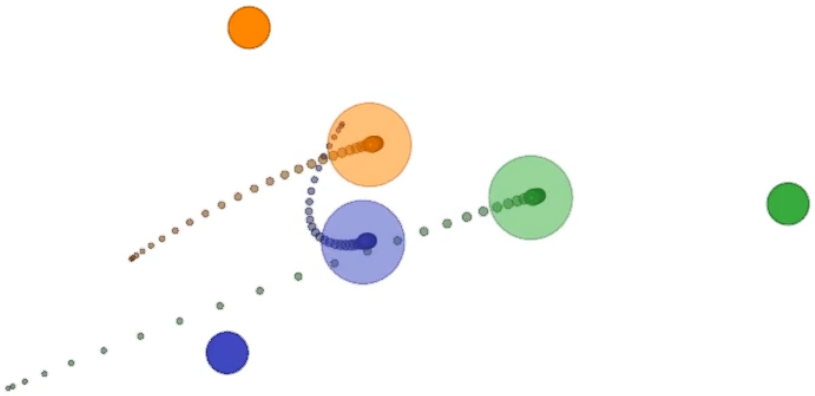


Understanding Linear Quadratic Drone Games through Simulation

Linear Quadratic Games as a Baseline for Evaluating Multi-Agent Reinforcement Learning Algorithms and Simulation as a Tool for Understanding and Innovation

Masters Thesis in Learning and Leadership

Filip Berglund
Lukas Wenåker

DEPARTMENT OF COMMUNICATION AND LEARNING IN SCIENCE

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2026
www.chalmers.se

Masters thesis 2026

Understanding Linear Quadratic Drone Games through Simulation

Linear Quadratic Games as a Baseline for Evaluating Multi-Agent Reinforcement Learning Algorithms and Simulation as a Tool for Understanding and Innovation

Filip Berglund
Lukas Wenåker

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Communication and Learning in Science
Chalmers University of Technology
Gothenburg, Sweden 2026

Understanding Linear Quadratic Drone Games through Simulation
Linear Quadratic Games as a Baseline for Evaluating Multi-Agent Reinforcement Learning Algorithms and Simulation as a Tool for Understanding and Innovation

Supervisor: Mika Persson, Department of Mathematical Sciences & Saab AB
Examiner: Philip Gerlee, Department of Mathematical Sciences

Masters Thesis 2026
Department of Communication and Learning in Science
Chalmers University of Technology
SE-412 96 Gothenburg
Sweden
Telephone +46 31 772 1000

Cover: Trajectory for Nash equilibrium solution to a three-player game.
Typeset in LaTeX
Gothenburg, Sweden 2026

Understanding Linear Quadratic Drone Games through Simulation
Linear Quadratic Games as a Baseline for Evaluating Multi-Agent Reinforcement Learning Algorithms and Simulation as a Tool for Understanding and Innovation
Filip Berglund, Lukas Wenåker
Department of Communication and Learning in Science
Chalmers University of Technology

Abstract

A relevant method of generating control policies for autonomous drones is through multi-agent reinforcement learning (MARL) algorithms. Novel MARL algorithms do not always have theoretical guarantees of convergence which motivates the need for robust and reliable baselines to benchmark these algorithms against. This thesis investigates the efficacy of derived Nash equilibrium (NE) solutions to the family of linear quadratic (LQ) games as one such possible baseline. The first part of the thesis derives the Nash equilibrium solutions for LQ games, which serve as the baseline policies. Based on these results, two experimental scenarios are designed to benchmark MARL algorithms against the baseline. The experimental results indicate that the MARL policies perform on par with the baseline in the two-player scenario, while outperforming the baseline in the cooperative five-player scenario. The second part of the thesis explores to what extent computer simulation can be used as an effective knowledge sharing method at an engineering company. An exploratory pilot study was conducted comparing two learning sessions, one session employing an online simulation tool developed for this purpose and the other being a traditional lecture. Responses collected after each learning session indicate that the visual and interactive elements of the simulation tool were conducive to generating engagement and curiosity among participants. Furthermore, providing necessary context and examples of applicability were deemed important aspects when sharing information about a novel topic among engineers.

Keywords: MARL, Linear Quadratic Games, Nash Equilibrium, Simulation-Based Learning

Acknowledgements

We want to thank Maria Stegberg, Christian Ehrenström Roos and David Olsson as well as all other members of XIL for helping us get set up at Saab and making our time there enjoyable. We also wish to thank Adam Andersson and Tom Adawi for their valuable input. Lastly we wish to sincerely thank our supervisor Mika Persson for going above and beyond in his continued support.

Filip Berglund, Lukas Wenåker, Gothenburg, May 2026

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

AAV	Aggregated Agent Value
AAVC	Aggregated Agent Value Comparison
DARE	Discrete Algebraic Riccati Equation
DPE	Dynamic Programming Equation
IAVC	Inter-agent Value Comparison
LS1	Learning session 1
LS2	Learning session 2
LQ	Linear Quadratic
MAPPO	Multi-Agent Proximal Policy Optimization
MARL	Multi-Agent Reinforcement Learning
MLP	Multi-layer Perceptron
NE	Nash Equilibrium
SECI	Socialization, Externalization, Combination and Internalization
UAV	Unmanned Aerial Vehicle
VMAS	Vectorized Multi-Agent Simulator

Nomenclature

Below is the nomenclature of variables, sets, operators and symbols that have been used throughout this thesis.

Variables

A_{ij}	The element at row i and column j in matrix A.
N	The number of players, $N \in \mathbb{N}$.
n	The number of state variables, $n \in \mathbb{N}$.
m	The number of action variables, $m \in \mathbb{N}$.
T	The maximum time step $t \in \mathbb{N}$ for one game instance.

Sets

$\mathbb{R}$	The set of all real numbers.
$\mathbb{R}^n$	The set of all pairs of n real numbers.
I	The set of players, $I = \{1, \ldots, N\}$.
I_{-i}	The set of players except i, $I_{-i} = I \backslash \{i\}$.
I_{p_x}	The set of state indices for the x-positions.
I_{p_y}	The set of state indices for the y-positions.
I_p	The set of state indices for the x and y-positions.
$I_{i,p} := \{k_{i,p_x}, k_{i,p_y}\}$	The set of x- and y-position state indices for player i.
$I_{i,g} := \{k_{i,g_x}, k_{i,g_y}\}$	The set of x- and y-goal position state indices for player i.

Operators

$\mathrm{Tr}(A)$	The trace of a matrix A, $\mathrm{Tr}(A) = \sum_{i=1}^{n} a_{ii}$.

$\|\|A\|\|$	The Frobenius norm of matrix A, $\|\|A\|\| = \left[\sum_{i=1}^{n}\sum_{j=1}^{n}(a_{ij})^2\right]^{1/2}$.
$\sigma(A)$	The set of eigenvalues of a matrix A, $\sigma(A) = \{\lambda \mid \det(A - \lambda I) = 0\}$.
$\mathbb{E}_w[X]$	The expected value of a stochastic variable X with respect to w.

Symbols

$A \prec 0$	The matrix A is negative definite, meaning that $\lambda < 0$ for all eigenvalues $\lambda \in \sigma(A)$.
$A \preceq 0$	The matrix A is negative semi definite, meaning that $\lambda \leq 0$ for all eigenvalues $\lambda \in \sigma(A)$.
$\mathcal{N}(\mu, \sigma)$	The gaussian distribution with mean μ and standard deviation σ.

Contents

1

Introduction

In this chapter, the background to the thesis is presented as well as aims and research questions. We also define the delimitations and some ethical considerations. Since the thesis covers two distinct parts, a reading guide is also provided to aid in navigating the thesis.

1.1 Background

In the modern landscape, it is becoming increasingly important for nations to maintain citizen protection and disaster response capabilities. In order to fulfill this role, we need to develop the technology for drones and understanding of managing and controlling drone swarms. Drones are unmanned aerial vehicles (UAV) that can be used to provide assistance during disasters, such as establishing network communication or transporting medical supplies to inaccessible areas [1].

This master's thesis focuses on the autonomy of drones and drone swarms. Autonomy refers to the capability of drones to perceive their environment and make decisions independently, without human intervention. This can be modeled as an optimization problem where drones take the sequence of actions that maximizes a defined objective function, prescribed by some reward function which specifies how good an action is in a certain state of the environment. For example, drones should not accelerate unnecessarily much, in order to conserve battery life. It is also crucial to control drone swarms in a decentralized manner, where each drone possesses its own onboard perception and decision-making capabilities. This increases the swarm's scalability and resilience to failures, as there is no single point of failure, unlike in a centralized architecture [2].

One way to obtain decentralized cooperative autonomous UAVs is through multi-agent reinforcement learning (MARL) methods where a so called *decision policy* is trained for each drone through trial and error interaction with the environment guided by a reinforcement learning algorithm. However, these algorithms tend to pose certain challenges such as non-stationarity, scalability to a larger number of agents and establishing when the agents are sufficiently trained [3]. This motivates the need to develop robust baselines with which to assess the feasibility and performance of novel MARL-algorithms. One such possible baseline is Nash equilibrium (NE) policies to linear quadratic (LQ) games, for which optimal solutions can be

derived.

LQ games are also of interest as benchmark problems for studying the convergence properties of policy optimization methods in MARL. For example, [4] showed that several policy-gradient methods converge to a global Nash equilibrium in zero-sum LQ games under suitable assumptions. These results are closely related to actor–critic methods such as the multi-agent proximal policy optimization algorithm (MAPPO) studied in this work. However, such guarantees generally do not extend to general-sum LQ games [5]. Consequently, the Nash equilibrium provide a useful reference point for evaluating MARL methods in settings where theoretical convergence guarantees are limited.

This thesis also explore effective knowledge sharing methods at an engineering company. Companies that employ effective knowledge sharing for tacit and explicit knowledge tend to have better financial outcomes, as shown by [6]. One proposed mechanism for this is that effective knowledge sharing leads to greater innovation speed [6]. Innovation can be described as the process by which information is acquired, assimilated and shared to create new knowledge that is embodied in products and services [7]. One way to share knowledge is through planned knowledge sharing sessions such as technical briefings or formal lectures. One general challenge of knowledge sharing, especially in engineering, is conveying complex information in an understandable way [8]. It is therefore of interest to explore ways in which technical information can be shared such that it is conceptually understandable, as well as engaging and conducive to innovation. An alternative to lecture-style knowledge sharing is through the use of interactive simulation tools. Learning sessions incorporating interactive simulation tools have been shown to increase conceptual understanding of topics in physics, for high school students [9]. This motivates studying its effectiveness in the context knowledge sharing for engineers.

1.2 Aims and Research Questions

The aim of this thesis is to derive solutions, in the form of Nash equilibrium policies to linear quadratic games and evaluate how they compare to policies trained using multi-agent reinforcement learning. This comparison is done in the context of drone coordination tasks. In conjunction with this, we also wish to explore whether simulation-based learning is an effective form of knowledge sharing in an engineering context. Specifically, we set out to explore its ability to generate engagement and conceptual understanding of advanced topics in formal to semi-formal knowledge sharing sessions, such as lectures or show and tells. With these aims in mind, we pose the following research questions:

1. How are Nash equilibrium solutions to linear quadratic games derived?
2. To what extent are NE-policies to linear quadratic games suitable for use as a baseline to evaluate MARL-algorithms?
3. To what extent are computer simulations an effective knowledge sharing method conducive to learning and engagement at an engineering company?

4. What factors do engineers perceive as important in a formal to semi-formal knowledge sharing session?

1.3 Scope and Delimitations

The scope of this thesis is restricted to the study of LQ-games with full observability. While assuming partial observability allows for the modeling of more realistic scenarios, it introduces complexity beyond the scope of this thesis. Furthermore, the evaluation is limited to comparison with a single MARL-algorithm.

As for answering research questions 3 and 4, an exploratory pilot study was designed to gather the necessary data. This study consisted of two different learning sessions at Saab AB, with the participants answering a questionnaire before and after the learning sessions. To properly study the effects of learning, a follow-up questionnaire to test the retention of the content would yield better results. This follow-up survey was deemed out of the scope of the pilot study and thus not included in this thesis.

1.4 Ethical Considerations

While this thesis is theoretical in nature, it was conducted with a strictly conceptual application toward autonomous drone coordination in collaboration with Saab AB, a company in the defense sector. The coordination of autonomous drones has both civilian applications such as building communication networks and military applications such as coordinating drone swarms in battle. This dual-use nature of the technology is a risk which we acknowledge, but we argue that establishing robust baselines for training is a prerequisite for the safe deployment of autonomous drones in any domain. By establishing baselines for how we expect autonomous drones to behave, we ensure a degree of predictability over the autonomous system. Consequently, this thesis does not directly advance a specific military capability, but instead contributes to the foundational safety and predictability of drone swarms.

1.5 Reading Guide

This thesis covers two distinct topics: linear quadratic games and simulation based learning. Chapter 1 presents a joint introduction with background and research questions for both topics. In Chapters 2 and 3 theory and results pertaining to linear quadratic games is presented and discussed to answer research questions 1 and 2. Chapter 4 is dedicated to theory and results pertaining to simulation based learning to answer research questions 3 and 4. Chapter 5 presents conclusions for both topics.

1.6 AI Usage in this Thesis

During the writing of this thesis, large language models (LLMs) were used to aid in proofreading. However, no LLM or generative AI model was used to generate text or content detached from ideas formulated by the authors.

2

Modeling Multi-Agent Interaction

In this chapter, we define *stochastic games* as a framework for modeling multi-agent interactions and derive the Riccati iteration algorithm to compute solutions for a subset of stochastic games: *linear quadratic* (LQ) games. Following this, we provide a brief overview of *multi-agent reinforcement learning* (MARL), as an alternative solution method to complicated stochastic games in general. While an overarching aim of this thesis is to explore whether solutions to linear quadratic games are suitable for evaluating solutions generated by MARL, this chapter focuses on establishing the theoretical foundation necessary to understand how these solutions are derived.

