures in one system, [Zhu et al.(2022)]

An alternative approach consists in sending a message via some proxy. In such setups proxy is an auxiliary agent who conducts communication among other agents, but does not have any immediate impact on the environment itself. A proxy can transform the messages and adapt them to a particular purpose. HAMMER [Gupta et al.(2022)] uses a central proxy that collects observations from all agents and then sends personalised messages to them. It can be viewed as a proxy agent forming a joint message $\mathbf{m}_t = M(\mathbf{o}_t)$ based on a joint observation $\mathbf{o}_t$, and each individual agent taking an action a_t^d according to $\pi^d(a_t^d|o_t^d, m_t^d)$.

2.4.2 Communication structure

A different angle to consider is the structure of the communication graph. Generally it can be divided into two categories: predefined and learnable. Full direct communication, like the one described in DIAL, has an immutable structure and can be viewed as a complete communication graph. The approach used in TarMAC [Das et al.(2020)] revolves around the same concept of full connectivity. There the agents produce messages, which consist of two parts: signature k^d and value v^d. At the receiving end, each agent predicts their private query q^d, in order to compute attention weights α^{di} based on the signatures of all other agents k^i. Then the integrated message c^d for the agent d is a weighted combination of all message values v^i : $c^d = \sum_{i \in \mathcal{D}} \alpha^{di} v^i$. Although each agent independently decides how to deal with the incoming messages, they are communicating with everyone at each time step.

A predefined structure does not necessarily imply full connectivity among the agents. Some models rely on predetermined rules on how to control the flow of information. In DGN [Jiang et al.(2020)] communication is only allowed between the neighbouring agents, which is defined by their proximity. Over time the agents can enter or exit the neighbourhood of the other agents, changing the structure of the communication graph.

Another approach to the communication policy is to learn it. In the simple case, an agent can individually decide if it communicates at a given time step. This idea is used in IC3Net [Singh et al.(2018)], where a gating mechanism controls the information flow from the agent. Providing even more flexibility to build a communication graph, ATOC [Jiang and Lu(2018)] allows the agents to form arbitrary communication groups. Every T time steps an agent can decide to initiate communication with any number of collaborators, which it can select itself. This communication group $\mathcal{G}$ lasts for T time steps, and within it an integrated message $\tilde{h}_t^i$ for each agent is generated based on the hidden states of participating agents $\mathbf{h}_t = \{h_t^j | j \in \mathcal{G}\}$, i.e. $\tilde{\mathbf{h}}_t = g(\mathbf{h}_t)$. Furthermore, an agent can be a member of multiple groups, thus bridging the information gap between different

groups. With learnt communication policy the agents can dynamically adjust their information flow according to the situation in the environment, which can be beneficial in certain scenarios.

2.4.3 Other characteristics

Approaches to communication in MARL have some further noteworthy characteristics. To begin with, the way that the messages are integrated into the agents' network can determine the capabilities of a system. The incoming messages can be concatenated, as is done in the DIAL model. This approach provides the agents with full control over the information, since it is not lost in the process, and the agents can flexibly learn which messages are important. However, it comes at the cost of expanding the input space and fixing the number of agents that communicate. An alternative solution would be forming some sort of an aggregated incoming message. Here the messages can be equally valued, as it is done in IC3Net, or unequally valued, like in TarMAC. The former case does not introduce any particular preference towards the senders and treats them as identical. The latter assigns weights to the incoming messages, which can be done by attention mechanisms. This has the benefit of focusing on the relevant information but can be challenging to learn.

Questions of real world applicability also influence design decisions of Comm-MARL systems. For this reason some approaches impose limitations on the communication bandwidth. GAXNet [Yun et al.(2021)] is designed to be a system for air-to-ground ultra-reliable and low-latency communication for UAVs. It uses an attention mechanism to optimise the communication, so that it satisfies the latency and error rate constraints. In [Inala et al.(2021)] authors also address the problem of messaging optimisation with a similar method. They train a transformer network and use its attention weights to cut off redundant branches in the communication graph. This new communication policy can significantly reduce the amount of necessary message passing during inference, which makes it more viable for realistic scenarios.

Finally, several other details determine the structure of communication in MARL. This includes the type of messages that are transmitted: existing knowledge, imagined future actions, environment dynamics, etc. Learning scheme, whether it is fully decentralised learning or CTDE, can have an impact on the effectiveness of information exchange. All in all, Comm-Marl systems have a lot of flexibility in their implementations, and each has to be adjusted according to the application scenario.

2.4.4 IC3Net

IC3Net [Singh et al.(2018)] is one of the approaches in MARL that implements communication among the agents in the system. This model uses direct communication with a learnable structure, where the agents can individually decide to not share any information. Furthermore, it treats incoming messages as equally valued, thus being suitable for systems that have uniform agent types. It was introduced as an upgrade to the CommNet [Sukhbaatar et al.(2016)] to be applicable to semi-cooperative and competitive scenarios, and to address the credit assignment problem. To enable this, IC3Net trains its model based on the individual rewards for each agent. This naturally

makes it fitting also for cooperative settings, where agents have explicit individual components in the reward function.

The central idea of this approach is a gating mechanism, that provides the agents with control over sending out their messages. At any time step t, the agent j takes a binary action $g_j^{t+1} \in \{0, 1\}$ that determines whether it is going to communicate on the next time step or not (equation 2.19). It then updates its hidden state h_j^{t+1} and receives a message c_j^{t+1} from all other communicating agents in the environment. Although the incoming message is calculated as an average of hidden states times some weight matrix C, it can also be viewed as the average of the sent messages. From this angle, a message sent by an agent j' would be equal to $C \cdot (h_{j'}^{t+1} \odot g_{j'}^{t+1})$.

$$g_j^{t+1} = f^g(h_j^t) \tag{2.19}$$

$$h_j^{t+1}, s_j^{t+1} = \text{LSTM}(e(o_j^t) + c_j^t, h_j^t, s_j^t) \tag{2.20}$$

$$c_j^{t+1} = \frac{1}{J-1} C \sum_{j' \neq j} h_{j'}^{t+1} \odot g_{j'}^{t+1} \tag{2.21}$$

$$a_j^t \sim \pi(a_j^t | h_j^t), \tag{2.22}$$

f^g is an arbitrary map from hidden state h_j^t into binary action g_j^{t+1}, and $e(\cdot)$ is some observation encoder.

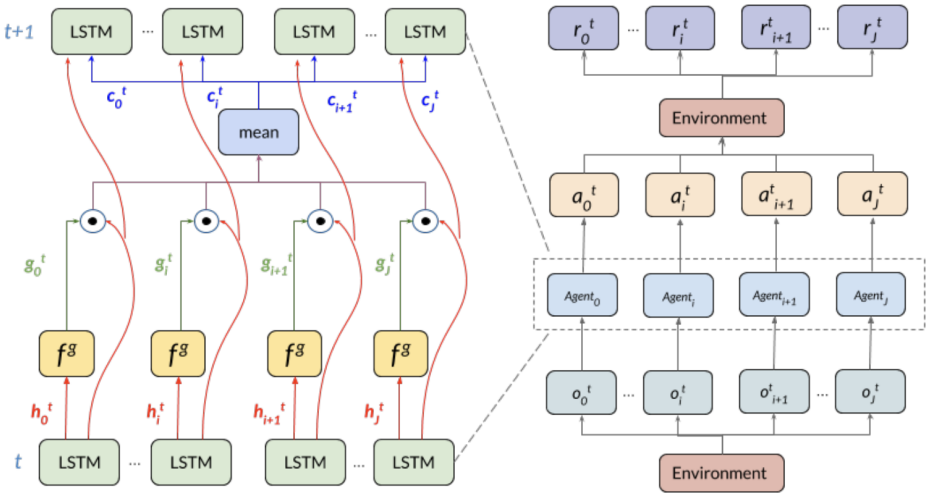

Figure 2.3: Architecture of IC3Net model, [Singh et al.(2018)].
Left: single communication step. **Right:** overview of the training procedure.

IC3Net was tested in different scenarios - cooperative (traffic junction), mixed and competitive (predator-prey). The results show that the agents prefer to communicate only when it is profitable. For example, the prey naturally learns to never communicate with the predators. In the competitive version of predator-prey, where the reward is distributed among those predators who have reached the prey, communication only happens in the initial stages for better exploration, and then it stops from the predators who reached the prey. Furthermore, in cooperative scenarios agents always learnt to communicate. The authors suggest fixing the gating outputs to 1 to speed up the training process in cooperative settings.

2.5 URBAN AIR MOBILITY

Urban Air Mobility (UAM) is an emerging branch of transportation in which automated air vehicles realise passenger transit, goods delivery and emergency systems in dense metropolitan areas [Cohen et al.(2021)]. Efficient space use in the urban context is crucial, and Vertical Take Off and Landing (VTOL) aircrafts are considered to be a viable solution to this. They require a special infrastructure that can facilitate their take-off, landing, maintenance and charging, which is called a vertiport. In reality, VTOL aircrafts are restricted to the network of vertiports, as they can not take off and land anywhere on demand. Coupled with the fact that vertiports have a limited capacity, some scheduling and spacing system in close proximity of the vertiport is necessary [Yang and Wei(2020)].

Some works addressing the arrival sequencing methods for UAM involve a linear optimisation based on particular hand-crafted constraints [Kleinbekman et al.(2018)]. To solve a sequencing problem the authors calculate an optimal required time of arrival for an aircraft. Approaches based on reinforcement learning are also suggested as possible solutions: [KrisshnaKumar et al.(2023)] make use of graph reinforcement learning to perform take-off and landing scheduling. There RL is used for the high-level control of an aircraft, representing all possible states and actions of an agent with a graph.