2.1 Stochastic Games

Stochastic games are used to model state-based environments in which several decision makers, called *players* or *agents*, take *actions* that probabilistically evolve the state and yield rewards to each player. A general stochastic game is defined as the tuple $(I, X, U, q, (r_i)_{i\in I})$ in accordance with [10, Definition 4.1], where

- $I = \{1, ..., N\}$ is a index set of $N \in \mathbb{N}$ players;
- X is a set of states, usually a subspace of a Euclidean space, $X \subseteq \mathbb{R}^n$, where $n \in \mathbb{N}$ is the number of state variables;
- $U_i \subseteq \mathbb{R}^{m_i}$ is the action space for player $i \in I$ with $m_i \in \mathbb{N}$ action variables, and $U_i(x) \subseteq U_i$ is its admissible actions in state $x \in X$. U is defined as the joint action space of all players $U := U_1 \times \cdots \times U_N$;
- $q(x(t+1) \mid x(t), u(t))$ is a stochastic kernel on X given $X \times U$;
- $r_i \colon X \times U \to \mathbb{R}$ is the one-step reward for player $i \in I$.

For the purposes of this thesis, a *stochastic kernel* in the above definition is a conditional probability density function. Specifically, $q(x(t+1) \mid x(t), u(t))$ represents the probability density of the next state $x(t+1) \in X$ given $x(t) \in X$ and $u(t) \in U$. We also require that

$$\int_{x(t+1)\in X} q(x(t+1) \mid x(t), u(t))\, \mathrm{d}x(t+1) = 1 \quad \text{for all } x(t) \in X \text{ and } u(t) \in U. \tag{2.1}$$

A stochastic game proceeds as follows: the game begins in some initial state $x(0) \in X$. At time $t \in \mathbb{N}$, each player $i \in I$ observes the state $x(t) \in X$. Having observed the state, they independently and simultaneously choose an individual action $u_i(t) \in U_i(x)$, resulting in the *joint action* $u(t) = (u_1, u_2, \dots u_N) \in U$. A player's choice of action depends on the *policy* of that player. In general, a policy is a mapping from all previous states and actions to current actions. In our case, however, we are only interested in deterministic stationary Markov policies, where the action only depends on the current state. These policies are defined as $\pi_i \colon X \to U_i$ which assigns to each state $x \in X$ some admissible action $u_i \in U_i(x)$. Player i receives an individual reward $r_i(x(t), u(t))$ dependent on the current state and their chosen action. The game state then transitions to the next state $x(t+1)$ which is sampled from the stochastic kernel $q(x(t+1) \mid x(t), u(t))$. These steps are repeated until the game terminates after reaching some terminal state $x_{\text{term}} \in X$ or reaching some maximum time $t = T$. Games can also be non-terminating, meaning that they continue for an infinite number of time steps [3, Chapter 3.3]. In the following sections we define a subset of stochastic games called *stochastic discrete-time linear quadratic games* and derive so called *Nash equilibrium* policies.

2.1.1 Linear Quadratic Games

Stochastic discrete-time linear quadratic games are a subset of stochastic games where the game state evolves linearly and the one-step reward functions are quadratic. One reason to study these games is that they allow for exact solutions, which is in line with the purpose of this work: to calculate baseline results with which to compare novel algorithmic solution methods. While general linear quadratic games are not necessarily discrete in time nor stochastic, we refer to stochastic discrete-time linear quadratic games simply as linear quadratic games or LQ-games for the rest of this thesis.

The dynamics governing state transitions of linear quadratic games is defined as the linear difference equation

$$x(t+1) = Ax(t) + \sum_{i \in I} B_i u_i(t) + w(t), \tag{2.2}$$

where time is indexed by $t \in \mathbb{N}$, $x(t) \in X \subseteq \mathbb{R}^n$ is the state at time t and $u_i(t) \in U_i(x) \subseteq \mathbb{R}^{m_i}$ is the action of player i at time t. The number of dimensions m_i of the action space can differ for each agent, however throughout this work the players are assumed to have the same action dimension therefore we write, $m_i = m$ for all players i. The terms $(w(t))_{t \geq 0}$ are zero-mean, independent and identically normally distributed random variables independent of $x(t)$. The matrix $A \in \mathbb{R}^{n \times n}$, describes the underlying environment dynamics, while $B_i \in \mathbb{R}^{n \times m}$ describes how actions u_i affect the state.

The individual quadratic one-step reward function $r_i \colon X \times U \to \mathbb{R}$ measures how good a particular action is in a particular state, and is defined as

$$r_i(x(t), u(t)) = x(t)^\top Q_i x(t) + \sum_{j \in I} u_j(t)^\top R_{ij} u_j(t), \tag{2.3}$$

where $u(t) = (u_j(t))_{j \in I}$ is the joint action at time t. The matrix Q_i describes how much player i values the current state $x(t)$ and matrix R_{ij} describes the reward that player i receives for the action taken by player $j \in I$. The matrix $Q_i \in \mathbb{R}^{n \times n}$ is symmetric and negative semi-definite, and the matrices $R_{ij} \in \mathbb{R}^{m \times m}$ are symmetric such that R_{ii} is negative definite and $R_{ij}, i \neq j$ are negative semi-definite. These assumptions guarantee that the reward has an upper bound for all players, since

$$\begin{aligned} x^\top Q_i x \leq 0, \forall x \in \mathbb{R}^n, \\ u^\top R_{ij} u \leq 0, \forall u \in \mathbb{R}^m, \end{aligned} \tag{2.4}$$

by the definition of negative definiteness.

For a player $i \in I$, a policy $K_i \in \mathbb{R}^{m \times n}$ is a linear map $K_i \colon X \to U_i$ which assigns an action $u_i = K_i x \in U_i(x)$ to each state $x \in X$. We define the *joint policy* $K_I = (K_i)_{i \in I}$ as the tuple containing the policies of all players. We also define $K_{I_{-i}} = (K_j)_{j \in I_{-i}}$ as the tuple of all policies except for K_i, where $I_{-i} = I \backslash \{i\}$. Consider the total reward player i accumulates when starting in some state $x(0) \in X$ and uses some policy K_i with $K_{I_{-i}}$ fixed. We call this the *value* for player i and define the *value function* as

$$V_i(x(0), K_i, K_{I_{-i}}) = \mathbb{E}_w \left[\sum_{t=0}^{\infty} \beta^t r_i(x(t), u(t)) \right], \tag{2.5}$$

where $\beta \in [0, 1)$ is the so-called the *discount factor* and determines the present value of future rewards, and $u(t) = (u_i(t))_{i \in I} = (K_i x(t))_{i \in I}$ is the joint action of all players. This value function is a measurement of how well a policy K_i performs, given an initial state $x(0)$ and assuming other players use policies $K_{I_{-i}}$. Given that each player wants to maximize its value, we define the *feedback Nash equilibrium* solution K_I^{NE} to the game as the joint policy $K_I^{\mathrm{NE}} = (K_i^{\mathrm{NE}})_{i \in I}$ such that

$$V_i(x(0), K_i, K_{I_{-i}}^{\mathrm{NE}}) \leq V_i(x(0), K_i^{\mathrm{NE}}, K_{I_{-i}}^{\mathrm{NE}}) \tag{2.6}$$

holds for any other policy K_i and for all players $i \in I$. A feedback policy refers to players determining their actions based only on the current state at each time t, contrary to an *open-loop* policy where all actions are determined from the starting state $x(0)$. Since open-loop policies aren't considered in this thesis, we omit the term 'feedback' and refer to feedback Nash equilibrium simply as Nash equilibrium (NE). The Nash equilibrium solution can be interpreted as a joint policy in which no player can improve its value by unilaterally changing its policy. The following sections are dedicated to deriving the Nash equilibrium solution K_I^{NE}, initially for a one-player game, and then for an arbitrary number of players.

2.1.2 Deriving Optimal One-Player Policy

We begin by considering a single-player linear quadratic game, where the dynamics and one-step reward function respectively are given by

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t) + w(t), \\ r_i(x(t), u(t)) &= x(t)^\top Q x(t) + u(t)^\top R u(t). \end{aligned} \tag{2.7}$$

The value function for a policy $K \in \mathbb{R}^{m \times n}$ is given by

$$V(x(0), K) = \mathbb{E}_w \left[\sum_{t=0}^{\infty} \beta^t \left(x(t)^\top Q x(t) + u(t)^\top R u(t) \right) \right], \tag{2.8}$$

where $\beta \in [0, 1)$, $Q \in \mathbb{R}^{n \times n}$ is negative semi-definite and $R \in \mathbb{R}^{m \times m}$ is negative definite. In order to derive the policy that maximizes the value function, we use methods of dynamic programming. Let $v^*(x) := \sup_{K \in \mathbb{R}^{m \times n}} V(x, K)$ be the unique function which satisfies the discounted reward dynamic programming equation (DPE) [11, Chapter 5]

$$v^*(x(t)) = \max_{u(t) \in U} r(x(t), u(t)) + \beta \mathbb{E}_w[v^*(x(t+1))]. \tag{2.9}$$

To solve the DPE, we make the following ansatz:

$$v^*(x) = x(t)^\top P x(x) + \gamma, \tag{2.10}$$

where $P \in \mathbb{R}^{n \times n}$ is a negative definite matrix that fulfills $P = P^\top$ and $P \prec 0$, and where $\gamma \in \mathbb{R}$. For notational clarity we set $x = x(t)$ and suppress the t index in the rest of the derivation. Plugging the ansatz into equation (2.9) and substituting $x(t+1)$ with the expression in equation (2.2), we get that

$$x^\top P x + \gamma = \max_{u \in \mathbb{R}^m} x^\top Q x + u^\top R u + \beta \mathbb{E}_w \left[\Big(Ax + Bu + w \Big)^\top P \Big(Ax + Bu + w \Big) + \gamma \right]. \tag{2.11}$$

The expectation can be expanded and rewritten as

$$\begin{aligned} &\mathbb{E}_w \left[\Big(Ax + Bu + w \Big)^\top P \Big(Ax + Bu + w \Big) + \gamma \right] = \\ =&\mathbb{E}_w \Big[(Ax + Bu)^\top P (Ax + Bu) + w^\top P (Ax + Bu) + (Ax + Bu)^\top P w + w^\top P w \Big] + \gamma \\ =&(Ax + Bu)^\top P (Ax + Bu) + \mathbb{E}_w[w^\top P w] + \gamma, \end{aligned}$$

where we use the fact that w has zero mean and is independent of x and u. Using the linearity of the trace operator and expectation, as well as the cyclic permutation property of the trace operator, $\mathrm{Tr}(AB) = \mathrm{Tr}(BA)$, the last term can be rewritten further as

$$\mathbb{E}_w[w^\top P w] = \mathbb{E}_w[\mathrm{Tr}(w^\top P w)] = \mathbb{E}_w[\mathrm{Tr}(P w w^\top)] = \mathrm{Tr}(P \mathbb{E}_w[w w^\top]) = \mathrm{Tr}(P \,\mathrm{Cov}(w)),$$

where the last equality uses that $\mathbb{E}_w[w w^\top]$ is the covariance matrix for w. The right-hand side (r.h.s.) can thus be written as

$$\text{r.h.s.} = \max_{u \in \mathbb{R}^m} x^\top Q x + u^\top R u + \beta (Ax + Bu)^\top P (Ax + Bu) + \beta \mathrm{Tr}(P \,\mathrm{Cov}(w)) + \beta \gamma.$$

Since Q, R and P are negative (semi) definite, the r.h.s. is bounded from above and can be maximized by setting the gradient of the expression, with respect to u, to zero:

$$\nabla_u (\text{r.h.s.}) = 2Ru + 2\beta B^\top P B u + 2\beta B^\top P A x = 0.$$

Solving for u gives the candidate optimal action law

$$\hat{u} = -\beta(R + \beta B^\top P B)^{-1} B^\top P A x = Kx, \tag{2.12}$$

which gives us an expression for the optimal policy K as

$$\begin{aligned} K &= -\beta(R + \beta B^\top P B)^{-1} B^\top P A, \\ K &= -\beta F B^\top P A, \end{aligned} \tag{2.13}$$

where F is introduced as $F = (R + \beta B^\top P B)^{-1}$ for notational convenience. Substituting $u = Kx$ back into the equation (2.11) we get

$$\begin{aligned} x^\top P x + \gamma =& x^\top Q x + x^T K R K x + \beta(Ax + BKx)^\top P(Ax + BKx) \\ &+ \beta \mathrm{Tr}(P \operatorname{Cov}(w)) + \beta\gamma. \end{aligned}$$

By identifying all terms multiplied with $x^\top$ from the left and x from the right on both sides we get an equation for P

$$\begin{aligned} x^\top P x &= x^\top \left(Q + K^\top R K + \beta(A + BK)^\top P(A + BK) \right) x, \\ \Rightarrow P &= Q + K^\top R K + \beta(A + BK)^\top P(A + BK), \end{aligned}$$

by factoring out x. Expanding the parentheses and reordering terms we get

$$\begin{aligned} P &= Q + K^\top R K + \beta(A + BK)^\top P(A + BK), \\ &= Q + \beta A^\top P A + \beta A^\top P K B + \beta B^\top K^\top P A + K^\top R K + \beta K^\top B^\top P B K, \\ &= Q + \beta A^\top P A + \beta A^\top P K B + \beta B^\top K^\top P A + K^\top \underbrace{\left(R + \beta B^\top P B \right)}_{= F^{-1}} K, \end{aligned}$$

and since we can substitute the K in the last term with $K = -\beta F B^\top P A$ from (2.13) we finally get that

$$\begin{aligned} P &= Q + \beta A^\top P A + \beta A^\top P K B + \beta K^\top B^\top P A - \beta K^\top B^\top P A, \\ &= Q + \beta A^\top P A + \beta A^\top P K. \end{aligned}$$

We also identify the constant terms on both sides and rewrite them as

$$\gamma = \beta \mathrm{Tr}(P \operatorname{Cov}(w)) + \beta\gamma \tag{2.14}$$

$$\Leftrightarrow \gamma = \frac{\beta}{1 - \beta} \mathrm{Tr}(P \operatorname{Cov}(w)). \tag{2.15}$$

We therefore know that the ansatz (2.11) holds if we can solve equations

$$P = Q + \beta A^\top P A - \beta^2 A^\top P B (R + \beta B^\top P B)^{-1} B^\top P A, \tag{2.16}$$

$$\gamma = \frac{\beta}{1 - \beta} \mathrm{Tr}(P \operatorname{Cov}(w)), \tag{2.17}$$

under the assumption that P is a symmetric matrix. Note that the matrix P does not depend on the random variable w because we assumed that it has mean zero.