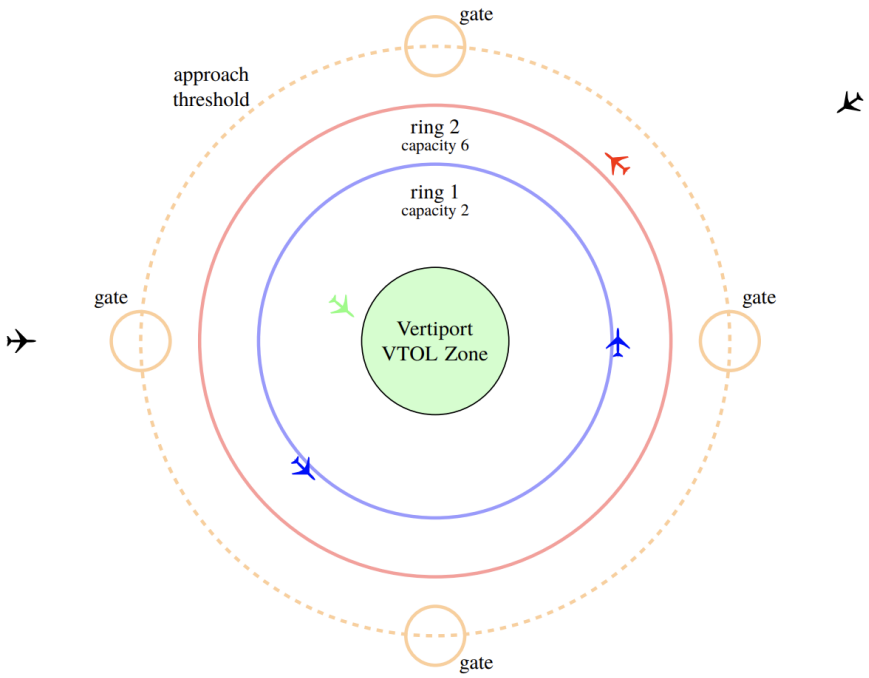

Figure 2.4: A model of a vertiport terminal area described in [Bertram and Wei(2020)]. An airspace consists of multiple concentric rings, along which the aircrafts should travel while waiting for the signal to proceed to the next level.

Moving on to RL-based aircraft control systems, in [Bertram and Wei(2020)] a vertiport terminal area controller (VTAC) is proposed, where the aircrafts learn to follow pre-determined trajectories to ensure safety. In particular, a vertiport is modelled as a sequence of concentric rings, each

with capacity determined by the radius of the respective ring. A schematic representation of this model is depicted in the figure 2.4. The agents progress towards the vertiport centre starting from the outermost ring in FIFO order, thus maintaining a strict priority. Depending on the status of an agent, VTAC places rewards in the environment at the future bearing or at the centre of the vertiport, thus giving direct instructions to follow some path. Given that the agents are controlled independently and do not contribute towards a common goal, this approach can be viewed as a single-agent MDP.

Multi-agent reinforcement learning methods are employed in various contexts within UAM. [Huang et al.(2023)] implement a strategic conflict management scheme, in which they learn to resolve the issues with overlapping trajectories of flights between different vertiports. In [Apaza et al.(2023)] authors use MARL to achieve optimal radio spectrum management and navigation at the same time. They impose requirements on the quality of signal and allow agents to change frequencies of the channels, via which they communicate with the broadcast stations.

Several approaches also make use of communication between agents. A cooperative sequential decision making process based on the Monte Carlo Tree Search is suggested in [Yang and Wei(2020)] for collision risk minimisation. There at each time step t agents sequentially sample their actions by conditioning their policy on all actions sampled by previous agents. That is, $a_t^d \sim \pi(a_t^d | o_t^d, \hat{\mathbf{a}}_t^d)$, where $\hat{\mathbf{a}}_t^d = \langle a_t^1, \ldots, a_t^{d-1}, \hat{a}_t^{d+1}, \ldots, \hat{a}_t^d \rangle$ is a vector of already sampled actions a_i and assumed baseline actions $\hat{a}_i$. Such a way of propagating the information can be viewed as intent communication. A CommNet model is used in [Park et al.(2023)] to efficiently manage a group of aircrafts travelling between multiple locations. The authors used an actual vertiport map to evaluate the applicability of their approach to the real world scenario and claimed that CommNet outperforms other methods that do not rely on communication.

3 EXPERIMENTAL FRAMEWORK

Scheduling and spacing system for the vertiport (VTAC), outlined in [Bertram and Wei(2020)], involves a centralised controller regulating the order in which the aircrafts are entering the terminal area. It operates under the first-in-first-out principle, where the agents sequentially move through the different levels of proximity to the vertiport. More than that, during training VTAC provides direct guidance on the desired trajectory by rewarding the agents for following it. It naturally requires some path calculations from the centralised controller and makes use of reinforcement learning only in the context of single-agent control. This leaves undiscovered the idea of a true multi-agent setting, where the agents collaborate and self-organise with the aim of optimising the vertiport capacity utilisation. The setup proposed further explores exactly this topic and addresses some limitations mentioned previously.

3.1 ENVIRONMENT

The suggested approach is similar to VTAC, but operates within the MARL paradigm. It focuses on solving the scheduling problem in the final stage of vertiport entry. The agents are not given any instructions about navigating the airspace, their only goal is to ensure a collision-free arrival at the centre of the vertiport. Furthermore, a centralised controller that directly dictates the orders to the agents is also removed. This change now demands that the aircrafts collectively decide on the order of entry, which is done by communication mechanisms.

3.1.1 General description

The environment is a 2d model of the vertiport's terminal area, in which each of N agents represents an aircraft, whose goal is to enter said vertiport without colliding with the other aircrafts. The environment consists of a vertiport itself - a special circular zone with some radius R_i, and an area around it with some radius $R_o >> R_i$, where the agents can roam freely. Agents should enter the goal one at a time, and once any agent enters the goal, the port "closes", i.e. a cool-down

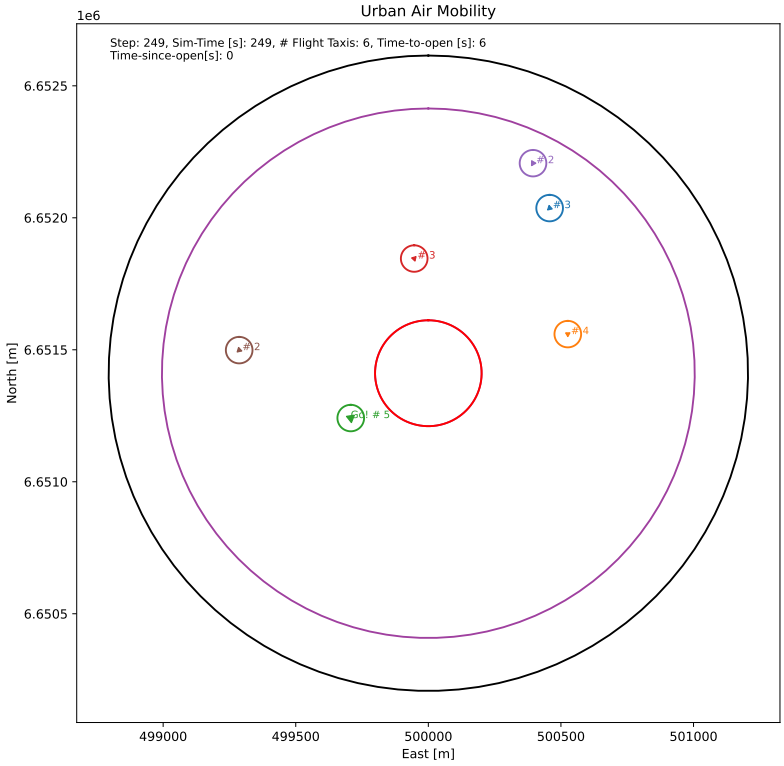

Figure 3.1: Example of the environment with 6 agents in a hidden priority setup. An occupied vertiport (red ring) is surrounded by its terminal area (purple ring) and outer boundary (black ring). Each agent (triangle) has their own separation zone around them (circles). The agent in green has the largest priority number in the group and is supposed to enter the vertiport once it becomes available. The size of the markers (triangles) indicates the power of the communicated message.

period τ_c starts in which the agents are not supposed to enter it. The agent that "closes" the goal is respawned at the edge of the terminal area, which is equivalent to the new agent entering the airspace. On top of that, if an agent moves too far away from the vertiport centre and crosses the outer boundary ring with the radius $R_r > R_o$, it is also respawned at the edge of the terminal area. The agents are supposed to travel at some distance from each other, known as a minimal separation distance D_s.

3.1.2 Reward structure

The reward function has two components: the first one reflects goal achievement, and the second one - collision risk. It can also be viewed from the angle of team impact: rewards associated with the goal are shared across all agents in the environment, while collision (or leaving the free roam area) only impacts the rewards of the agents involved.

The goal achievement reward consists of different elements. First, whenever an agent enters the open goal, everyone receives a one-time reward of +250. Second, the agents are rewarded based

on the status of the vertiport. If it is open, they receive -2.5 every time stamp, turning into +10 if it is closed. This incentivises the agents to enter the goal as soon as possible. Additionally, providing constant small rewards makes the learning process more sample efficient compared to just a single large reward for entering the goal. On top of that, the agents are punished for staying within the inner area of the vertiport while it is closed: all agents receive -1 for every agent that enters a goal while it is on cool-down. Furthermore, the cool-down does not progress while there is someone within the inner area of the goal. This is done to ensure that the agents learn not to enter the vertiport while it is occupied.