This by extension means that K also does not depend on w according to (2.13). By setting $\tilde{A} = \sqrt{\beta}A$ and $\tilde{R} = \frac{1}{\beta}R$ we arrive at an equation known as the *discrete algebraic Riccati equation* (DARE)

$$P = Q + \tilde{A}^\top P\tilde{A} - \tilde{A}^\top PB(\tilde{R} + B^\top PB)^{-1}B^\top P\tilde{A}. \tag{2.18}$$

We assume that $R \prec 0$, $Q \preceq 0$ and (A, B) are a *stabilizable pair* i.e. all $\lambda \in \sigma(A), \lambda \geq 1$ are *controllable* by B. An eigenvalue λ to A is controllable with respect to B if there exists a matrix K such that $A + BK$ can change λ arbitrarily. If these assumptions are met then we can find a symmetric matrix P^* that solves the DARE [12]. For more details about the solutions to the Riccati equation, see Appendix A.1. With P^* we can construct the optimal policy K^* which maximizes the value function (2.5).

To summarize: from equation (2.12) we see that the optimal control policy K is given by

$$K = -\beta(R + \beta B^\top PB)^{-1}B^\top PA, \tag{2.19}$$

where P is obtained by solving the DARE in equation (2.18). Note also that the expected value when using the policy K^* is $v^* = x(0)^\top Px(0) + \gamma$. This result will now be used to derive Nash equilibria for the N-player case.

2.2 Deriving the Nash Equilibrium and the Riccati Iteration Algorithm

Recall the dynamics and one-step reward function given by equations (2.2) and (2.3), respectively, as

$$\begin{aligned} x(t+1) &= Ax(t) + \sum_{i \in I} B_i u_i(t) + w(t), \\ r_i(x(t), u(t)) &= x(t)^\top Q_i x(t) + \sum_{j \in I} u_j(t)^\top R_{ij} u_j(t). \end{aligned}$$

Given a joint policy K_I, we initially restrict ourselves to finding the *best response* policy K_i^{BR} for player i while fixing the other players policies $K_{I_{-i}}$. A best response policy K_i^{BR} is the policy which maximizes the value for player i given that all other policies $K_{I_{-i}}$ are fixed. From equation (2.12) we have that

$$u_j(t) = K_j x(t) \tag{2.20}$$

for all $j \in I_{-i}$. This allows us to rewrite the dynamics in equation (2.2) as

$$x(t+1) = \left(A + \sum_{j \in I_{-i}} B_j K_j\right) x(t) + B_i u_i(t) + w(t). \tag{2.21}$$

Similarly we can rewrite the one-step reward function $r_i(x(t), u(t))$ for player i as

$$r_i(x(t), u(t)) = x(t)^\top \left(Q_i + \sum_{j \in I_{-i}} K_j^\top R_{ij} K_j\right) x(t) + u_i(t)^\top R_{ii} u_i(t).$$

By introducing the matrices $\hat{A}_i$ and $\hat{Q}_i$ given by

$$\hat{A}_i = A + \sum_{j \in I_{-i}} B_j K_j, \tag{2.22}$$

$$\hat{Q}_i = Q_i + \sum_{j \in I_{-i}} K_j^\top R_{ij} K_j, \tag{2.23}$$

the dynamics and reward function for multiple players can, respectively, be rewritten as

$$\begin{aligned} x(t+1) &= \hat{A}_i x(t) + B_i u_i(t) + w(t), \\ r_i(x(t), u(t)) &= x(t)^\top \hat{Q}_i x(t) + u_i(t)^\top R_{ii} u_i(t). \end{aligned} \tag{2.24}$$

Since the policies $K_{I_{-i}}$ are fixed, the matrices $\hat{A}_i$ and $\hat{Q}_i$ are fixed as well, essentially turning this into a single-player game for player i. By following the same steps as described in Section 2.1.2 we get that the best response K_i^{BR} with fixed policies $K_{I_{-i}}$ is given by

$$K_i^{\mathrm{BR}} = -\beta(R_i + \beta B_i^\top P_i^{\mathrm{BR}} B_i)^{-1} B_i^\top P_i^{\mathrm{BR}} \hat{A}_i, \tag{2.25}$$

with P_i^{BR} being the solution to the DARE

$$P_i^{\mathrm{BR}} = \hat{Q}_i + \tilde{\hat{A}}_i^\top P_i^{\mathrm{BR}} \tilde{\hat{A}}_i - \tilde{\hat{A}}_i^\top P_i^{\mathrm{BR}} B_i (\tilde{R}_i + B_i^\top P_i^{\mathrm{BR}} B_i)^{-1} B_i^\top P_i^{\mathrm{BR}} \tilde{\hat{A}}_i. \tag{2.26}$$

The joint policy where all players' policies are the best response policies to all other players' policies is equivalent to the Nash equilibrium solution K_I^{NE} in (2.6) that we seek to find. In other words, in our Nash equilibrium solution, all policies K_I^{NE} have to simultaneously fulfill the system of equations

$$\begin{aligned} K_1^{\mathrm{NE}} &= -\beta(R_{11} + \beta B_1^\top P_1^{\mathrm{NE}} B_1)^{-1} B_1^\top P_1^{\mathrm{NE}} \hat{A}_1, \\ &\vdots \\ K_N^{\mathrm{NE}} &= -\beta(R_{NN} + \beta B_N^\top P_N^{\mathrm{NE}} B_N)^{-1} B_N^\top P_N^{\mathrm{NE}} \hat{A}_N, \end{aligned} \tag{2.27}$$

with $(P_i^{\mathrm{NE}})_{i \in I}$ satisfying

$$\begin{aligned} P_1^{\mathrm{NE}} &= \hat{Q}_1 + \tilde{\hat{A}}_1^\top P_1^{\mathrm{NE}} \tilde{\hat{A}}_1 - \tilde{\hat{A}}_1^\top P_1^{\mathrm{NE}} B_1 (\tilde{R}_1 + B_1^\top P_1^{\mathrm{NE}} B_1)^{-1} B_1^\top P_1^{\mathrm{NE}} \tilde{\hat{A}}_1, \\ &\vdots \\ P_N^{\mathrm{NE}} &= \hat{Q}_N + \tilde{\hat{A}}_N^\top P_N^{\mathrm{NE}} \tilde{\hat{A}}_N - \tilde{\hat{A}}_N^\top P_N^{\mathrm{NE}} B_N (\tilde{R}_N + B_N^\top P_N^{\mathrm{NE}} B_N)^{-1} B_N^\top P_N^{\mathrm{NE}} \tilde{\hat{A}}_N. \end{aligned} \tag{2.28}$$

Here, we denote P_i^{NE} as the specific DARE solution which corresponds to K_i^{NE}. Note that the system of algebraic Riccati equations are coupled through $\hat{A}$ and $\hat{Q}$, since they are constructed using all other players' policies. Currently, there exists no analytical method for solving the system of coupled Riccati equations. However, since the best-response policy of a single player can be computed while keeping the policies of all other players fixed, an iterative algorithm can be constructed in which each agent sequentially updates its policy by computing its best response to the current policies of the other agents. This procedure is known as the iterative Riccati

Algorithm 1 Iterative Riccati algorithm for Nash equilibria.

Require: $A, B, Q, R, K_I^{(0)}, \beta, \varepsilon$
Ensure: $Q_{ii} \preceq 0$
Ensure: $R_{ii} \prec 0, i \in I$
Ensure: $R_{ij} \preceq 0, i \neq j$
Ensure: (A, B_i) is stabilizable $i \in I$
Ensure: $\beta \in [0, 1)$
difference $\leftarrow \infty$
$j \leftarrow 1$
while difference $> \varepsilon$ **do**
  for $i \in I$ **do**
    $\hat{A}_i \leftarrow A + \sum_{q=1}^{i-1} B_q K_q^{(j)} + \sum_{\ell=i+1}^{n} B_\ell K_\ell^{(j-1)}$
    $\hat{Q}_i \leftarrow Q_i + \sum_{q=1}^{i-1} (K_q^{(j)})^\top R_{iq} K_q^{(j)} + \sum_{\ell=i+1}^{n} (K_\ell^{(j-1)})^\top R_{i\ell} K_\ell^{(j-1)}$
    $\tilde{\hat{A}}_i \leftarrow \sqrt{\beta} \hat{A}_i$
    $\tilde{R}_{ii} \leftarrow \frac{1}{\beta} R_{ii}$
    $P_i \leftarrow \texttt{solve\_discrete\_are}\left(\tilde{\hat{A}}_i, B_i, \hat{Q}_i, \tilde{R}_{ii}\right)$
    $K_i^{(j)} \leftarrow -\beta (R_{ii} + \beta B_i^\top P_i B_i)^{-1} B_i^\top P_i \hat{A}_i$
  end for
  difference $\leftarrow \max_{i \in I} \left(||K_i^{(j-1)} - K_i^{(j)}|| \right)$
  $j \leftarrow j + 1$
end while

algorithm and is described in Algorithm 1. Specifically, one iteration of the algorithm corresponds to solving equations (2.25)–(2.26) for every agent, sequentially incorporating the previously updated policies such that the matrices $\hat{A}_i$ and $\hat{Q}_i$ for player i, at iteration j of the algorithm are

$$\hat{A}_i^{(j)} = A + \sum_{q=1}^{i-1} B_q K_q^{(j)} + \sum_{\ell=i+1}^{n} B_\ell K_\ell^{(j-1)}, \tag{2.29}$$

$$\hat{Q}_i^{(j)} = Q_i + \sum_{q=1}^{i-1} (K_q^{(j)})^\top R_{iq} K_q^{(j)} + \sum_{\ell=i+1}^{n} (K_\ell^{(j-1)})^\top R_{i\ell} K_\ell^{(j-1)}. \tag{2.30}$$

For ease of notation, we define $K_i^{(j)}$ as the best response policy for player i at iteration j of the algorithm. We are however required to initialize the algorithm at iteration $j = 0$ with some joint policy K_I. This is done by setting $K_i^{(0)} = 0$ for all $i \in I$. The algorithm is terminated if

$$||K_i^{(j+1)} - K_i^{(j)}|| < \varepsilon, \tag{2.31}$$

for all $i \in I$ and for some tolerance $\varepsilon > 0$. This means that no significant change of policies is made between iterations and since each policy update yields a higher reward for the updating player, the termination criteria is equivalent to finding the Nash equilibrium K_I^{NE} in (2.6), where no player can unilaterally improve their reward.

Line 7 in Algorithm 1 refers to the `solve_discrete_are()` function from the python library Scipy which calculates the solution to the DARE [13]. The variable `difference` in the Algorithm 1 is initially set to some number greater than the tolerance ensuring that the while loop runs at least once. For a programmatic implementation this could be `MAX_INT` or simply $\varepsilon + 1$.

2.2.1 Algorithm Limitations

This algorithm currently has no *a priori* convergence guarantee. Nortmann et al. have, however, shown that if a Nash equilibrium is found then there exists a region of attraction around it which is locally asymptotically stable [14]. When this region of attraction is entered, the algorithm is guaranteed to converge to the Nash equilibrium. The properties of this region and how it is affected by the problem setup is currently unknown. Experimentally, for specific problem setups such as those described in Chapter 3, initializing all policies $K_i^{(0)} = 0$ results in convergence to a Nash equilibrium. It is known that a general LQ-game can have more than one Nash equilibrium [15]. Despite this we have not been able to find more than one Nash equilibrium for the specific problems described in Chapter 3.

2.3 Multi-agent Reinforcement Learning (MARL)

The purpose of the thesis is to derive Nash equilibria for LQ-games in order to obtain a reliable baseline for evaluating novel multi-agent reinforcement learning (MARL) algorithms. Stochastic games are suitable as underlying models of MARL problems where multiple adaptive agents learn to maximize individual goals that may be both cooperative and competing [16]. MARL algorithms seek to find equilibrium policies in an approximate way via a process of trial and error, which is called learning. Using equivalent index notation as in Section 2.1.1, the general learning process involves players $1, \dots, N$ using some joint policy $\pi_I = (\pi_{\theta_1}, \dots, \pi_{\theta_N})$, to make actions, observe the state and receive rewards for a number of independent runs of a game. We call these independent runs *episodes*. The experiences, i.e. states, rewards and actions, are collected in the so-called *experience replay buffer* $\mathcal{D} = (x(t), u(t), r(t), x(t+1))_{t=1,\dots\mathcal{D}_{\text{size}}}$, where $\mathcal{D}_{\text{size}}$ is the size of the buffer, and are used by a MARL algorithm $\mathbb{L}$ to update the joint policy according to

$$\pi_I^{\text{new}} = \mathbb{L}\left(\mathcal{D}, \pi_I^{\text{old}}\right).$$

In MARL it is not always assumed players have access to global state information as the players in LQ-games do. Instead they may rely on local, partial observations of the state space. For the purposes of this thesis, however, we will assume that all players have full state observability. While the optimal policy for LQ-games is a strictly linear mapping, the policies used in MARL is typically a non-linear function approximated by a feed-forward neural network such as a Multi-Layer Perceptron (MLP). Player i's policy $\pi_{\theta_i}(u_i(t) \mid x(t))$ represents the probability density of choosing action $u_i(t)$ given state $x(t)$. Here, the parameters θ_i represent the weights and biases of the network layers. MARL has some challenges, one being the non-stationarity caused by the players continually adapting their policies to the changing

policies of other players. This can be contrasted with the difficulty of solving the coupled Riccati equations (2.28). A further challenge is multi-agent credit assignment: determining which agents' actions were responsible for the received reward [3, Section 1.4]. To address these challenges, several MARL algorithms have been developed using distinct learning paradigms, ranging from independent learning and fully decentralized schemes to centralized training with decentralized execution (CTDE). In this thesis, we limit ourselves to describing one such algorithm called *Multi-Agent Proximal Policy Optimization* (MAPPO) [17].

The MAPPO-algorithm is characterized by using centralized training and decentralized execution. This means that every player uses an individual policy network π_{θ_i}, parametrized by θ_i, to produce actions from state observations. These individual policies however, are trained by a separate, centralized *critic*-network parametrized by ϕ. The critic has access to full state information as well as all actions taken by every player and is in itself trained to estimate the reward for actions taken in certain states. The centralized training framework aims to address the problem of multi-agent credit assignment. If a group of players have symmetric reward functions, a method called *policy sharing* allows the players to share the same policy network which makes training more efficient [17, Section 3.3]. This is however not considered in this thesis.