The collision risk component of the reward function is individual for each agent. It is dependent on the euclidean distance to the closest other agent in the system, and for any agent i it is calculated as follows:

$$R_{c_i} = \alpha \cdot exp\left(-\frac{\min\limits_j D_{i,j}^2}{\beta}\right),$$

where $D_{i,j}$ is the distance between the agents i and j; α and β are some positive scaling constants. In this particular instance, this reward function was scaled to give a reward of -5 at the minimum separation distance $D_{i,j} = D_s$, and a reward of -0.01 at $D_{i,j} = 5 \cdot D_s$. Whenever any two agents are closer than D_s, it is considered a Loss of separation event (LOS), or incident, and they receive a -10 reward flat.

3.1.3 Observation and action spaces

As the environment represents a simple model of a vertiport, the aircrafts have basic control capabilities. Agents are moving with a constant true air speed v_j, sampled uniformly from the range $v_b \pm \Delta v$ at the initialisation individually for each agent. The aircrafts can only influence the direction they are heading towards. The action space of each agent is discrete and consists of three available actions: turn left or right (which correspondingly changes the angle of their heading by $\Delta\phi$), or do not turn.

Observation vector [1] O_i of an agent i is a concatenation of agent-specific and external observations, i.e.

$$O_i = \left(O_i^* \, , \, O_i'\right).$$

Agent-specific information O_i^* includes the distance to and the bearing of the goal - D and Θ respectively, goal cool-down timer τ, as well as one-hot encoding of the agent id number ι, which was empirically found to improve the performance:

$$O_i^* = (D_i, \Theta_i, \tau, \iota).$$

External observations O_i' of an agent i comprise the information about all other agents $j \in \overline{1..N} \setminus \{i\}$ relative to the agent i:

$$O_i' = \left((O_{i,1}') \, , \ldots , (O_{i,N}')\right),$$

$$O_{i,j}' = \left(d_i^j, \theta_i^j, v_i^j, \phi_i^j, \bar{d}_i^j, \bar{\tau}_i^j\right),$$

[1] Time index is omitted for simplicity of notation

where $d_i^j, \theta_i^j, v_i^j, \phi_i^j$ are distance, bearing, speed difference and heading difference. In order to more efficiently address the collision avoidance, the closest point of approach (CPA) between the agents i and j is also calculated, and $\bar{d}_i^j, \bar{\tau}_i^j$ are distance and time to reach it.

3.2 EXPERIMENTS

Besides direct path following instructions, VTAC system relies upon a strict order of aircraft arrival at the vertiport. For the purpose of a cooperative multi-agent approach, this constraint needs to be lifted. Two potential ways to do so are suggested: one leaves some hidden priority, and the other removes it completely. Such distinction is motivated by the fact, that in the former case the agents have an explicit piece of information that is relevant for the whole group and thus needs to be communicated, whereas in the latter the messages can be arbitrary. By contrasting the communication protocols, derived from these setups, it would become possible to see if the agents learn to communicate the innate properties of the environment and if they are able to independently create some form of hierarchy.

The proposed approaches entail some changes to the observation space and the reward function. Furthermore, the experiments around these setups are designed to understand the importance of inter-agent communication in enabling a safe and efficient use of the vertiport capacity.

3.2.1 Hidden priority

The first setup adapts the idea of explicit priority to the communication approach. In this setting, each agent gets assigned a random number $c \in \{1, \dots, N\}$ which is called a priority number. These numbers can be sampled arbitrarily, with or without replacement across the agents, i.e. multiple agents can have the same priority number. Each agent only gets access to their own priority number through the observation space, without explicitly knowing the priorities of other agents. Their goal is to establish the agent(s) with the largest priority and have them enter the vertiport. Once any agent enters the vertiport, priority numbers are re-rolled for everyone. In fact, sampling priority numbers c^d with replacement is a more reasonable approach, since it avoids the situation where the priority number N is always the largest, thus promoting communication. In this setup, the reward for having a goal closed depends on the validity of the last agent entrance. That is, everyone receives a +10 reward for the closed goal only if the agent who entered it most recently had the largest priority, making it +0 otherwise. Note that a reward of -2.5 is still given if the goal is open. Furthermore, a one-time positive reward at the moment of entrance is also only given for the valid entrance. This encourages all agents to close the goal while largely promoting coordination in having only the highest priority agent enter it.

3.2.2 Equal priority

In the second setup an explicit priority constraint is lifted. It can also be viewed as assigning equal priority to all agents. This setup operates in the exact same way as the first one, with the exception

that here any agent can validly enter the goal. Thus every agent receives full rewards for the goal status and entrance. The motivation and difficulty behind the equal priority setup compared to the hidden priority, is that the aircrafts must come up with their own communication protocol to ensure only one of them enters the goal, while maintaining the safety of the whole cohort.

3.3 IMPLEMENTATION DETAILS

Particular values that were set during the training and simulation are presented in the table 3.1.

True air speed $v_b \pm \Delta v$, $\frac{m}{s}$	13 ± 3	Terminal area radius R_o, m	1000
Angular velocity $\Delta\phi$, $\frac{^\circ}{s}$	5	Vertiport inner area radius R_i, m	200
Minimum separation distance D_s, m	100	Outer boundary R_r, m	1200
Collision distance D_c, m	10	Cool-down time τ_c, s	30

Table 3.1: Parameters of the vertiport environment

A model that is used to control the agents in the environment is IC3Net [Singh et al.(2018)], described in the section 2.4.4. Some minor technical changes that are connected with communication action g are introduced to this model. First, the communication action g_j^{t+1} of an agent j is now computed based on the updated hidden state h_j^{t+1}, instead of the past hidden state h_j^t. This is done to ensure that communication action makes use of the observation o_j^t, received at the current time step t. Additionally, in the original paper the gating action was binary, i.e. sampled from Bernoulli distribution $g_j^{t+1} \sim Be(f^g(h_j^t))$. This naturally entails an increased degree of variance. To mitigate this issue and provide continuity in the gradients, associated with the communication mechanism, communication action is now calculated as

$$g_j^{t+1} = \sigma(f^g(h_j^{t+1})), \tag{3.1}$$

where σ is a sigmoid function. This allows the agents to flexibly adjust the importance of their messages in a continuous way.

During the training phase, m copies of the environment are rolled out for 250 timesteps, where trajectories of N agents are collected. This results in a batch size of $250 \cdot N \cdot m$. The model is trained by Independent PPO algorithm [Yu et al.(2022)], which means that a processed batch is discarded after each training step. Before a new rollout the environment is first reset. Training is done for a total of 960000 environment rollouts. On top of having two approaches to deal with priority, both setups are trained with 6 and 16 agents in the environment. It is motivated by the fact that models with different agent amounts might develop different strategies due to decreased space per agent.

The environment was implemented using an interface of Gymnasium library [Towers et al.(2023)] on top of TU Dresden RL suite [Waltz and Paulig(2022)], which provides a low-level logic of a vertiport operation. Aircraft dynamics model and physics simulation is based on the Bluesky Air Traffic simulator [Hoekstra and Ellerbroek(2016)].

4 SIMULATIONS AND RESULTS

A model of a vertiport, described in chapter 3, requires a high degree of cooperation from a group of aircrafts. To help achieve adequate collective behaviour, suggested experimental setups create a framework that encourages an information exchange between the agents. In order to quantify the effectiveness of the resulting inter-agent communication, the models obtained from both experimental setups are compared to their non-communicative versions. [1] They are evaluated in terms of safety and efficiency based on the simulation data, which was collected over 50 episodes with a length of 1000 time steps each.

The analysis is performed across 8 models in total. Two different priority management schemes are referred to as "Hidden" and "Equal", as described in the section 3.2. Setups can contain either 6 agents or 16 agents, which is indicated by "Large". Finally, NC stands for "Non-communicative" models. It might not be always meaningful to judge the performance of the models across priority or size, as they might not evolve identical behaviour patterns during training. Therefore, the most informative comparison can be made between normal and NC systems, which allows to establish the importance of communication in a given setup.

4.1 GOAL ACHIEVEMENT AND SAFETY

In this section some general information about models' performance is outlined. A comparative analysis is performed with the aim of understanding how communication impacts the utilisation of vertiport capacity and safety in a group of aircrafts.

4.1.1 Trajectories and distance to the goal

First point to consider is the self-organisation of the aircrafts in a group. In all setups the agents learn to generally navigate around the goal in a circular trajectory, however the degree of how

[1] "Non-communicative" setups refer to the models that were trained without communication from the very beginning. Arrangements, in which communication is severed in the models that were trained to rely on it, are not considered.