MAPPOs learning process iterates between a data collection phase and a training phase. MAPPO is an *on-policy* algorithm which means that the policies are trained on data collected using only the previous policy iteration. This differs from *off-policy* algorithms where the policy can be trained on data collected from any previous iteration of the policy [3, Section 2.6]. During the data collection phase an episode is initialized with some state $x(0)$, usually sampled from some probability distribution. At time step t each player i receives an observation of the state $x(t)$ and samples an action $u_i(t)$ using its policy $\pi_{\theta_i}(u_i(t) \mid x(t))$. In MAPPO, the policy is stochastic, meaning that a state maps to a probability distribution on actions, rather than a deterministic action. This enables exploration during training by introducing a probability of choosing different actions in the same state. During each time step the critic also calculates an estimation $V_\phi(x(t))$ of how valuable the current state is. The joint action $u(t) = (u_1(t), \dots, u_N(t))$ is used to transition to state $x(t+1)$ and a vector of rewards $r(t) = (r_1(x(t), u_1(t)), \dots, r_N(x(t), u_N(t)))$ is returned. A vector $(x(t), u(t), r(t), V_\phi(x(t)), x(t+1))$ containing information about the time step is stored in the experience replay buffer $\mathcal{D}$ and the process repeats for the fixed number of time steps which defines an episode. Next, a new episode is initiated. This process is repeated over several episodes until the buffer is full, ending the data collection phase.

During the training phase the critic uses the accumulated rewards to compute advantage estimates $\hat{A}_i(t)$. These estimates quantify the relative benefit of actions taken compared to the average expected outcome for those states. Following this, the current policy parameters θ_i are adjusted by calculating the gradient of the *Proximal Policy Optimization* (PPO) clipped surrogate objective

$$L_{\theta_i}^{\text{CLIP}} = \mathbb{E}_t \left[\min \left(\rho_{\theta_i}(t) \hat{A}_i(t), \text{clip}(\rho_{\theta_i}(t), 1-\epsilon, 1+\epsilon) \hat{A}_i(t) \right) \right]$$

where $\rho_{\theta_i}(t) = \frac{\pi_{\theta_i}(u_i(t)|x(t))}{\pi_{\theta_{i,\text{old}}}(u_i(t)|x(t))}$ is the probability ratio of choosing action $u(t)$ under the new policy π_{θ_i} and the old policy $\pi_{\theta_{i,old}}$, and $\epsilon > 0$ is a hyperparameter determining the clipping range. It describes taking the minimum value between the probability ratio and the clipped probability ratio where the clipping function clip() maps the probability ratio to the interval $[1-\epsilon, 1+\epsilon]$. This ensures that the policy updates remain in a stable range. The gradient of this objective function indicates how the weights should change to increase the probability of actions with positive advantages. Simultaneously, the critic parameters ϕ are updated by minimizing the mean squared error between the network's value prediction $V_{\phi_i}(x(t))$ and the discounted rewards observed in the buffer. With that, one iteration of training is complete and the next iteration proceeds using the updated policies. For a more detailed description of the MAPPO training process, see [17, Appendix A].

3

Scenario Formulations and Results

In this chapter we initially present two scenarios in which MARL policies are evaluated against Nash equilibrium policies. How the MARL-policies were implemented and trained is described as well as metrics used for the comparison. The results are then presented and discussed.

3.1 Scenario Formulations

This section introduces two linear quadratic games: a two-player leader-follower scenario and a five-player scenario which simulates a swarm of drones. Note that while these are formulated as LQ-games, they are denoted using the broader term 'scenarios' to avoid confusion when later discussing them in a MARL context. For each scenario the state vector, action vector and necessary matrices corresponding to the scenario dynamics and player rewards are defined.

3.1.1 Scenario 1: Two-Player Leader-Follower

A leader-follower scenario is a scenario in which one player acts as *leader* and one or more players act as *followers*. In the following scenario we have two players acting in a 2D environment. The leader aims to reach a certain goal position, and the followers' goal is staying as close to the leader as possible.

The state vector $x(t) \in \mathbb{R}^{10}$ with 10 state variables at time $t \in \mathbb{N}$ is defined as

$$x(t) = \begin{bmatrix} \mathbf{p}_L(t)^\top & \mathbf{v}_L(t)^\top & \mathbf{p}_F(t)^\top & \mathbf{v}_F(t)^\top & \mathbf{g}(t)^\top \end{bmatrix}^\top, \tag{3.1}$$

where L and F stands for leader and follower respectively. The vectors $\mathbf{p}_i(t) = [p_{i,x}(t) \quad p_{i,y}(t)]^\top$ and $\mathbf{v}_i(t) = [v_{i,x}(t) \quad v_{i,y}(t)]^\top$ contain the position and velocity respectively in the x- and y-directions for players $i \in \{L, F\}$ at time t. Similarly $\mathbf{g}(t) = [g_x(t) \quad g_y(t)]^\top$ contains the goal position of the leader in the x- and y-directions at time t. The corresponding index for each state is summarized in Table 3.1. The action vector $u_i(t) \in \mathbb{R}^2$ has two input variables for each player $i \in \{L, F\}$ and is defined as

$$u_i(t) = \begin{bmatrix} a_{i,x}(t) & a_{i,y}(t) \end{bmatrix}^\top. \tag{3.2}$$

Table 3.1: State vector index mapping for the leader-follower scenario with $N = 2$ players.

State	Indices
$\mathbf{p}_L,\ \mathbf{v}_L$	$x_1, \ldots, x_4$
$\mathbf{p}_F,\ \mathbf{v}_F$	$x_5, \ldots, x_8$
$\mathbf{g}$	$x_9,\ x_{10}$

The elements $a_{i,x}(t)$ corresponds to acceleration applied in the x-direction to player $i \in I$, and $a_{i,y}(t)$ corresponds to acceleration applied in the y-direction at time t. The players' movement should follow the Newtonian laws of motion such that

$$p_{i,d}(t+1) = p_{i,d}(t) + hv_{i,d}(t) + \frac{h^2}{2}a_{i,d}(t), \tag{3.3}$$

$$v_{i,d}(t+1) = v_{i,d}(t) + ha_{i,d}(t), \tag{3.4}$$

where $d \in \{x, y\}$ since these equations hold in both the x and y direction. The constant h is the time discretization, in this example $h = 0.1$. Note that $t + 1$ corresponds to the index of the next time step and not a time step of size 1. The $\frac{h^2}{2}$ factor in Equation (3.3) has to do with the position being the second order primitive of the acceleration, since the acceleration is assumed to be constant during each time step. Further detail can be found in Appendix A.2. The goal position should stay constant such that

$$\mathbf{g}(t+1) = \mathbf{g}(t). \tag{3.5}$$

The dynamics is encoded into the matrix A. The *double integrator block* is defined as

$$A_{\text{dyn}} = \begin{bmatrix} 1 & 0 & h & 0 \\ 0 & 1 & 0 & h \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.6}$$

with the full dynamics matrix A as

$$A = \text{blkdiag}(A_{\text{dyn}}, A_{\text{dyn}}, I_2) \in \mathbb{R}^{10\times 10}. \tag{3.7}$$

Here, blkdiag is the block matrix obtained by placing the corresponding block matrices on the main diagonal. The two A_{dyn} blocks govern the dynamics of the leader and the follower while the identity block keeps the goal position constant. The *control dynamics block* is defined as

$$B_{\text{dyn}} = \begin{bmatrix} \frac{h^2}{2} & 0 \\ 0 & \frac{h^2}{2} \\ h & 0 \\ 0 & h \end{bmatrix}. \tag{3.8}$$

The full control dynamics matrix $B_i \in \mathbb{R}^{10\times 2}$ is constructed by inserting B_{dyn} at the four rows corresponding to player i and zeros elsewhere:

$$B_L = \begin{bmatrix} B_{\text{dyn}} \\ 0_{6\times 2} \end{bmatrix}, \quad B_F = \begin{bmatrix} 0_{4\times 2} \\ B_{\text{dyn}} \\ 0_{2\times 2} \end{bmatrix}. \tag{3.9}$$

Using these matrices with equation (2.2) gives us the desired laws of motion (3.3)–(3.4) and keeps the goal position constant. The noise vector $w(t) \in \mathbb{R}^{10}$ has elements

$$w_i(t) \sim \begin{cases} \mathcal{N}(0, 0.005), & i \in \{1, \dots, 8\}, \\ 0, & i \in \{9, 10\}. \end{cases} \tag{3.10}$$

This means that noise is only applied to the position and velocity variables for each player and not the goal positions.

The one-step reward functions differ for the players in this scenario. The leader should be rewarded for minimizing the distance to the goal position while also minimizing the size of its actions. The one-step reward function for the leader is

$$r_L\left(x(t), u(t)\right) = -\,\omega_g \left((p_{L,x} - g_x)^2 + (p_{L,y} - g_y)^2\right) - \omega_u \|u_L\|^2, \tag{3.11}$$

where parameters $\omega_g, \omega_u > 0$ scale the reward associated with the goal and action term respectively. The follower wishes to minimize the distance between itself and the leader while also minimizing the size of its actions. Similarly to $r_L(x(t), u(t))$ this is written as

$$r_F\left(x(t), u(t)\right) = -\,\omega_g \left((p_{L,x} - p_{F,x})^2 + (p_{L,y} - p_{F,y})^2\right) - \omega_u \|u_F\|^2. \tag{3.12}$$

For this scenario the weights are set to $\omega_g = 1$, $\omega_u = 1$. Rewriting $r_L(x(t), u(t))$ and $r_F(x(t), u(t))$ in terms of Q_i and R_{ij} as in (2.3) with $i, j \in \{L, F\}$, the Q_i matrices are 10×10 and sparse. For a reward term $-\omega(x_a - x_b)^2$ the contribution to Q_i is $Q_i[a, a] = -\omega$, $Q_i[b, b] = -\omega$, $Q_i[a, b] = Q_i[b, a] = \omega$. By setting $Q_i[a, b] = Q_i[b, a] = \omega$ we ensure that Q_i is symmetric. The Q_L matrix therefore has the following entries:

$$\begin{aligned} Q_L[k, k] &= -\omega_g = -1, \quad k \in \{1, 2, 9, 10\}, \\ Q_L[9, 1] &= Q_L[1, 9] = \omega_g = 1, \\ Q_L[10, 2] &= Q_L[2, 10] = \omega_g = 1. \end{aligned} \tag{3.13}$$

The indices are from matching terms from the one-step reward function (3.11) with the associated indices in Table 3.1. Similarly, Q_F has the entries

$$\begin{aligned} Q_F[k, k] &= -\omega_g = -1, \quad k \in \{1, 2, 5, 6\}, \\ Q_F[5, 1] &= Q_F[1, 5] = \omega_g = 1, \\ Q_F[6, 2] &= Q_F[2, 6] = \omega_g = 1. \end{aligned} \tag{3.14}$$

The action reward matrices are

$$\begin{aligned} R_{LL} = R_{FF} &= \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}, \\ R_{LF} = R_{FL} &= \mathbf{0}. \end{aligned} \tag{3.15}$$

Setting $R_{ij} = 0$ for $i \neq j$ is a modeling choice and implies that the players will only be rewarded for the size of their own actions and not the actions of other players. Note that this also implies that $\hat{Q}_i = Q_i$ for all $i \in I$ which simplifies calculations in (2.23) somewhat. Finally, we use a discount factor $\beta = 0.99$.

3.1.2 Scenario 2: Five-Player Swarm

In order to test the scalability of the comparison, a second scenario with five players was set up. Each player seeks to approach its individual goal position while simultaneously maintaining proximity to other players. In this scenario there are five players and five goals yielding 30 state variables corresponding to positions and velocities for the five players, as well as positions of the five goals. The state vector $x(t) \in \mathbb{R}^{30}$ is defined as

$$x(t) = \begin{bmatrix} \mathbf{p}_1(t)^\top & \mathbf{v}_1(t)^\top & \dots & \mathbf{p}_5(t)^\top & \mathbf{v}_5(t)^\top & \mathbf{g}_1(t)^\top & \dots & \mathbf{g}_5(t)^\top \end{bmatrix}^\top, \tag{3.16}$$

where $\mathbf{p}_i(t)$, $\mathbf{v}_i(t)$ and $\mathbf{g}_i(t)$ are defined as in 3.1.1. The corresponding index for each state is summarized in Table 3.2. Each player $i \in I$ can change its own acceleration in the x- and y-directions, meaning that there are 2 actions for every player. The action vector is

$$u_i(t) = \begin{bmatrix} a_{i,x}(t), & a_{i,y}(t) \end{bmatrix}^\top. \tag{3.17}$$

Using the notation set up in Section 3.1.1, the dynamics matrix A is given by

$$A = \text{blkdiag}(A_{\text{dyn}}, A_{\text{dyn}}, A_{\text{dyn}}, A_{\text{dyn}}, A_{\text{dyn}}, I_2, I_2, I_2, I_2, I_2) \tag{3.18}$$

Table 3.2: State vector index mapping for the five-player swarm scenario with $N = 5$ players.

State	Indices
$\mathbf{p}_1,\ \mathbf{v}_1$	$x_1, \dots, x_4$
$\mathbf{p}_2,\ \mathbf{v}_2$	$x_5, \dots, x_8$
$\mathbf{p}_3,\ \mathbf{v}_3$	$x_9, \dots, x_{12}$
$\mathbf{p}_4,\ \mathbf{v}_4$	$x_{13}, \dots, x_{16}$
$\mathbf{p}_5,\ \mathbf{v}_5$	$x_{17}, \dots, x_{20}$
$\mathbf{g}_1$	$x_{21},\ x_{22}$
$\mathbf{g}_2$	$x_{23},\ x_{24}$
$\mathbf{g}_3$	$x_{25},\ x_{26}$
$\mathbf{g}_4$	$x_{27},\ x_{28}$
$\mathbf{g}_5$	$x_{29},\ x_{30}$

where the A_{dyn} blocks govern the dynamics of the players and the identity matrix blocks keeps the goals constant. The control matrices B_i are

$$B_1 = \begin{bmatrix} B_{\text{dyn}} \\ 0_{26\times 2} \end{bmatrix}, \quad B_2 = \begin{bmatrix} 0_{4\times 2} \\ B_{\text{dyn}} \\ 0_{22\times 2} \end{bmatrix}, \quad B_3 = \begin{bmatrix} 0_{8\times 2} \\ B_{\text{dyn}} \\ 0_{18\times 2} \end{bmatrix}, \quad B_4 = \begin{bmatrix} 0_{12\times 2} \\ B_{\text{dyn}} \\ 0_{14\times 2} \end{bmatrix}, \quad B_5 = \begin{bmatrix} 0_{16\times 2} \\ B_{\text{dyn}} \\ 0_{10\times 2} \end{bmatrix}. \tag{3.19}$$

The noise vector $w(t) \in \mathbb{R}^{30}$ has elements

$$w_i(t) \sim \begin{cases} \mathcal{N}(0, 0.005), & i \in \{1, \dots, 20\}, \\ 0, & i \in \{21, \dots, 30\}. \end{cases} \tag{3.20}$$

This means that noise is only applied to position and velocity variables but not goal positions. Every player $i \in I$ is rewarded for its proximity to other players as well as the proximity to their own goal. The reward functions are

$$\begin{aligned} r_i(x(t), u(t)) = &- \omega_g \left((p_{i,x} - g_{i,x})^2 + (p_{i,y} - g_{i,y})^2 \right) \\ &- \omega_o \sum_{j \in I_{-i}} \left[(p_{i,x} - p_{j,x})^2 + (p_{i,y} - p_{j,y})^2 \right] - \omega_u \|u_i\|^2, \end{aligned} \tag{3.21}$$

For this scenario the weights are $\omega_g = 1$, $\omega_o = 0.2$, $\omega_u = 1$. For notational clarity, when constructing the reward matrices we define the set $I_{p_x} = \{1, 5, 9, 13, 17\}$ and $I_{p_y} = \{2, 6, 10, 14, 18\}$ as the sets of state indices for the x and y positions respectively. The total position index set is defined as $I_p = I_{p_x} \cup I_{p_y}$. For player i, the sets $I_{i,p} := \{k_{i,p_x}, k_{i,p_y}\} = \{4i - 3, 4i - 2\}$ and $I_{i,g} := \{k_{i,g_x}, k_{i,g_y}\} = \{19 + 2i, 20 + 2i\}$ is defined as their position and goal indices respectively. For player i the state reward matrix Q_i has the nonzero rows/columns at indices of all positions and their goal, which is the set $I_p \cup I_{i,g}$. The diagonal elements at these indices are

$$Q_i[k, k] = -\omega_g - 4\omega_o = -1.8, \quad k \in I_{i,p}, \tag{3.22}$$

$$Q_i[k, k] = -\omega_o = -0.2, \quad k \in I_p \backslash I_{i,p}, \tag{3.23}$$

$$Q_i[k, k] = -\omega_g = -1, \quad k \in I_{i,g}, \tag{3.24}$$

with the off-diagonal elements being

$$Q_i[k_{i,p_x}, k_{i,g_x}] = Q_i[k_{i,g_x}, k_{i,p_x}] = \omega_g = 1, \tag{3.25}$$

$$Q_i[k_{i,p_y}, k_{i,g_y}] = Q_i[k_{i,g_y}, k_{i,p_y}] = \omega_g = 1, \tag{3.26}$$

$$Q_i[k_{i,p_x}, j] = Q_i[j, k_{i,p_x}] = \omega_o = 0.2, \quad j \in I_{p_x} \backslash k_{i,p_x}, \tag{3.27}$$

$$Q_i[k_{i,p_y}, j] = Q_i[j, k_{i,p_y}] = \omega_o = 0.2, \quad j \in I_{p_y} \backslash k_{i,p_y}. \tag{3.28}$$

This captures the terms associated with the state in the one-step reward function $r_i(x(t), u(t))$ in equation (3.21). The action reward matrices for $i \in I, j \in I_{-i}$ are

$$R_{ii} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}, \quad R_{ij} = R_{ji} = \mathbf{0},$$

where once again the matrices R_{ij} are the zero matrix for $i \neq j$. Finally the discount factor $\beta = 0.99$.

3.1.3 Feasibility of Scenarios

The R_{ii} matrices being diagonal with only negative elements fulfill the criteria of being negative definite since all eigenvalues of R_{ii} are negative. The Q_i matrices also fulfill the criteria for being negative semi definite. We also have to ensure that $(A, \quad [B_1 \dots B_N])$ are a stabilizable pair. Since $[B_1 \dots B_N]$ can control any state in $x(t)$ except for the constant terms associated with the goal position, we only have to ensure that the eigenvalues λ corresponding to the constant terms fulfill $|\lambda| < 1$. This condition is fulfilled since we use $\beta < 1$ which leads to $\tilde{A} = \sqrt{\beta}A$ in equation (2.18) to have eigenvalues less than one corresponding to the constant terms.

3.2 MARL Implementation Details

This section describes how the scenarios described in Section 3.1 were implemented in a MARL framework, how the MARL models were trained and how their performance was evaluated against the Nash equilibrium policies. To simulate the scenarios, the *Vectorized Multi-agent Simulator* (VMAS) was used. VMAS uses PyTorch vectorization which allows for several different environments using the same instructions set to be run in parallel, making training more efficient. The simulator is integrated in the MARL library *BenchMARL* which was used for training. For both scenarios, the BenchMARL implementation of the MAPPO algorithm was used to train MLP models, each containing two fully connected layers of 256 neurons each. A LeakyReLU activation function was used, making the MLPs non-linear. Other activation functions could be used but LeakyReLU was deemed sufficient for this use case [18]. To account for the assumptions inherent in the LQ formulation of the scenarios, some adjustments were made to the physics engine in VMAS, as well as in the configuration of MAPPO. One adjustment made to the MAPPO algorithm was to allow full-state observation for every agent at every time step. As discussed in Section 2.3, agents typically only have access to local observations. In LQ-games however, agents have access to full state information. To make the comparison fair between the MARL-policies and the NE-policies, we set $o_i(t) = x(t)$, for all times t and all players i. In the VMAS physics, friction was removed to accommodate the dynamics described by equations (3.3)–(3.4).

3.2.1 Training Setup

To avoid confusion we will henceforth refer to the derived Nash equilibrium policies as *R-policies* where the R denotes Riccati. To make MARL-policies and R-policies comparable, the MARL models are trained on the quadratic reward functions described in Sections 3.1.1–3.1.2. This however posed a challenge in the implementation. The positional reward term $x_t^\top Q_i x_t$ in the linear quadratic reward function rewards the quadratic distance to some specified positional state at each time step. This is generally ill suited for policy gradient algorithms such as MAPPO due to the large gradients that are produced during the initial exploration phase of training, where quadratic distances can grow too large for the optimizer to handle. To remedy this, we initially scale the reward to be 0.1% of the actual reward. This reduces the

size of the gradients during training which improves stability. The scaling was then gradually increased to 1 to avoid gradients becoming too small as players adapted to the objective. Too small gradients cause very conservative policy updates which, while it is a sign of stability, also slows training. To achieve further stability, the action space was clamped to the range $[-10, 10]$. Since the R-policies assume an unbounded action space, the bounds were chosen after analyzing the maximum magnitude of actions taken by the R-policies and ensuring that the R-policies will never be limited by the clamped action space.

The numerical values of the hyperparameters used during training are found in Table 3.3 and were chosen as they were the standard settings in BenchMARL and excessive hyperparameter tuning was beyond the scope of this thesis. As mentioned in Section 2.3, parameter sharing was disabled in order to derive individual policies for each player, as this mirrors the linear quadratic case.

Table 3.3: Hyperparameters used during training. For details, see [17].

Hyperparameter	Value
Clipping parameter (ϵ)	0.2
Entropy coefficient	0.001
GAE lambda (λ)	0.9
Discount factor (β)	0.99
Learning rate (α)	0.0001
Batch size ($\mathcal{D}_{\text{size}}$)	15000
Num minibatches (N_{mb})	20
Num training epochs (N_e)	8
Optimizer	Adam
Adam epsilon	1e-6
Num parallel environments	100
Evaluation	
Evaluation interval	2
Num evaluation episodes	100

3.3 Method and Metrics for Comparison

The main metric by which we evaluate the MARL-policies against the R-policies is value during a game instance. och tog bort *The inter-agent value comparison* (IAVC) is the relative difference between a specific agents' value using the R-policy and its value using the MARL-policy, during a specific game instance. *The aggregated-agent value* (AAV) is the sum of all agents' values for a specific game instance, using a specified joint policy, and the *aggregated agent value comparison* (AAVC) is the relative difference of AAV using the R-joint policy and the AAV using the MARL joint policy. We also qualitatively evaluate the policies by inspecting the trajectories they choose during game instances. A game instance is an independent run of a game.

To accomplish this, *state-action histories* of 30000 simulated game instances were collected using each joint policy:

$$h^j = \left[x(0)^\top, u(0)^\top, x(1)^\top, u(1)^\top, \ldots, x(T)^\top, u(T)^\top\right]^\top,$$

where $u(t)$ is the joint action vector, and $x(t)$ is the state vector at time $t \in \{0, \ldots, T\}$, and T is the terminal time.

Initially, a data set containing 30000 starting states $x(0) \sim U([-1, 1]^n)$, where n is the number of state variables was sampled for each scenario. For each sampled starting state and each joint policy, a game instance with terminal time $T = 150$ was simulated using VMAS, and the state-action history from the episode was collected. After all game instances were simulated, the state-action histories for each joint policy were collected in two vectors:

$$h_{\mathrm{R}} = (h_{\mathrm{R}}^1, \ldots, h_{\mathrm{R}}^{30000}), \tag{3.29}$$

$$h_{\mathrm{MARL}} = (h_{\mathrm{MARL}}^1, \ldots, h_{\mathrm{MARL}}^{30000}). \tag{3.30}$$

The histories were used to compute the value for each game instance j and each player i in accordance with equation (2.5)

$$V_{\mathrm{R}}^{j,i} = \sum_{t=0}^{T} \beta^t r_i(x(t), u(t)), \tag{3.31}$$

$$V_{\mathrm{MARL}}^{j,i} = \sum_{t=0}^{T} \beta^t r_i(x(t), u(t)), \tag{3.32}$$

where $r_i \colon X \times U \to \mathbb{R}$ is the player $i \in I$ and scenario specific quadratic reward function. Finally, the inter-agent value was calculated for each player i and game instance j as

$$V_{\mathrm{inter}}^{j,i} = \frac{V_{\mathrm{R}}^{j,i} - V_{\mathrm{MARL}}^{j,i}}{|V_{\mathrm{R}}^{j,i}|}. \tag{3.33}$$

We define the aggregate agent values for game instance j as

$$V_{\mathrm{agg,R}}^{j} = \sum_{i \in I} V_{\mathrm{R}}^{j,i}, \tag{3.34}$$

$$V_{\mathrm{agg,MARL}}^{j} = \sum_{i \in I} V_{\mathrm{MARL}}^{j,i}, \tag{3.35}$$

and the aggregated-agent value comparison for game instance j was calculated as

$$V_{\mathrm{agg}}^{j} = \frac{V_{\mathrm{agg,R}}^{j} - V_{\mathrm{agg,MARL}}^{j}}{|V_{\mathrm{agg,R}}^{j}|}. \tag{3.36}$$

These were the metric used for evaluating the performance of MARL-policies against R-policies.

3.4 Results

In this section the comparison between the performance of Nash equilibrium policies and policies trained by MAPPO are presented for each of the two scenarios described in Section 3.1. For each scenario, the comparison of inter-agent value (3.33) as well as the comparison of aggregated agent value (3.36) is presented.

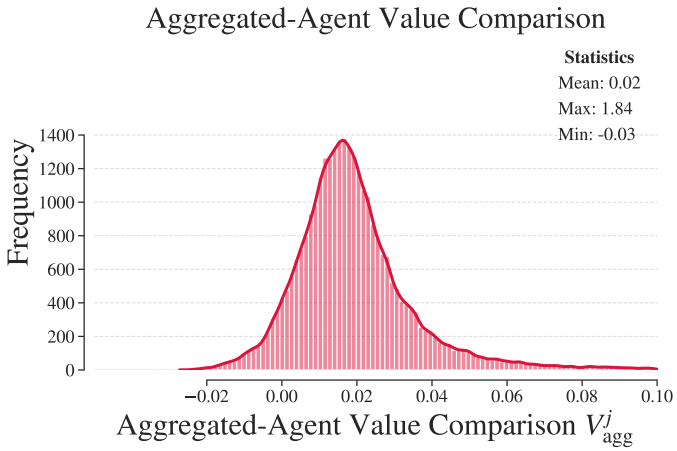
3.4.1 Results and Comparison for Scenario 1

For Scenario 1, the overall mean difference between the policies was negligible, however, R-policies consistently outperformed MARL-policies across a majority of scenario instances, both in terms of inter-agent value, and aggregated agent value. Table 3.4 describes the mean, maximal and minimal AAVC and IAVC for each player, over all scenario instances and Figure 3.1 shows the distributions of IAVC and AAVC over all scenario instances.

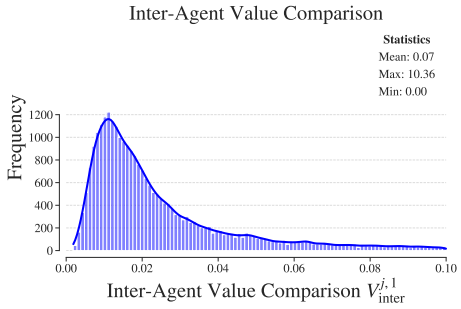
Table 3.4: Comparison of inter-agent value and aggregated agent value for Scenario 1.

	Mean	Max	Min	% of instances where MARL > R
Leader (IAVC)	0.07	10.36	0.0	0.0
Follower (IAVC)	0.03	1.04	-0.14	22.9
AAVC	0.02	1.84	-0.03	5.0

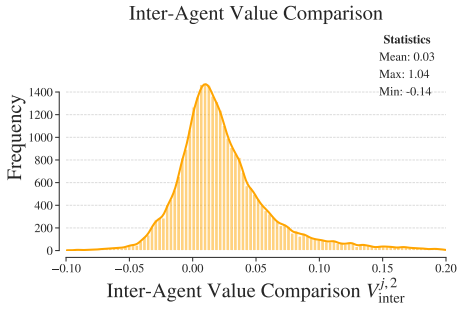
The low mean suggest that the MARL-policies are well trained, even outperforming the R-policies, in 5.0% of scenario instances when considering AAVC. The leader's reward function in this scenario is not dependent on the follower. From the leaders point of view it is, in essence, a single-player game for which the R-policy is guaranteed to maximize the expected value according to Section 2.1.2. It is then reasonable and expected that the R-policy outperforms the MARL-policy in every scenario instance. The follower's MARL-policy, on the other hand, outperformed the R-policy in 22.9% of the scenario instances. Due to the follower's dependence on actions taken by the leader, a slight deviation from optimal actions by the leader can be exploited by the follower. This in turn, leads to higher value for the follower using the MARL policy compared to the follower using the R-policy. The MARL-policies performed better than the R-policies in 5.0% of cases when considering the aggregated agent value. This is due to the follower's MARL-policy outperforming its respective R-policy. The mean IAVC and AAVC was low however, and we can conclude that the policies generally produce equal value. In Figures 3.2c–3.2d, the players' trajectories during the scenario instance which produced the maximal AAVC are shown. In this scenario instance both players' starting positions were very close to the goal which disadvantaged the players using MARL-policies. This is because finetuning the position when close to the goal was observed as general weakness for the MARL-policies. In Figures 3.2a and 3.2b the players' trajectories during the scenario instance in which the AAVC was minimal are shown.