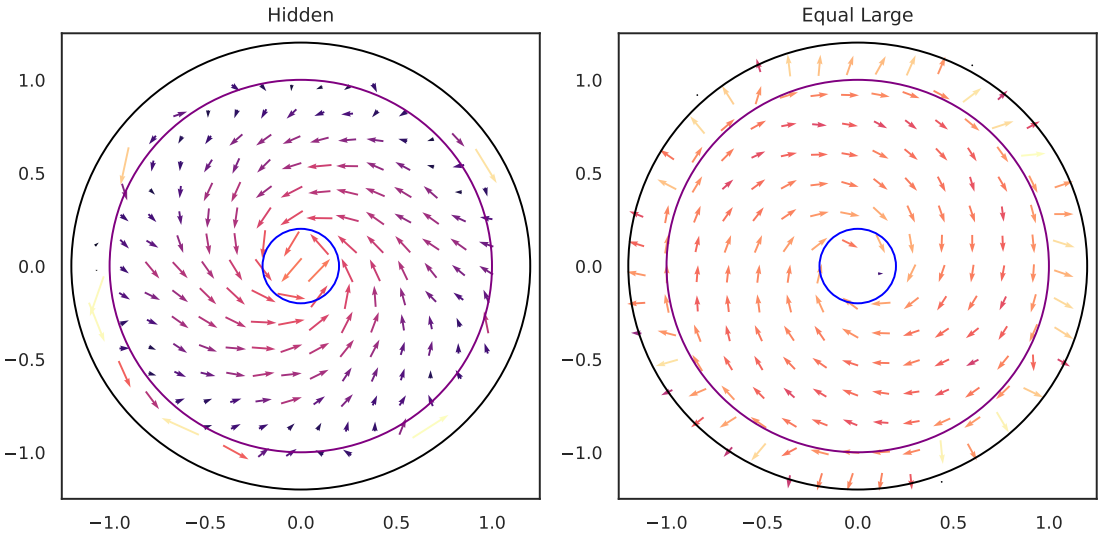

Figure 4.1: Examples of mean movement directions in the vertiport environment for Hidden and Equal Large setups. Hidden setup involved 6 agents, Equal Large involved 16. Arrow direction indicates a mean value of the agents' speed vectors at a given location, color shows the magnitude of the mean value. The agents can develop arbitrary preferred movement direction.

strictly they follow it can vary depending on the distance to the centre and the number of agents in the system. Figure 4.1 shows two examples of average movement directions within an environment, which represent a typical behaviour across all setups. Blue, purple and black circles reflect the goal, terminal area and respawn area boundaries. Arrow directions indicate the mean value of agents' speed vectors at a given location, and colour shows the magnitude of the mean value.

As the agents do not have any particular guidance on their desired trajectory, they can learn to move in either clockwise or counter-clockwise directions arbitrarily. More than that, in the Hidden setup when the agents are far from the goal they do not have a preferred trajectory at all, which can be concluded from the small magnitude of the mean direction vector. A tendency to move at some particular angle becomes more pronounced in the areas closer to the centre. At the same time, there exists little deviation in the movement trajectories regardless of the distance in the Equal Large setup. Such strictly organised behaviour can be attributed to safety considerations, since with a bigger number of agents there is less room for manoeuvre. Furthermore, for the same reasons here the aircrafts approach the vertiport more gradually, than in the case of a Hidden setup, which can be concluded by comparing the relative bearings towards the centre at corresponding locations.

Due to the radial symmetry of movement angles, evident from the figure 4.1, a preferred distance to the vertiport centre does not depend on the azimuth towards it. A distribution of these distances is presented in figure 4.2, measured as a proportion to the terminal area radius.

A common observation can be pointed out that all models learn to roam close to the edges of the terminal area, implied by the peaks around the distances of 0.95 units. This behaviour is significantly more prominent in the Large setups and the same explanation of safety motives applies. A larger preference for travelling near the border is also exhibited by the NC setups, compared to the communicative analogues. In this case it can be associated with worse coordination capabilities and resulting reliance on some strict movement trajectories.

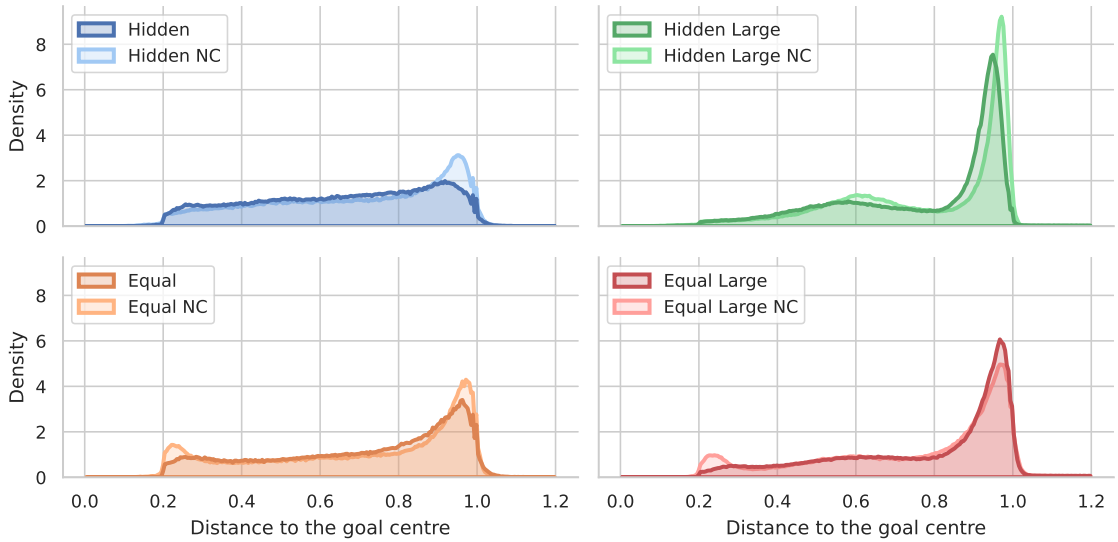

Figure 4.2: Distributions of distances to the vertiport centre across different setups. NC stands for non-communicative model. Peaks around 0.25 and 0.9 hint that the agents travel in circular trajectories with such radii.

Equal priority allows the agents to develop a different strategy, in which they travel in a tight circle around the goal while waiting for it to open. It is reflected by an increase in density close to the vertiport boundary of 0.2. Additionally, Large setups have another local maximum in the midpoint of 0.6, which is most noticeable in Hidden Large NC. This multimodality of distance distributions implies that the agents self-organised to navigate around the goal in multiple rings. This emergent phenomenon validates the original design decision with multiple levels of proximity around the vertiport centre, described in [Bertram and Wei(2020)].

4.1.2 Vertiport capacity utilisation

Utilisation of the vertiport capacity should also be taken into account. To achieve proper efficiency the waiting time before an agent enters the goal should be as small as possible, taking into account the safety of travel. Table 4.1 provides an overview on the entrance rate and average waiting time of the systems. It captures the percentage of time the vertiport is open, the mean amount of time steps between the opening of the goal and any agent entering it, and the number of such entrances per 1000 time steps. In addition, number of closed entrances per 1000 time steps counts the number of situations, in which an aircraft attempted to enter the goal while it was closed.

It can be seen that the introduction of communication improves the performance in the large setups, which is reflected by the decrease in waiting times of 35 % and 25 % for the Hidden Large and Equal Large setups respectively. The same can not be said about the regular ones, where NC models have only slightly smaller waiting times. Number of goal entrances directly impacts open time percentage and mean waiting time, and naturally paints the same picture. Such difference in utilisation can be attributed to the fact that in a communicative approach the agents tend to travel towards the centre in a direct path, whereas in the NC they gradually approach the goal while keeping the circular formation. This behaviour is more prominent in the Hidden setups. As mentioned previously, with Equal priority some aircrafts are always located close to the vertiport

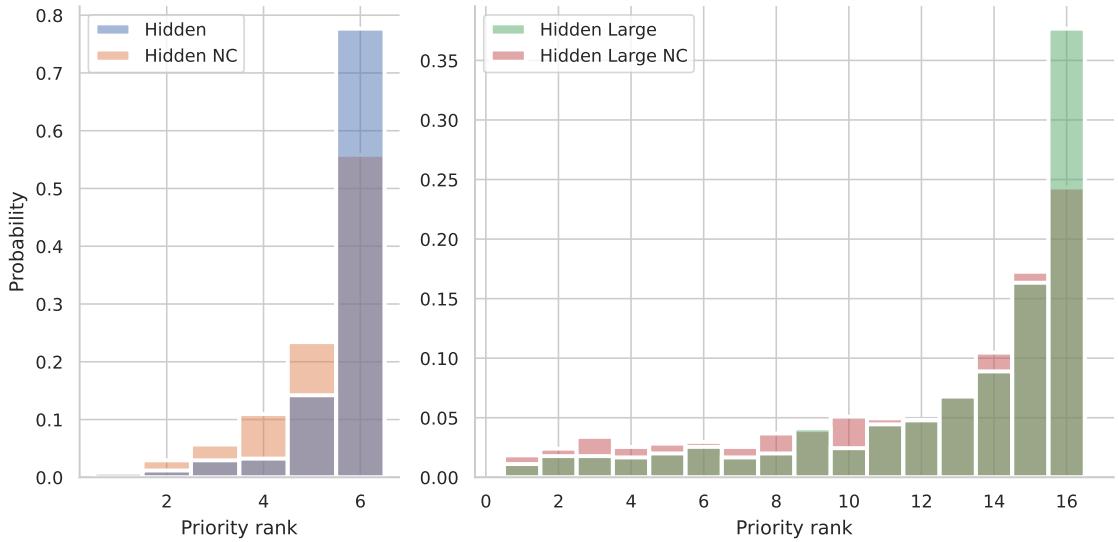

Figure 4.3: Arrival frequencies by priority rank in Hidden setups. NC models learn some innate bias, but communicative models can capture it more fully.

Setup	Open %	Waiting time	# Entrances	# Closed entrances
Hidden	36.2	17.5	20.4	1.0
Hidden NC	36.9	18.5	19.7	2.0
Equal	20.3	7.8	25.9	0.4
Equal NC	21.1	8.4	25.2	2.6
Hidden Large	41.3	22.7	18.0	2.1
Hidden Large NC	50.4	34.3	14.1	3.4
Equal Large	21.0	8.2	25.7	0.8
Equal Large NC	25.4	10.9	23.8	4.2

Table 4.1: Entrances and waiting times

and are prepared to enter once possible, which results in significantly smaller waiting times for these approaches.