(a) Distribution of AAVC over 30000 instances of Scenario 1.



(b) Distribution of IAVC for the leader over 30000 instances of Scenario 1.



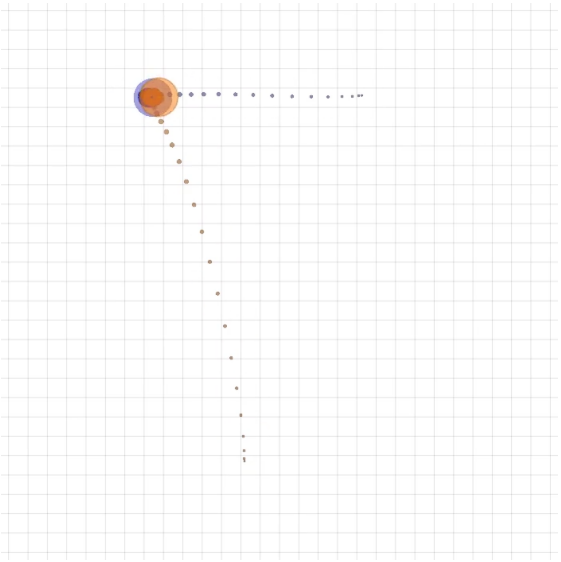
(c) Distribution of IAVC for the follower over 30000 instances of Scenario 1.

Figure 3.1: Distribution of AAVC and IAVC over 30000 instances of Scenario 1.

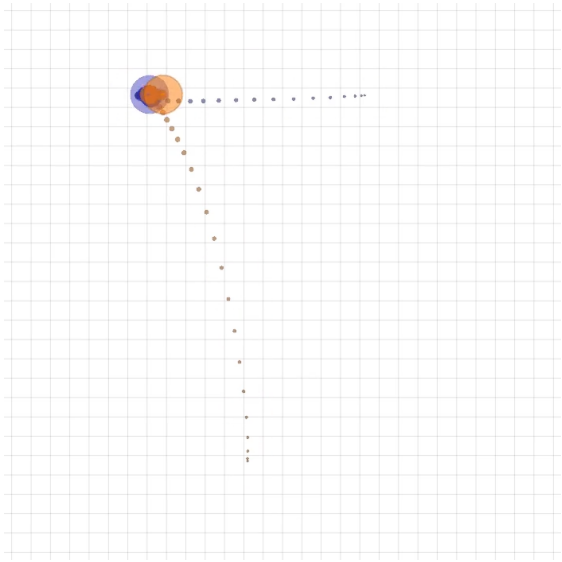
3.4.2 Results and Comparison for Scenario 2

For Scenario 2, the MARL-policies generally produced the same value as the R-policies and even outperformed the R-policies in a majority of scenario instances. The mean, minimum and maximum AAVC as well as IAVC for all players and over all scenario instances, are described in Table 3.5. An initial observation is that despite players having symmetrical reward functions, they have trained different policies. This is seen most clearly in the differing percentage of times each player's MARL-policies beats their respective R-policy. Players 3 and 4 outperform their respective R-policies a significantly smaller proportion of scenario instances. The distributions of IAVC over all games is shown for each player in Figures 3.4b–3.4f, and the distribution AAVC is shown in Figure 3.4a.

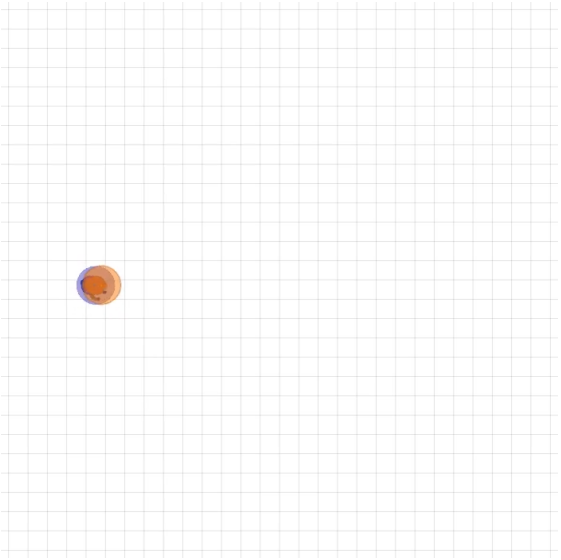
In general the MARL-policies tended to perform worse than the R-policies in scenario instances where the goal positions were in close proximity to each other, as indicated by Figures 3.3a–3.3b. However, they performed better in scenario instances where the goal positions were spread apart, as shown in Figures 3.3c–3.3d. Possible reasons for this are discussed in further detail in Section 3.4.3, but one explanation is that the MARL-policies may be biased toward cooperation and thus



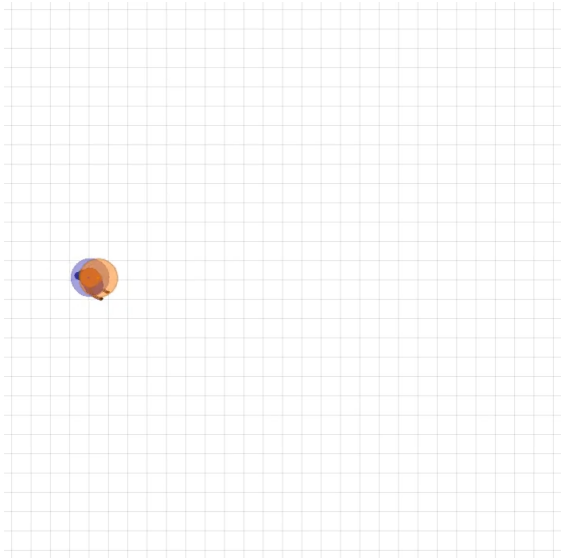
(a) Trajectories for players using R-policies in the instance of Scenario 1 which produced the minimum AAVC.



(b) Trajectories for players using MARL-policies in the instance of Scenario 1 which produced the minimum AAVC.

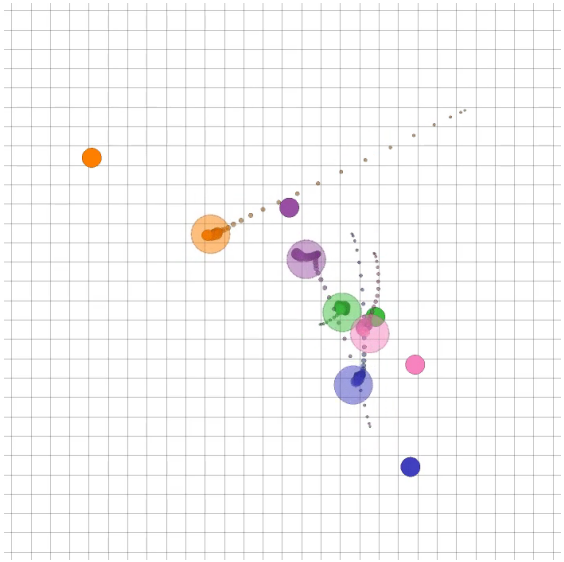


(c) Trajectories for players using R-policies in the instance of Scenario 1 which produced the maximum AAVC.

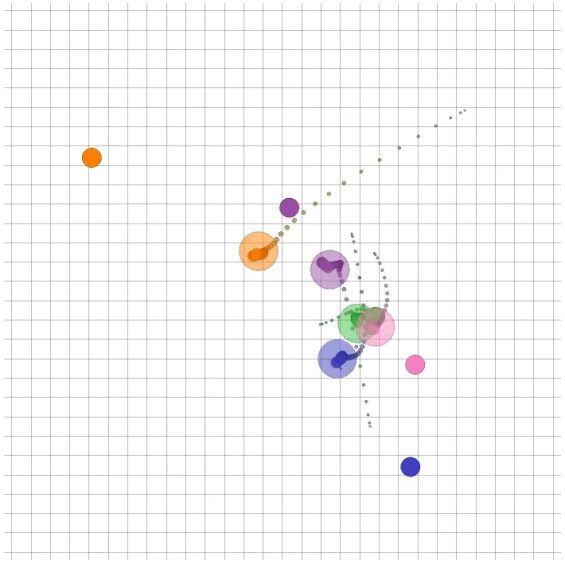


(d) Trajectories for players using MARL-policies in the instance of Scenario 1 which produced the maximum AAVC.

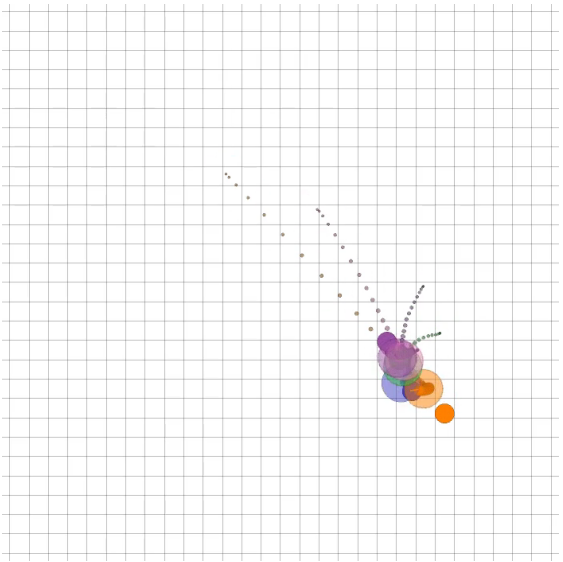
Figure 3.2: Trajectories for Leader-follower games with maximum and minimum AAVC. The leader is blue and the follower is orange. The target destination for the leader is the smaller blue circle.



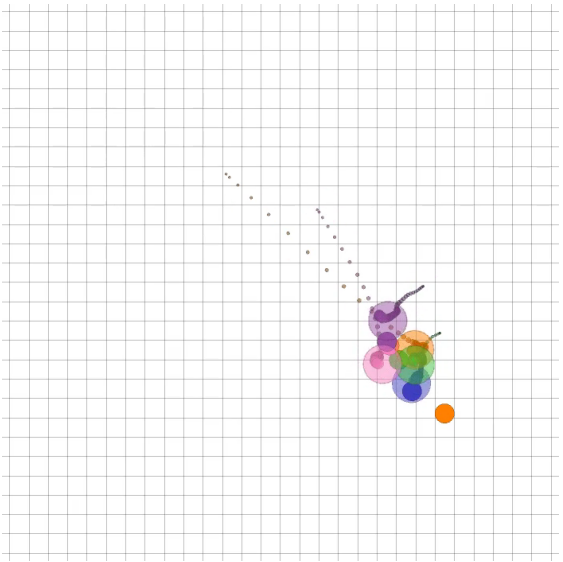
(a) Trajectories for players using R-policies in the instance of Scenario 2 which produced the minimum AAVC.



(b) Trajectories for players using MARL-policies in the instance of Scenario 2 which produced the minimum AAVC.



(c) Trajectories for players using R-policies in the instance of Scenario 2 which produced the maximum AAVC.



(d) Trajectories for players using MARL-policies in the instance of Scenario 2 which produced the maximum AAVC.

Figure 3.3: Trajectories for the instance of scenario 2 with maximum and minimum aggregate agent value. Player 1 is blue, player 2 is orange, player 3 is green, player 4 is pink and player 5 is purple. Target destinations for each player have corresponding colors.

Table 3.5: Comparison of inter-agent value and aggregated agent value for Scenario 2.

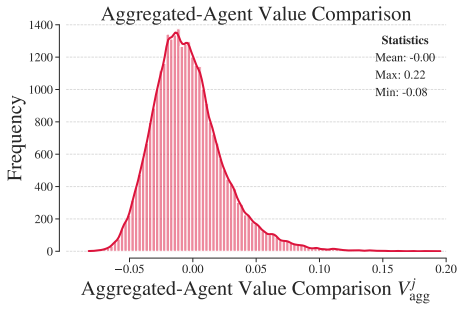
	Mean	Max	Min	% of instances where MARL > R
Player 1 (IAVC)	-0.02	0.56	-0.38	63.6
Player 2 (IAVC)	-0.01	0.86	-0.39	59.6
Player 3 (IAVC)	0.01	0.71	-0.35	46.6
Player 4 (IAVC)	0.01	0.53	-0.36	42.3
Player 5 (IAVC)	-0.03	0.65	-0.37	66.9
AAVC	0.00	0.22	-0.08	64.2

tend to favor staying close to each other over staying close to their individual goal. This is in contrast to the purely self-interested nature of Nash equilibrium policies such as the R-policies. In scenario instances where the goals are in close proximity, the eventual cooperative loss incurred by the self-interested R-policies is diminished. As stated in Section 3.4.1, a general difficulty which the MARL-policies exhibited was finetuning their position when close to the goal. One reasonable explanation for the variance shown in Figures 3.4 is the trade off between the precision of the R-policies and the collaborative gain of the MARL-policies.

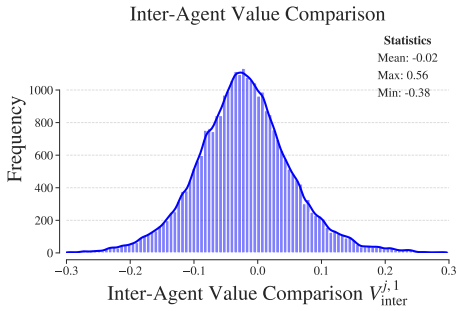
3.4.3 Discussion

In this section we aim to discuss the question of whether derived Nash equilibrium solutions to LQ-games are suitable as a baseline for evaluating MARL algorithms. Based on the results, there are a few points to consider in this regard. A general observation from the two simulated Scenarios is that the degree to which players objectives are coupled affect the degree to which the MARL-policies can outperform R-policies. This is due to the nature of the Nash equilibrium as a solution concept. A Nash equilibrium is an inherently non-cooperative solution concept as it assumes each player aims to maximize its individual value. While it provides a good baseline for individual stability, as evidenced by the leader in Scenario 1, it does not necessarily take into account eventual increases in aggregated value which could be achieved through a greater degree of cooperation. While it is possible to formulate individual quadratic reward functions that encourage cooperation, using the individual reward functions as defined in 3.1.2 is still of interest since it elucidates how the same reward function can give rise to a different solution concept when using MARL.

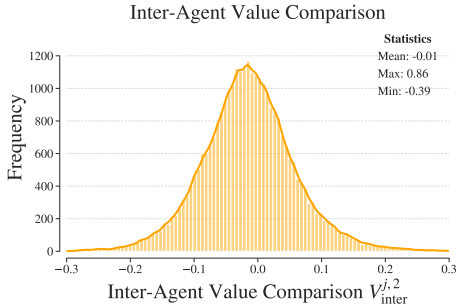
It also elucidates that using R-policies as a baseline for more complex cooperative tasks may be misleading as there can exist more efficient policies which rely on cooperation. This is evidenced by Scenario 2, where the more cooperative MARL-policies outperformed Nash equilibrium policies in 64.2% of the scenario instances. MAPPO, using a centralized training scheme, facilitates cooperation as it aims to maximize an objective function which is shared between all players [17]. This is characteristic of a *social* optimum, in which the optimized reward function is the sum of all individual reward functions, rather than a Nash equilibrium [3, Chapter 4]. One indication of this is that players 3 and 4 in Scenario 2 outperform the



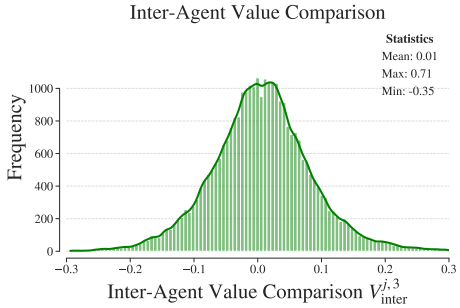
(a) Distribution of AAVC over 30000 instances of scenario 2.



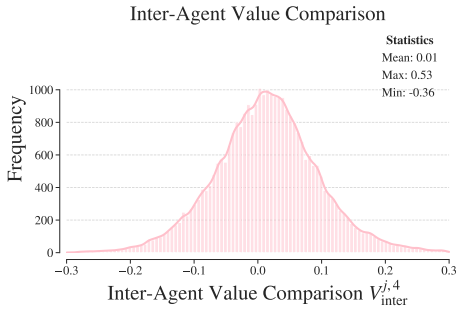
(b) Distribution of IAVC for player 1 over 30000 instances of scenario 2.



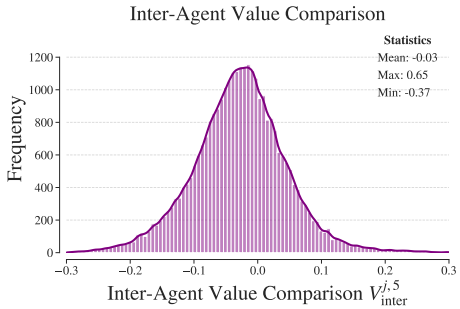
(c) Distribution of IAVC for player 2 over 30000 instances of scenario 2.



(d) Distribution of IAVC for player 3 over 30000 instances of scenario 2.



(e) Distribution of IAVC for player 4 over 30000 instances of Scenario 2.



(f) Distribution of IAVC for player 5 over 30000 instances of Scenario 2.

Figure 3.4: Distributions of AAVC and IAVC for each player, over 30000 instances of Scenario 2

R-policies in fewer scenario instances than the other players, suggesting that their policies have been trained to be more cooperative, giving up some individual reward to increase the cooperative reward. In a Nash equilibrium, this is impossible due to the aforementioned self-interested nature of NE-policies.

Another point to consider is the restrictive nature of LQ-games. When evaluating MARL-algorithms, we may want to measure their ability to handle scenarios which cannot be modeled using the LQ framework, due to the restriction of having a linear system and quadratic reward functions. Constraints such as collision or obstacle avoidance, temporal constraints, as well as limited communication range between players are all limitations when modeling LQ-games. A point of interest when evaluating MARL algorithms is how well they handle partial state observations. While there are methods of implementing partial observability into the linear quadratic framework [14, Section IV], it is unclear whether the methods presented in this thesis are a suitable benchmark for partial observability. Another assumption we make in order for the derivation of Nash equilibria to be feasible is that the action space of the LQ-game is continuous and unbounded. In addition to introducing the inability to modeling discrete actions, this assumption may also be unsuitable to assume an unbounded action space, partly because it is not physical, and partly because it may lead to large control signals and instability during training.

While the R-policies for LQ-games fail to capture many interesting real world scenarios we may want to model, they are not completely without utility. The R-policies represents the outcome when all players are purely self-interested which in itself is an interesting baseline to consider. If a MARL algorithm is incapable of reaching the performance of the R-policies in scenarios in which the players are self-interested, it can be an indication that adjustments are needed to the MARL algorithm. Another strength of the R-policies as a baseline is their performance in the presence of additive Gaussian noise with zero mean. As shown in 2.1.2, the Nash equilibrium policies maximize the expected value independent of the presence of noise. While there are more sophisticated methods to handle noise which can be employed in MARL-algorithms [19], the noise resistance of R-policies can nonetheless be useful as a baseline for how well an algorithm handles noise.

It is appropriate to reiterate the limitations inherent in the Riccati iteration algorithm. While the algorithm converged for the specific scenario formulations that are presented in this thesis, there is no guarantee that the algorithm converges for any general formulation of a linear quadratic game [14]. Finding more predictable algorithms for calculating Nash equilibria should be further investigated.

4

Simulation-Based Learning

In this chapter we present theory about the importance of knowledge sharing, relevant learning theories and ending with simulation-based learning. Then we present the methodology and results of an exploratory pilot study into the effectiveness of simulation-based learning as a method for knowledge sharing at an engineering company. The results of this study is presented and then discussed. Note that this chapter does not build on Chapters 2 and 3, and is only tangentially related to them.

4.1 Theory

This section presents theory about knowledge sharing in the context of an organization, as well as relevant learning theories and simulation-based learning.

4.1.1 Knowledge Sharing

As stated in the introduction, innovation can be described as the process by which information is acquired, assimilated and shared to create new knowledge that is embodied in products and services [7]. Innovation is also a viable market strategy which can lead to differentiation and competitive advantages for the company [20]. A facilitator for knowledge creation and innovation is effective methods for knowledge sharing. The *SECI*-model describes different processes for transferring knowledge [21]. The model states that knowledge can either be tacit or explicit and that there are four processes for sharing knowledge.

- Socialization describes transferring tacit knowledge from one individual to another, becoming tacit knowledge within the other person.
- Externalization translates tacit knowledge from an individual and makes it explicit, for example through writing a book.
- Combination refers to explicit knowledge from different sources being collected and synthesized into new explicit knowledge.
- Internalization takes explicit knowledge and makes it tacit, such as when an individual reads a written document and converts the explicit knowledge to their own tacit knowledge.

The key take away is, firstly, how tacit knowledge held by employees can be articulated and shared with others to facilitate knowledge transfer. Secondly, how knowledge creation can be viewed as a spiral process that progresses through the four stages. Within an organization, socialization generates ideas that are subsequently externalized and combined, eventually leading employees to internalize the newly created knowledge. This, in turn, initiates a new cycle of knowledge-creation.

One way to generate novel ideas is through inspiration from other disciplines since it can lead to questioning our own assumptions [21, p.78]. A potential challenge with this is that different disciplines use distinct mental models and may abstract away knowledge in different ways. One way to bridge the gap between different knowledge sharing methods across specialized professions is through *boundary objects*. A boundary object is something that acts as a bridge and allows individuals with different specializations to collaborate efficiently despite their varied backgrounds. The concept of a boundary object was initially described as a way to think about the structure of ill-structured paths towards a solution in scientific practices, where scientists *"cooperate without having good models of others' work; successfully work together while employing different units of analysis, methods of aggregating data, and different abstractions of data; cooperate while having different goals, time horizons, and audiences to satisfy"* [22], [23]. Examples of boundary objects for innovation management are prototypes, drawings, sketches and simulation models, among many others [22].

4.1.2 General Theories of Learning

Knowledge can be conceptualized into different dimensions, ranging from basic forms such as recalling facts to advanced applications such as creation. A framework useful for differentiating knowledge into different cognitive domains is Bloom's taxonomy [24]. It originally consisted of six noun-based categories, but was later reworked into six verb-based categories instead [25], indicating how the knowledge is used. These different cognitive domains are *remembering, understanding, applying, analyzing, evaluating* and *creating*. The first three are considered lower-order thinking skills, while the latter three are considered higher-order thinking skills, which enable a greater degree of synthesis.

Since learning and teaching is prevalent in early life, they are often exclusively associated with children. Therefore, well-executed education is frequently thought of as *pedagogical*. It is important to realize, however, that we do not cease to learn as we age, but the priorities of learning changes. This introduces a counterpart to pedagogy, aptly named *andragogy*, which is the study of how adults learn. Several concepts effective for teaching children also transfer to adult learning, such as good structure and clear learning objectives [26, Chapter 4]. A major difference, however, between teaching children and adults is what motivates them to learn. Self-direction is vital for adults, as they want to be in control of their learning. Furthermore, adults require an understanding of the purpose or applicability of the content in order to be receptive to learning [27].

4.1.3 Simulation-Based Learning

David Kolb, with his *experiential learning theory*, described learning as *"the process whereby knowledge is created through the transformation of experience"* [28, p.38]. This process can be thought of as a cycle with four stages consisting of *concrete experience*, *reflective observation*, *abstract conceptualization*, and *active experimentation*. Effective learning occurs as a person iterates through these stages, converting experiences into abstract concepts that are then tested in new scenarios [28, p.40–42]. One approach to implementing such experiential learning is through the use of *simulations*. Simulation is a *technique* used in several fields to mimic and model experiences or phenomena. Simulations can be low-fidelity or high-fidelity, which corresponds to what degree the simulation emulates the real phenomenon. In the medical field, high-fidelity simulations can be effective as a learning tool, where they create a low-stakes environment in which students can apply their knowledge [29]. Engineers rarely have to perform with the same kind of immediate risk associated with their decisions. For engineers, simulations can instead offer new ways to transfer knowledge and concepts.

A subcategory of simulation is *computer simulation*, which can be used, for example, to demonstrate physical phenomena [30]. In this work, simulation and computer simulation will be used interchangeably. Computer simulations allow for learning in an exploratory manner which can aid in comprehending advanced concepts. At the same time, simply having a high-fidelity simulation does not guarantee that students learn from it [30]. Care needs to be taken when planning the session in order for students to actually learn from the computer simulation. For such e-learning meant to inform, there exists principles of what to consider. For example, the multimedia principle states that e-learning courses should include graphics as well as text and the segmenting principle states that continuous lessons should be broken down into smaller chunks [31, Chapter 4, 10]. Application of these principles lead to implementation guidelines such as

- using animations to illustrate abstract ideas,
- using cueing devices such as color to direct attention,
- placing text near corresponding graphic on the screen and
- breaking content down into small topic chunks that can be accessed at the learner's preferred rate [31, Chapter 18].

There is a large body of literature on simulation as a method for learning, aptly called *simulation-based learning* [32]. The literature shows that simulation-based learning can be used as an effective tool for teaching and conveying complex ideas, but as previously stated, consideration is required when creating the simulation.

4.2 Method

To address research questions 3 and 4, an exploratory pilot study was designed to compare two *learning sessions*. A computer simulation (LS1) and a traditional lecture (LS2) serving as the baseline for comparison. The following subsections describes the design for both learning sessions as well as how the data was gathered and evaluated.

4.2.1 Learning Session Design

Initially, a simulation tool was produced in the form a website, with content based on Chapters 2 and 3 using the python library `streamlit`. The website was designed with general e-learning design principles in mind, such as breaking the content down into manageable topic chunks separated on different pages [31, Chapter 18].

In conjunction with this, material for a lecture was produced based on the same content. To ensure andragogical consistency across both learning sessions, the lecture was designed to mirror the website's pacing and structural progression. The main difference between the learning sessions was the interactive element which was not present in the lecture. The development of the lecture still aimed to follow general practices for lesson planning grounded in literature. One such practice is formulating learning outcomes and adjusting the content to the appropriate level of the target audience, in this case participants with an engineering background [26, Chapter 4]. The duration of both learning session was about 40 minutes.

The learning outcomes of these sessions were to give participants a conceptual understanding of

- what a linear quadratic game is,
- what a Nash equilibrium is,
- how linear quadratic games differ from reinforcement learning and
- what linear quadratic games can be used for.

Given that this was a pilot study and that the participation pool was restricted to engineers, the sample size for the study was 21. The computer simulation was carried out for 11 of the engineers and the lecture was held for the remaining 10 engineers.

4.2.2 Questionnaire Design and Evaluation

Data was gathered through two questionnaires, a background questionnaire before the learning session and a questionnaire after the learning session. The questions were originally formulated in Swedish but the translated versions of the questions are presented here.

The background questionnaire aimed to gather general information and provide context about the participants which ensured a fair comparison between the learning

Table 4.1: Questionnaire for participants before attending the learning session.

Question	**Answer type**
1) Estimate how well-versed you are in *game theory*	1:Not at all–5:Very
2) Estimate how well-versed you are in *dynamical systems*	1:Not at all–5:Very
3) Estimate how well-versed you are in *reinforcement learning*	1:Not at all–5:Very
4) What is your preferred method for learning? (For example lectures, laboratory work, self studies, ...)	Text answer

Table 4.2: Questionnaire for participants after attending the learning session.