At the same time, number of closed entrances is substantially impacted by the absence of communication. In Equal setups, both regular and large, the number of such occurrences increases by more than 5 times, while in Hidden the increase is twofold. A common trajectory pattern in NC setups takes place, when multiple aircrafts thrust towards an open vertiport simultaneously, and only one can arrive first. Since the agents do not exchange the information, they can not estimate when the goal will close, which leads to them being unable to adjust in time and avoid entering the goal. This problem is exacerbated by the fact that with Equal priority all agents are encouraged to enter the goal. In Hidden setups the agents are generally biased against entering the goal if they have a lower priority, as shown in figure 4.3, thus reducing the amount of potential clashes. In general, communicative models are more capable of dealing with such conflicts and exhibit the described behaviour less frequently.

Figure 4.3 presents the frequencies of entrances in the Hidden setups by the priority rank of the agent p^j. As described in section 3.2, each agent gets assigned some priority number $c^j \in \{1, \ldots, N\}$, which in the shown example was individually sampled from the binomial distribu-

tion $c^j \sim B(N-1, \frac{1}{2}) + 1$ across all episodes. Then priority rank is defined as $\langle p^1, \dots p^N \rangle = \text{Rank}(\langle c^1, \dots c^N \rangle)$. Considering that in the Hidden setup the agent with the largest priority gets a bonus reward for entering the goal, the NC models are only able to learn some bias towards bigger priority numbers c^j. From the increase in entrance frequencies of the highest-priority agents, it can be concluded that communicative setups can recognise the correct hierarchy. It is worth noting that non-max priority entrances still happen due to the base reward for keeping the vertiport occupied.

4.1.3 Safety

Goal arrival is not the only relevant aspect, as the safety of navigation is a crucial topic in the vertiport environment. From this angle, the aircrafts need to travel at a certain interval from each other to allow for some manoeuvring. Figure 4.4 provides the information on this matter. It describes the distribution of distances from an agent to the closest other agent, measured in unit distance, normalised by the terminal area radius. Furthermore, solid and dashed lines indicate the incident distance (Loss of separation event, violation of safety measures) and accident distance (collision), equal to 0.1 and 0.01 respectively. In table 4.2 frequency of such events is outlined. It is expressed as the percentage of times when the closest agent is located at less than incident or accident distance.

From the graph it can be seen that Large setups are naturally characterised by smaller intervals between the agents. In addition, the fact that the peaks of their distributions are more sharp tells that the aircrafts are spread more evenly across the terminal area, i.e. they keep consistent distances from each other. In Equal priority setups the absence of communication does not significantly impact the distribution, except in the region close to incident distances. As the table 4.2 shows, with Equal NC and Equal Large NC models a Loss of separation event is 1.4 and 1.7 times more likely to happen. This can be explained by the previously mentioned movement pattern, where multiple aircrafts fly towards the vertiport centre simultaneously.

At the same time, the picture with hidden priorities is different: here the NC agents tend to put a bigger emphasis on safety assurance and keep larger distances among themselves, compared to their communicative counterparts. This effect becomes more apparent with the increase in the amount of aircrafts, as indicated by the significant drop in the incident and accident rates in the Hidden Large setup. Such behaviour comes at a cost of increased waiting times and underutilisation of vertiport capacity, as described in the section 4.1.2.

Setup	Incidents,%	Accidents, %
Hidden	0.3397	0.0021
Hidden NC	0.3106	0.0035
Equal	0.3876	0.0014
Equal NC	0.5539	0.0113
Hidden Large	0.5137	0.0045
Hidden Large NC	0.1275	0.0008
Equal Large	0.9670	0.0080
Equal Large NC	1.6469	0.0114

Table 4.2: Incidents and accidents rates

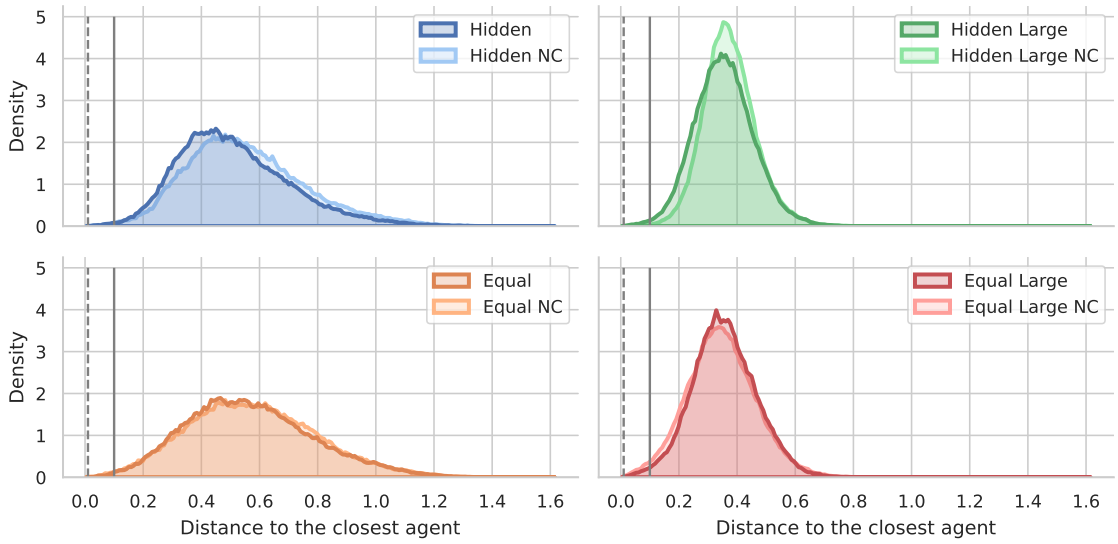

Figure 4.4: Distributions of distances to the closest agent across different setups. Solid black line indicates minimum separation distance, dashed line indicates collision distance. Absence of communication in Equal setups entails more frequent incidents.

4.2 COMMUNICATION

As it was seen in the section 4.1, introducing communication mechanisms into the multi-agent system opens up new possibilities for different navigation strategies, for example removing the incentive for each individual agent to approach the goal, when some other is already on it's way. In this regard, it is beneficial to develop an understanding of how exactly the agents use their communication capabilities, in what context the communication takes place, and what might be the factors that influence the decision to communicate.

4.2.1 Communication action description

Before investigating the agent's use of communication, it is necessary to scrutinise the mechanism behind it in the IC3Net model. Section 2.4.4 outlines the technical concepts of the original approach, and section 3.3 explains the changes, that were made to it for the purpose of this experiment. In IC3Net, an output of a gating function g_j^t is referred to as "communication action" (comm-action) of an agent j at time step t. In the current model it is calculated according to the equation 3.1. The message that is sent out is then $C \cdot (h_j^t \odot g_j^t)$, where h_j^t is a hidden state of an agent j and C is some weight matrix.

Since such a comm-action is continuous, it can be intuitively understood as a measure to regulate the strength of the signal, broadcasted by an agent. An additional advantage of this manifests itself in the way the information is integrated into the system. Given that all incoming messages are condensed into a mean (equation 2.21), continuity of the comm-action allows to flexibly adjust the weight of the message in the resulting sum, making it possible for the agent to emphasise the importance of a sent signal. From this point of view, it is logical to evaluate comm-actions not only by their absolute value but also in terms of the cumulative density function of the distribution of

all comm-actions a given model learnt, i.e. consider in which quantile a comm-action lies. On top of this, it might not be totally valid to compare the absolute values of different models, since they might develop differing communication protocols. Therefore it is more sensible to observe the changes in comm-action depending on the influencing factors.

4.2.2 Communication action and trajectory

The aircraft's position in the airspace can be important information that might need to be shared with the other traffic participants. Taking into account the radial symmetry of the mean movement angles, concluded from the figure 4.1, the location itself would not have a direct influence on communication. Instead it is sufficient to consider how the distance and relative bearing towards the goal impact comm-action. Figure 4.5 illustrates this relation.

The plots are presented in polar coordinates, where the angle indicates a relative bearing of an agent towards the vertiport centre, and the radius reflects the distance to it, measured relatively to the terminal area radius. Here the border of a vertiport is located at the distance equal to 0.2. The data itself represents a cumulative density function (CDF) of comm-actions distribution, mapped onto distance and bearing. It was computed as follows. First, all comm-actions within a given setup were transformed by a rank function (applied an inverse of the CDF of their distribution, i.e. extracted a quantile of a given comm-action). Second, the space of distance-bearing pairs is split into discrete bins, and a mean value of all comm-action quantiles taken within a bin is calculated. This visualisation does not consider the sample size for a distance-bearing pair and only indicates a mean value. For the purpose of better representation the bins with a sample size of less than 2 are omitted (i.e. distance-bearing pairs that happened only twice over the whole dataset). A blank space in the graph indicates that no samples with a given distance and bearing were recorded during the simulation process.

From this figure it is possible to establish a preferred movement direction within a setup. In all of them the data is present only for some set of relative bearings. For example, in Equal Large it is between 315° to 180° clockwise, which tells that the aircrafts mostly travel in a clockwise direction around the goal. In the case of Hidden setup it is mostly counter-clockwise, since in the bins corresponding to small distances to the centre (close to 0.2) the data is present only for the bearing of 225° to 45° clockwise. This reasoning is supported by the figure 4.1.

A distinctive pattern in the open goal across all setups is apparent: comm-action becomes the strongest at close distances for the values of relative bearing close to 0. This corresponds to the aircraft directly heading towards the vertiport centre. It can be viewed as them communicating their intent to enter the goal. More than that, the agents are not only able to inform the others about their decision to move towards the goal, but they also adapt their own trajectories according to other agent's intent. This can be concluded from the decrease in the amount of closed entrances compared to NC models, presented in the table 4.1. This finding reinforces the argument made in the section 4.1.2 with regard to conflict resolution capabilities of communicative models.

A reverse observation can be made about the regions of weak communication signals. Regardless of the goal status, they are concentrated around the bearing of 90° (270° for Hidden) at varying distances depending on the size of the setup. The interpretation is straightforward: with such relative bearing towards the vertiport centre, the distance to it remains constant, i.e. the aircrafts

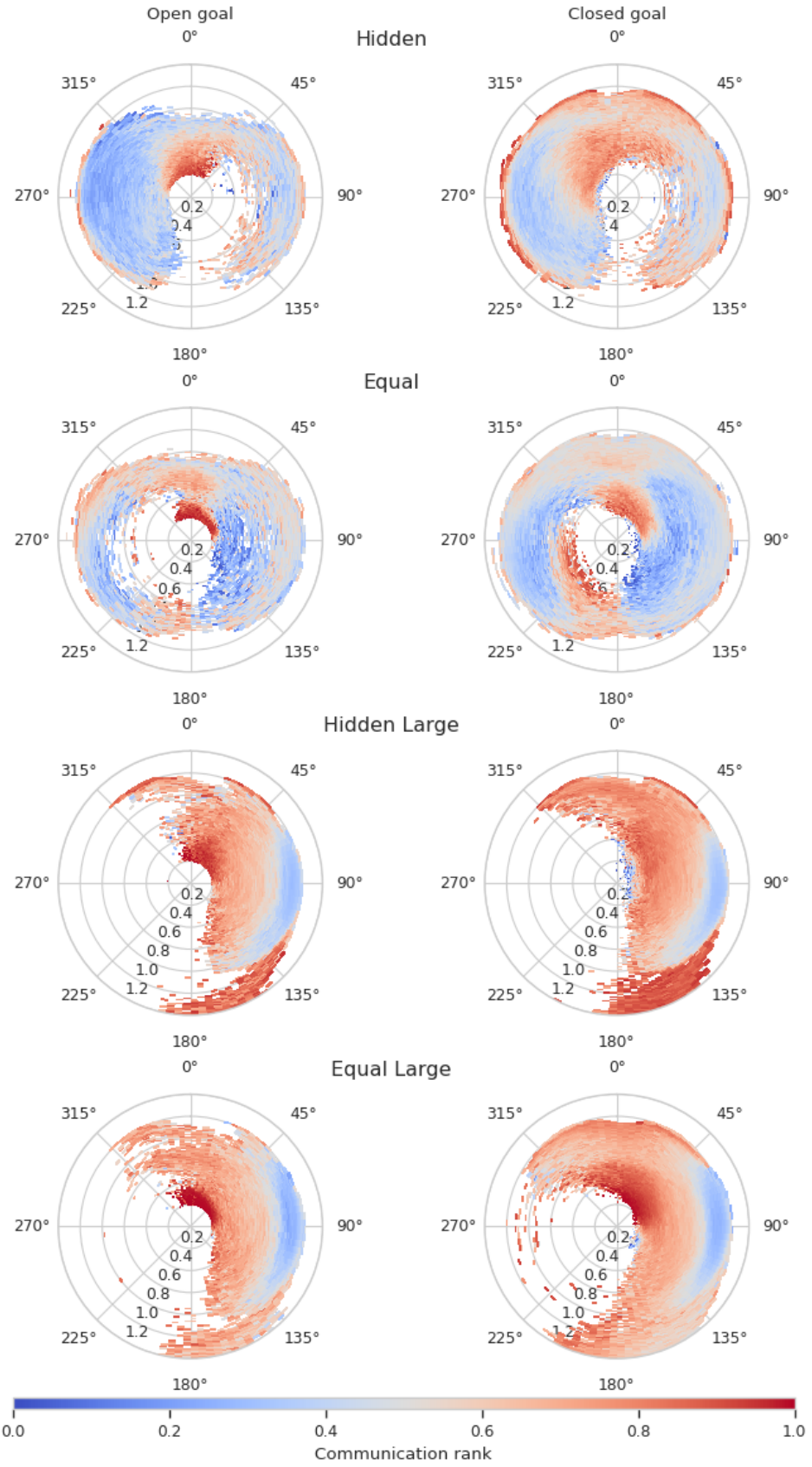

Figure 4.5: Communication across different setups by location and bearing. Figure shows at what distance and relative bearing the agents preferred to take comm-actions, that are stronger or weaker than median. **Left:** Open goal. **Right:** Closed goal.

are not performing any manoeuvres and are following the path along the edge of the terminal area. This corresponds to them not indicating intent to enter the goal. Furthermore, the size of this region on the plot shrinks with the increase in the amount of aircrafts. It is connected to the self-organisation of agents in the Large setups, described in the section 4.1.1, where the vast majority of them travel close to the border. Therefore any movement towards the centre comes with an above-average comm-action.

Typically communication strength does not change depending on the goal status, with the exception of strongest communication being associated with an open goal. At the same time, the strongest comm-actions are also present in the Equal Large setup even for the closed goal. As the table 4.1 suggests, the utilisation of the vertiport in this setup is one of the highest, with 79% of the time goal being closed. This implies that the aircrafts advance towards the vertiport in anticipation of it becoming available, thus preemptively communicating an intent to enter.

Figure 4.5 concerns communication distribution in the context of the whole dataset. It shows a clear connection between close proximity to the goal and an increase in comm-action. To see if the agents actually implement into reality their communicated intent, an additional analysis is needed. To this end, figure 4.6 shows the probability of an agent entering the goal within the next 30 time steps, given a quantile of their comm-action. The distribution is split into 20 consecutive quantile ranges, i.e. each bin corresponds to 5% of the distribution. From this plot it becomes evident that a comm-action, that falls into the top 5%, is associated with the largest chance of this particular agent being next to enter the goal. Also, in Equal setups the agents complete their intent more frequently, compared to the Hidden setups, with the probabilities of entrance being equal to 0.8 and 0.7 in the former cases, and 0.6 and 0.45 in the latter. In addition, the fact that the probability of entrance increases substantially only for the top 10% of all comm-actions in large setups is an effect of more agents existing in the system. Since a larger amount of aircrafrs is producing the messages, and only one of them can enter a vertiport, large messages that communicate intent get pushed further towards the tail of a distribution.

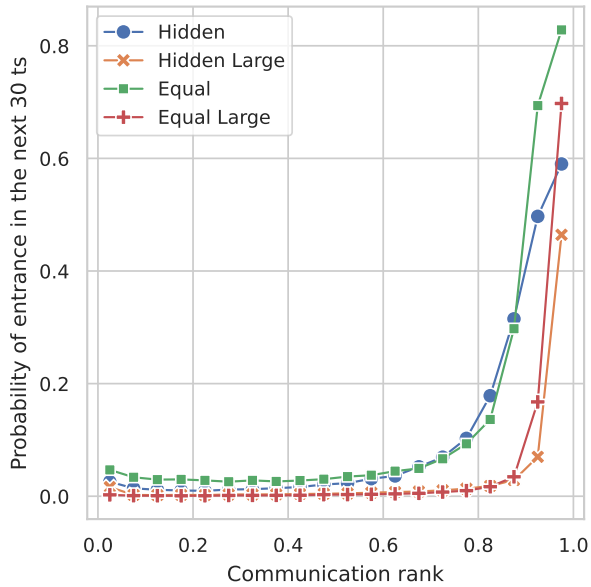

Figure 4.6: Probability of entrance given communication action quantile. If the agent sends the signal in the top 10%, its chances to enter increase substantially.

In general, it has been seen that the models in Hidden and in Equal setups share the same communication patterns. The fact, that in the former they were incentivised to communicate some "ground truth", and still converged to the same protocol without any external factors in the latter, suggests that the agents are creating some form of momentary hierarchy. This allows them to perform a vertiport arrival scheduling in a structured manner even without explicit guidance.

4.2.3 Communication action and priority

As was established in the section 4.2.2, inter-agent communication is strongly connected with their movement towards the goal. In Hidden priority setups, an aircraft with the largest priority number is also supposed to arrive at the vertiport, and the models are able to learn this rule, as shown by figure 4.3. In light of such a connection, it is reasonable to study the influence a priority has on a communication policy. Figure 4.7 illustrates a comparison between comm-actions' distributions of the agents with maximum priority and the rest of the agents. In this graph maximum priority is understood in the sense of rank, as described in the section 4.1.2. A clear distinction between two distributions within a setup can be made, as comm-actions of the maximum priority agents have a much larger spread. This hints that these agents adjust their communication flexibly, depending on the context. Another remark can be made with regard to the distributions: the median comm-actions in regular and large setups are not identical, being close to 0.55 and 0.39 respectively, which potentially implies different communication strategies. The fact, that a noticeable disparity in the absolute values is present, validates the proposition to compare different setups in terms of quantiles of corresponding distributions.

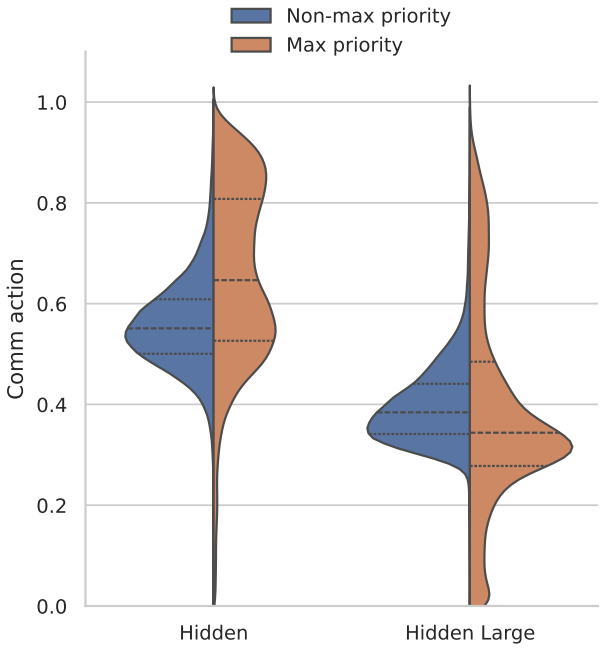

Figure 4.7: Communication action by maximum priority. Communication is more spread out in agents with the largest priority.

In addition to the priority rank, it is also possible to draw a connection between communication and the priority number itself. In figure 4.8 such a relation is presented. Each box reflects the distribution of comm-actions that were taken by an agent with a given priority number, depending on the status of its future goal entrance. First, it can be recognised that the models indeed develop different communication strategies, since for non-entering agents the comm-action grows with priority in the regular setup, and falls in the large one. The reasons behind this phenomenon require further in-depth study. Second, developing an argument, made with regard to the maximum priority agents, it is easy to see the reason behind an increased spread of their comm-actions. In the large setup, the agents with high priority numbers are able to recognise their importance in the environment, and not only communicate their intent to enter the goal, but also explicitly signal if they are not planning to do so. This plot clearly shows three modes of communication: strong comm-action if the agent travels towards the goal, weak comm-action if the agent considers itself relevant but does not travel towards the goal, and medium comm-action otherwise.

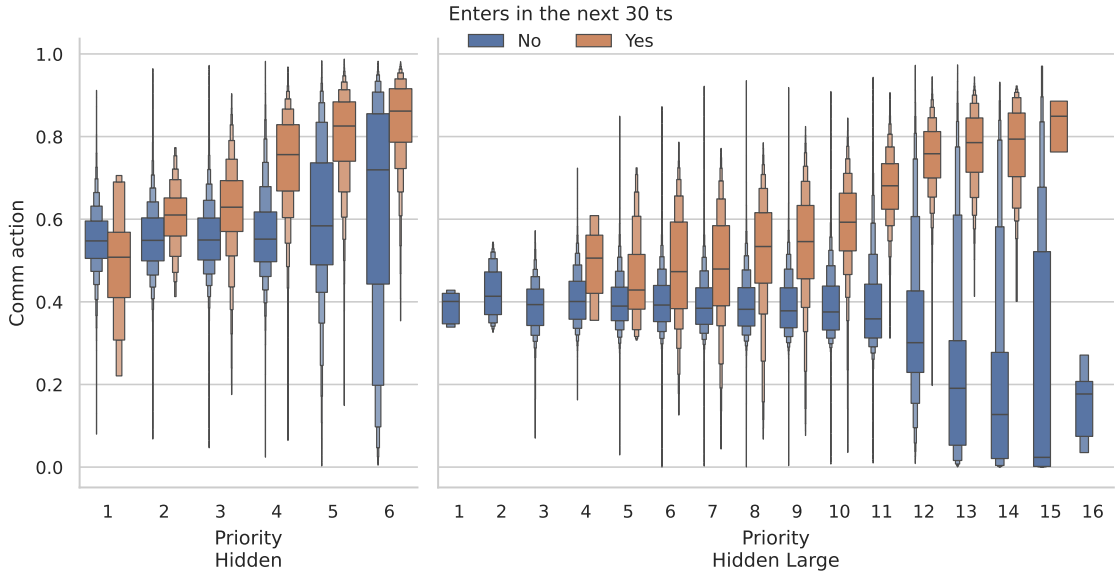

Figure 4.8: Communication action by priority number and future entrance. In Large setup the agents recognise their importance and explicitly signal whether they plan to enter or not.

5 CONCLUSION

Communication in MARL is a flexible and practical approach to coordinating the behaviour of multiple agents interacting within one system. The agents can adapt to the surroundings and send diverse kinds of messages, depending on the situation. Generally, the design of the environment also determines the structure of inter-agent information exchange that takes place, and therefore the impact of communication needs to be interpreted using some domain knowledge. In this case, its effect was considered in the context of Urban Air Mobility, in particular for the vertiport terminal area control. Existing solutions to this problem involve some central managing system, that coordinates the arrival order of the aircrafts. An approach, that was implemented, makes use of a decentralised multi-agent reinforcement learning model, in which the agents learnt to exchange information and collaborate to achieve a common goal of optimising the vertiport utilisation.

Given that the nature of communication is domain-dependent, in order to establish its content and effect on the environment, two distinct experimental setups were designed. One setup, called "Hidden priority", maintains some notion of arrival order and assumes it to be some form of "ground truth" that would be necessary to communicate in order to achieve the goal. The other, called "Equal priority", abolishes any explicit hierarchy among the agents and allows them to develop arbitrary communication protocols. Comparing the performances of communicative and non-communicative models within both of these setups allows to understand the influence of information sharing on coordination.

Simulation results revealed that the MARL-based system is capable of self-organising in the terminal area of a vertiport in a way, that ensures safety of travel and proper capacity utilisation. In particular, it was observed that without any specific instructions the agents learnt to navigate the airspace similarly to the arrangement proposed by [Bertram and Wei(2020)], in which predetermined movement trajectories comprised a set of concentric rings with different radii. Furthermore, communicating models were shown to have better coordination capabilities over non-communicative, resulting in a substantially smaller amount of situations, in which two aircrafts attempted to enter a vertiport at the same time. This finding suggests that some form of message exchange should be considered for a decentralised aircraft control system, that does not rely on external arrival management.

By contrasting the communication protocols developed by the models in two experimental setups, it was possible to conclude that a common structure of communication is learnt. In both settings the aircrafts signaled their intent to approach the vertiport. Furthermore, with the increase in strength of such signal from one agent, the rest of them became cooperative and more likely to let that particular agent enter the goal. In addition, the agents with explicit hierarchy were seen to recognise their importance in the environment, which entailed a strong signal communicating either an intent to enter or an absence thereof.

An outlined experimental framework was designed with the aim of understanding the essence behind communication, happening in a decentralised system. It does not consider how such systems compare to their centralised counterparts in terms of efficiency. Organising communication pipelines requires additional resources, therefore an adoption of this approach needs to have a competitive justification.

Potential future research can be directed at quantifying the benefits of decentralised communication-based MARL methods over structured and strictly regulated vertiport scheduling procedures. In addition, for the purpose of designing and deploying such systems in realistic conditions, questions of robustness and scalability of communication need to be investigated. In terms of methodological aspects, it is possible to consider how other priority management schemes, like First-In-First-Out, can affect the usefulness of inter-agent communication. Furthermore, it would be worthwhile to adapt the described approach to accommodate the variable amount of vehicles in the system.

BIBLIOGRAPHY

[Apaza et al.(2023)] Rafael D. Apaza, Hongxiang Li, Ruixuan Han, and Eric J. Knoblock. 2023. Multi-agent Deep Reinforcement Learning for Spectrum and Air Traffic Management in UAM with Resource Constraints. *2023 IEEE/AIAA 42nd Digital Avionics Systems Conference (DASC)* (2023), 1–7. https://api.semanticscholar.org/CorpusID:265700924

[Bauranov and Rakas(2021)] Aleksandar Bauranov and Jasenka Rakas. 2021. Designing airspace for urban air mobility: A review of concepts and approaches. *Progress in Aerospace Sciences* 125 (2021), 100726. https://api.semanticscholar.org/CorpusID:236253535

[Bertram and Wei(2020)] Josh Bertram and Peng Wei. 2020. An Efficient Algorithm for Self-Organized Terminal Arrival in Urban Air Mobility. https://api.semanticscholar.org/CorpusID:214104864

[Cohen et al.(2021)] Adam P. Cohen, Susan A. Shaheen, and Emily Farrar. 2021. Urban Air Mobility: History, Ecosystem, Market Potential, and Challenges. *IEEE Transactions on Intelligent Transportation Systems* 22 (2021), 6074–6087. https://api.semanticscholar.org/CorpusID:236226750

[Das et al.(2020)] Abhishek Das, Théophile Gervet, Joshua Romoff, Dhruv Batra, Devi Parikh, Michael Rabbat, and Joelle Pineau. 2020. TarMAC: Targeted Multi-Agent Communication. arXiv:1810.11187 [cs.LG]

[de Witt et al.(2020)] Christian Schroeder de Witt, Tarun Gupta, Denys Makoviichuk, Viktor Makoviychuk, Philip H. S. Torr, Mingfei Sun, and Shimon Whiteson. 2020. Is Independent Learning All You Need in the StarCraft Multi-Agent Challenge? arXiv:2011.09533 [cs.AI]

[EASA(2022)] EASA. 2022. *EASA issues world's first design specifications for vertiports*. https://www.easa.europa.eu/en/newsroom-and-events/press-releases/easa-issues-worlds-first-design-specifications-vertiports

[Foerster(2018)] J Foerster. 2018. *Deep multi-agent reinforcement learning*. Ph. D. Dissertation. University of Oxford.

[Foerster et al.(2017)] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. 2017. Counterfactual Multi-Agent Policy Gradients. arXiv:1705.08926 [cs.AI]

[Foerster et al.(2016)] Jakob N. Foerster, Yannis M. Assael, Nando de Freitas, and Shimon Whiteson. 2016. Learning to Communicate with Deep Multi-Agent Reinforcement Learning. arXiv:1605.06676 [cs.AI]

[Guerreiro et al.(2020)] Nelson M. Guerreiro, George Hagen, Jeffrey Maddalon, and Ricky W. Butler. 2020. Capacity and Throughput of Urban Air Mobility Vertiports with a First-Come, First-Served Vertiport Scheduling Algorithm. *AIAA AVIATION 2020 FORUM* (2020). https://api.semanticscholar.org/CorpusID:225720837

[Gupta et al.(2022)] Nikunj Gupta, G Srinivasaraghavan, Swarup Kumar Mohalik, Nishant Kumar, and Matthew E. Taylor. 2022. HAMMER: Multi-Level Coordination of Reinforcement Learning Agents via Learned Messaging. arXiv:2102.00824 [cs.MA]

[Hoekstra and Ellerbroek(2016)] Jacco M. Hoekstra and Joost Ellerbroek. 2016. BlueSky ATC Simulator Project: An Open Data and Open Source Approach. https://api.semanticscholar.org/CorpusID:42745656

[Huang et al.(2023)] Cheng Huang, I. Petrunin, and Antonios Tsourdos. 2023. Strategic Conflict Management using Recurrent Multi-agent Reinforcement Learning for Urban Air Mobility Operations Considering Uncertainties. *Journal of Intelligent & Robotic Systems* 107 (2023), 1–21. https://api.semanticscholar.org/CorpusID:256268581

[Inala et al.(2021)] Jeevana Priya Inala, Yichen Yang, James Paulos, Yewen Pu, Osbert Bastani, Vijay Kumar, Martin Rinard, and Armando Solar-Lezama. 2021. Neurosymbolic Transformers for Multi-Agent Communication. arXiv:2101.03238 [cs.MA]

[Jiang et al.(2020)] Jiechuan Jiang, Chen Dun, Tiejun Huang, and Zongqing Lu. 2020. Graph Convolutional Reinforcement Learning. arXiv:1810.09202 [cs.LG]

[Jiang and Lu(2018)] Jiechuan Jiang and Zongqing Lu. 2018. Learning Attentional Communication for Multi-Agent Cooperation. arXiv:1805.07733 [cs.AI]

[Kleinbekman et al.(2018)] Imke C Kleinbekman, Mihaela A Mitici, and Peng Wei. 2018. eVTOL arrival sequencing and scheduling for on-demand urban air mobility. In *2018 IEEE/AIAA 37th Digital Avionics Systems Conference (DASC)*. IEEE, 1–7.

[KrisshnaKumar et al.(2023)] Prajit KrisshnaKumar, Jhoel Witter, Steve Paul, Hanvit Cho, Karthik Dantu, and Souma Chowdhury. 2023. Fast Decision Support for Air Traffic Management at Urban Air Mobility Vertiports using Graph Learning. arXiv:2308.09075 [cs.MA]

[Lowe et al.(2020)] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. 2020. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. arXiv:1706.02275 [cs.LG]

[Marwah et al.(2023)] Nirmal Marwah, Vivek Kumar Singh, Gautam Siddharth Kashyap, and Samar Wazir. 2023. An analysis of the robustness of UAV agriculture field coverage using multi-agent reinforcement learning. *International Journal of Information Technology* 15 (2023), 2317–2327. https://api.semanticscholar.org/CorpusID:258561285

[Oliehoek et al.(2008)] Frans A. Oliehoek, Matthijs T. J. Spaan, and Nikos A. Vlassis. 2008. Optimal and Approximate Q-value Functions for Decentralized POMDPs. *J. Artif. Intell. Res.* 32 (2008), 289–353. https://api.semanticscholar.org/CorpusID:1627313

[Papoudakis et al.(2021)] Georgios Papoudakis, Filippos Christianos, Lukas Schäfer, and Stefano V. Albrecht. 2021. Benchmarking Multi-Agent Deep Reinforcement Learning Algorithms in Cooperative Tasks. arXiv:2006.07869 [cs.LG]

[Park et al.(2023)] Chanyoung Park, Gyu Seon Kim, Soohyun Park, Soyi Jung, and Joongheon Kim. 2023. Multi-Agent Reinforcement Learning for Cooperative Air Transportation Services in City-Wide Autonomous Urban Air Mobility. arXiv:2306.04137 [cs.MA]

[Pradeep and Wei(2018)] Priyank Pradeep and Peng Wei. 2018. Heuristic Approach for Arrival Sequencing and Scheduling for eVTOL Aircraft in On-Demand Urban Air Mobility. *2018 IEEE/AIAA 37th Digital Avionics Systems Conference (DASC)* (2018), 1–7. https://api.semanticscholar.org/CorpusID:54464840

[Schulman et al.(2017)] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. arXiv:1707.06347 [cs.LG]

[Shapley(1953)] Lloyd S Shapley. 1953. Stochastic games. *Proceedings of the national academy of sciences* 39, 10 (1953), 1095–1100.

[Silva et al.(2018)] Christopher J. Silva, Wayne Johnson, Eduardo Solis, Michael D. Patterson, and Kevin R. Antcliff. 2018. VTOL Urban Air Mobility Concept Vehicles for Technology Development. *2018 Aviation Technology, Integration, and Operations Conference* (2018). https://api.semanticscholar.org/CorpusID:116532349

[Singh et al.(2018)] Amanpreet Singh, Tushar Jain, and Sainbayar Sukhbaatar. 2018. Learning when to Communicate at Scale in Multiagent Cooperative and Competitive Tasks. arXiv:1812.09755 [cs.LG]

[Stone and Veloso(1998)] Peter Stone and Manuela M. Veloso. 1998. Towards collaborative and adversarial learning: a case study in robotic soccer. *Int. J. Hum. Comput. Stud.* 48 (1998), 83–104. https://api.semanticscholar.org/CorpusID:171919

[Sukhbaatar et al.(2016)] Sainbayar Sukhbaatar, Arthur Szlam, and Rob Fergus. 2016. Learning Multiagent Communication with Backpropagation. arXiv:1605.07736 [cs.LG]

[Sutton and Barto(2018)] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.

[Towers et al.(2023)] Mark Towers, Jordan K. Terry, Ariel Kwiatkowski, John U. Balis, Gianluca de Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Arjun KG, Markus Krimmel, Rodrigo Perez-Vicente, Andrea Pierré, Sander Schulhoff, Jun Jet Tai, Andrew Tan Jin Shen, and Omar G. Younis. 2023. Gymnasium. https://doi.org/10.5281/zenodo.8127026

[Vinyals et al.(2019)] Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, L. Sifre, Trevor Cai, John P. Agapiou, Max Jaderberg, Alexander Sasha Vezhnevets, Rémi Leblond, Tobias Pohlen, Valentin Dalibard, David Budden, Yury Sulsky, James Molloy, Tom Le Paine, Caglar Gulcehre, Ziyun Wang, Tobias Pfaff, Yuhuai Wu, Roman Ring, Dani Yogatama,

Dario Wünsch, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy P. Lillicrap, Koray Kavukcuoglu, Demis Hassabis, Chris Apps, and David Silver. 2019. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature* 575 (2019), 350 – 354. https://api.semanticscholar.org/CorpusID:204972004

[Waltz and Paulig(2022)] Martin Waltz and Niklas Paulig. 2022. RL Dresden Algorithm Suite. https://github.com/MarWaltz/TUD_RL.

[Watkins and Dayan(1992)] Christopher Watkins and Peter Dayan. 1992. Q-learning. *Machine Learning* 8 (1992), 279–292. https://api.semanticscholar.org/CorpusID:208910339

[Williams(1992)] Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning* 8 (1992), 229–256. https://api.semanticscholar.org/CorpusID:2332513

[Xia et al.(2022)] Zhaoyue Xia, Jun Du, Jingjing Wang, Chunxiao Jiang, Yong Ren, Gang Li, and Zhu Han. 2022. Multi-Agent Reinforcement Learning Aided Intelligent UAV Swarm for Target Tracking. *IEEE Transactions on Vehicular Technology* 71 (2022), 931–945. https://api.semanticscholar.org/CorpusID:244509143

[Yang and Wei(2020)] Xuxi Yang and Peng Wei. 2020. Scalable Multi-Agent Computational Guidance with Separation Assurance for Autonomous Urban Air Mobility. *Journal of Guidance Control and Dynamics* 43 (2020), 1473–1486. https://api.semanticscholar.org/CorpusID:219435229

[Yu et al.(2022)] Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. 2022. The Surprising Effectiveness of PPO in Cooperative, Multi-Agent Games. arXiv:2103.01955 [cs.LG]

[Yun et al.(2021)] Won Joon Yun, Byungju Lim, Soyi Jung, Young-Chai Ko, Jihong Park, Joongheon Kim, and Mehdi Bennis. 2021. Attention-based Reinforcement Learning for Real-Time UAV Semantic Communication. arXiv:2105.10716 [cs.MA]

[Zhu et al.(2022)] Changxi Zhu, Mehdi M. Dastani, and Shihan Wang. 2022. A Survey of Multi-Agent Reinforcement Learning with Communication. *ArXiv* abs/2203.08975 (2022). https://api.semanticscholar.org/CorpusID:247518595