Question	**Answer type**
1) What is a *linear quadratic game*?	Text answer
2) What is a *strategy* in a game theoretical context?	Text answer
3) What is a *Nash equilibrium*?	Text answer
4) What is the purpose of studying linear quadratic games? What are they suitable/less suitable for?	Text answer
5) Did you feel like you wanted to learn more about the subject?	Yes/No
6) What do you think were the reasons you did/did not feel that way? (Regarding question 5)	Text answer
7) Do you think the format of the lecture is suitable for spreading ideas and inspiring new thoughts? Please explain why.	Text answer
8) How challenging was it to absorb the content during the lecture?	1:Easy–5:Challenging
9) What aspects of the structure facilitated/hindered the absorption of the content?	Text answer
10) What could have made the lecture better?	Text answer

sessions. The questions for the background questionnaire are presented in Table 4.1. After the learning sessions, the participants answered a questionnaire pertaining to both the content of the learning session, and their perceived engagement. The content questions provided an indicator of how well the computer simulation fared in comparison to a traditional lecture in terms of achieving the learning outcomes. The answers to questions 6,7,9 and 10 were open-ended and allowed the participants to share their thoughts and opinions. The questions for this questionnaire are presented in Table 4.2. Questions 1 to 4 in Table 4.2 were used to gauge whether the learning session successfully conveyed the theory at a conceptual level to the participants. Criteria for satisfactory answers for questions 1 to 4 are presented in Table 4.3.

The responses from questions 6,7,9 and 10 in Table 4.2 were analyzed using inductive thematic analysis, meaning that the codes used to generate themes were not decided beforehand [33]. They were instead generated through the coding of responses by

Table 4.3: Criteria for satisfactory answer for learning questions 1 to 4 in questionnaire.

Question	Criteria for correct answer
1	Answer should mention that the dynamics of the system evolves linearly and that the reward function is quadratic.
2	Describes a strategy as some rule for how the players choose their actions.
3	Describes how no players want to change their strategy on their own.
4	Describes any application mentioned during the learning session or some proposed application that is technically viable with what LQ-games can be used for.

both authors independently, and then combined after reaching a consensus. Each response could either generate new codes or potentially refine previous codes. For example, a response "The visual and interactive elements helped in understanding the concepts" could be coded for *visuals* and *interactivity*. One response could therefore be coded for multiple themes. Furthermore, this response contains a keyword *helped* which was interpreted as the visual and interactive elements being positive. Therefore, these codes would be *visuals* $\rightarrow+$ *understanding* and *interactivity* $\rightarrow+$ *understanding*. Another example of coding for the tone is how a response describing the limited content presented as 'introductory' or 'shallow' was coded as positive or negative, respectively. A response could potentially be neutrally coded as well. Explanation of the code notation is found in Appendix A.4.

4.3 Results

In this section, the features of the simulation tool which was produced are described. The identified themes in the answers to the questionnaires as well as the quantitative answers to the questionnaires are presented.

4.3.1 The Simulation Tool

A simulation tool was produced in the form of a multi-page website using the design principles described in Section 4.2. The website contains nine pages, summarized in Table 4.4, all of which either introduces some conceptual theory and/or allows participants to simulate a specific scenario. There is also one page containing all mathematical details.

The platform utilized a combination of text and visuals. This approach was implemented since simulations alone lacked the necessary explanatory context to make the simulation tool useful for learning. By mixing these elements, the design directed the participants to relevant information when they were ready for it. This was meant to facilitate a clear and logical progression throughout the pages. The

simulation tool was also introduced progressively, with additional guidance during the initial appearance of the simulation. Screenshots of the website and simulation environment are in Appendix A.3.

Table 4.4: The structure the website pages and their aim and content.

Page	Content	Aim
Start	Background and purpose of this learning session and basic game theory example	Introduce the website and provide context
One Player	Introduction of the simulator and a simulation of a one-player game	Give instructions on how the simulator is used and a basic introduction to the drone physics
Two Players	Simulation and description of how the Nash equilibrium is algorithmically calculated for two players	Conceptually introduce the algorithm for computing Nash equilibria
Two Players distance	Simulation of drones keeping a fixed distance from each other	Spark interest in what other scenarios can be modeled using LQ games
Swarm	Simulation of swarm behavior	Show the scalability and adaptability of LQ games
Create Custom Game	Simulation of a game modeled by the participant	Spark interest through free exploration
LQ Games In Context	Background and other points of interest concerning LQ games	Spark interest by showing adjacent research questions
Information	Mathematical background	Give concrete mathematical background to those who are interested
Questionnaire	Link to the questionnaire after the learning session	Provide the link to the questionnaire

4.3.2 Answers to the Questionnaires

The answers to the background questionnaire, presented in Figure 4.1, showed that participants generally considered themselves to be familiar with adjacent mathematical topics. It furthermore showed that, by far the most preferred method of learning is through self study. All preferred learning styles are summarized in Table 4.5.

Table 4.5: Participants preferred learning styles from both learning sessions.

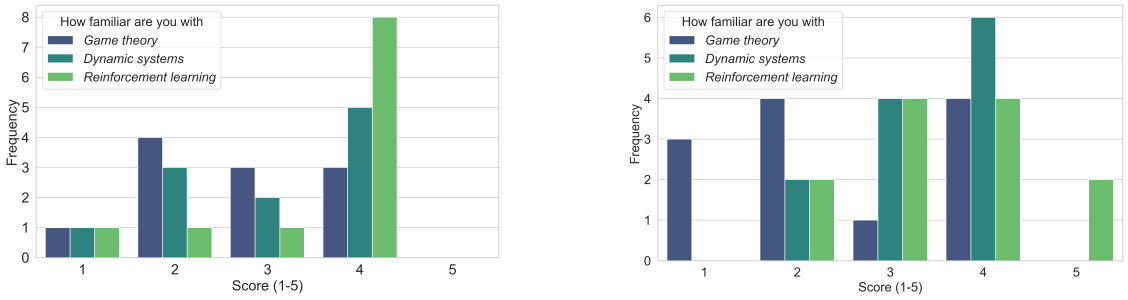
Learning style	From session 1	From session 2
Self studies	7	9
Laboratory work	4	5
Trial & error	3	1
Lecture	2	1
Discussion	1	1
Active learning	1	0
Group work	1	0
Total responses	**11**	**11**

Table 4.6: The number of correct answers for questions 1–4 from both learning sessions.

Question	Correct answers (LS1)	Correct answers (LS2)
Question 1	9	5
Question 2	8	8
Question 3	9	9
Question 4	11	10
Total responses	**11**	**10**

The answers to the questionnaire provided to participants after the learning session, showed that the majority of participants had achieved a conceptual understanding of the material presented. This is seen in the number of acceptable answers to the initial four questions on the questionnaire presented in Table 4.6. The only difference between the learning sessions is the number of correct answers to question 1.

Question 5 showed that only one participant in LS1, and two participants in LS2 were not interested in learning more about the topic. This indicates that the participants, in general, were interested in learning more about the topic. The average response



(a) Learning session 1. **(b)** Learning session 2.

Figure 4.1: Background knowledge questionnaire results for both learning session.

to question 8 was 2.6 in LS1 and 1.7 in LS2, indicating that the difficulty of the presented material was perceived as lower in LS2.

A thematic analysis of answers to questions 6, 7, 9 and 10 was carried out and four main themes pertaining to the effectiveness of the learning sessions were identified. They are presented below in Subsections 4.3.3–4.3.6.

4.3.3 Theme 1: Generating Curiosity and Engagement Through Interactivity and Visuals

The first identified theme has to do with the visual and interactive elements ability to generate curiosity and engagement in participants. A general sentiment expressed by several participants of LS1, particularly to questions 6 and 7, was that the ability to modify parameter values for different scenarios and get visual feedback drove curiosity and increased engagement. This in turn drove curiosity about the scenarios' underlying theory, and in what other contexts it can be applied. Two answers to question 6 illustrate this sentiment:

> "Yes, the format works well. When you change the simulation parameters yourself, it can spark a curiosity about how things work; the curiosity creates reflection, and the reflections lead to new thoughts. However, it got a bit repetitive after a while, which can dampen the inspiration a little". — LS1.2

> "Yes, it was a good mix of theoretical learning and experimentation. The opportunity to set up your own game is also a good way to think independently and explore how things work. If the goal is to give people a taste of the subject and spark their curiosity, then this is a good setup in my eyes". — LS1.4

While LS2 contained no interactive elements, the simulations shown during the lecture were generally perceived as positive. This is exemplified by the following answer to question 6:

> "There was interesting visualization of results in the form of simulations, and interesting applications". — LS2.3

4.3.4 Theme 2: Balance Between Mathematical Rigor and Concepts

The second theme, which was prevalent in answers from both learning sessions, concerns the balance between mathematical rigor and big picture concepts. Several participants in LS2 expressed a positive sentiment about focusing on larger concepts rather than mathematical details. One answer to question 7 illustrates this:

> "Yeah, I think so. I thought there was a good focus on discussing the ideas and keeping the equations to a minimum. In my opinion, we could have shifted the focus even further away from the equations". — LS2.5

Other participants, however, expressed the opposite sentiment and wanted a more detailed and rigorous explanation of the material. They perceived the level of detail as appropriate for inspiration but too broad for effective knowledge sharing:

> "Yes, [good format] to inspire but too shallow for knowledge sharing" — LS2.10.

This sentiment was however less widespread.

4.3.5 Theme 3: The Role of Context

The third theme was identified in answers from both learning sessions and pertains to the role of context in facilitating engagement. Several participants expressed a desire for more concrete, real world examples of how the concepts and theory could be applied. This is illustrated by the following answer to question 10:

> "More suggestions on how it could be developed into something actually 'useful,' perhaps more examples in a military context, something to brainstorm further" — LS2.7.

A few participants in LS1 suggested that the session would have been improved by either being accompanied by a lecture or by an introductory presentation which introduces the subject and the simulation tool. They did not, however, expand on why they believed this would improve the learning session. This sentiment shows that the simulation tool used in LS1 was insufficient in providing context and initial direction for the participants. The lack of direction also connects to Theme 4.

4.3.6 Theme 4: The Balance Between Free Exploration and Direction

The fourth theme concerns opinions on the structure of the learning sessions. A general sentiment expressed by several participants in LS1 was that the ability to freely explore the content at their own pace and in whatever order they chose, was conducive to curiosity and engagement. Another sentiment is that the lack of direction hindered learning as it was difficult to grasp what information was important and approximately how much time they were supposed to spend on each page. This dual nature of free exploration is illustrated by two answers to questions 9 and question 7 respectively:

> "Facilitated: Structured content on the page, simple and basic explanations. The ability to view it in multiple steps, and to run the simulations yourself to learn the meaning of the parameters. The opportunity to go through the material at your own pace" — LS1.2,

> "The setup was fun! It was possible to take it at your own pace, but that made it a bit difficult to determine what was most important. Without a time limit, it feels like a good way to learn the concept; in this case, however, it became a bit difficult to decide when to move on in the hope of understanding it later" — LS1.7.

That the simulation tool combined simulations with explanatory text was considered positive by participants of LS1. This is illustrated by the following answer to question 9:

> "It was a good setup with a mix of an interactive format and informative text. I feel that the interactive format makes you more interested in reading the text itself than if it were standalone, so you absorb more of the information" — LS1.10.

4.4 Discussion

This discussion aims to explore the extent to which simulation-based learning can serve as an effective method of knowledge sharing for engineers. The type of knowledge sharing that we consider are semi-formal to formal forms of knowledge sharing in which a person presents a topic to other employees, which lies outside of their immediate professional scope. This can range from formal lectures to informal show and tells, and can act as a form of professional development or as an informal forum for discussion.

In regards to the effectiveness of the simulation-based approach on comprehension of the topic, the results show that the proportion of correct answers to questions 1–4 was high for both learning sessions. There was however no significant difference between LS1 and LS2. In both learning sessions participants generally reported being familiar with adjacent topics which likely created a ceiling effect, potentially masking the andragogical effects of the simulation tool. The way in which questions 1–4 were formulated only test lower-order thinking skills, specifically Level 1 (Remembering) and Level 2 (Understanding), of Blooms taxonomy. In a knowledge sharing context, the ability of participants to simply recall information may however not be of interest. Rather, activating higher-order cognitive functions such as analyzing or evaluating the information may provide more utility in generating new ideas and innovation. This misalignment render questions 1–4 unable to meaningfully differentiate which effect on conceptual understanding the two learning approaches had. While it wasn't tested, it can be argued that the simulation-based approach to a greater extent activates higher-order cognitive functions as it is more closely aligned with experiential learning. This general idea was also echoed by participants in Theme 1.

The themes identified in the open ended questions are primarily centered around the participants perceived level of engagement during each learning session. They also provide an indication of which elements are important for generating engagement for engineers. Theme 1 strongly indicates that interactivity and visual feedback is highly conducive to sparking curiosity and engagement. Participants of LS2 generally also found the learning session to be engaging. Their engagement however, was reported as being mainly due to personal or professional interest in the topic. This result, by itself, suggest that while both methods are capable of generating engagement, the intrinsic qualities of a simulation tool – interactivity and visuals – may be able to generate engagement without having to rely on the participants pre-existing

motivation.

Theme 2 indicates that the participants, to a greater extent, prefer being presented with big picture concepts and ideas rather than equation-dense mathematical content. This may be surprising since it contrasts our perception that engineers inherently favor mathematical detail. It is however in line with Theme 3 in which the level of applicability and context surrounding the content is highlighted as important for engagement which is consistent with the theory of andragogy. Themes 2 and 3 are highly relevant not only in the context of simulation-based learning but also knowledge sharing in general, particularly in a cross-functional setting. For specialized engineers, a complete understanding of a topic typically requires an understanding of complex, technical details, to which the cognitive barrier of understanding is high. In this regard, a simulation tool may work as a boundary object by transferring a common conceptual understanding independent of technical details. However, if the recipient of the information is not familiar with the topic it is important to give them necessary direction about where to focus their attention, as indicated by Theme 4.

It is relevant to also discuss the operational feasibility of using simulation tools as a mechanism for knowledge sharing. One possible benefit of simulation tools is their ability to share knowledge with minimal coordination or planning. A formal lecture usually requires significant planning and requires employees to take time out of their schedule, while a simulation tool can be employed any time. We can, however, not draw any conclusions about the efficacy of this as the study was conducted in a controlled environment and it is not unreasonable to believe that the result may have been different in another setting. Another thing to consider is that simulation does not lend itself for specific topics. This, coupled with the relative difficulty and work required to produce a simulation tool, suggests that choosing simulation-based learning should be done with care.

5

Conclusion

This thesis sought to derive Nash equilibria (NE) for linear quadratic (LQ) games and evaluate their utility as a robust baseline for benchmarking novel multi-agent reinforcement learning (MARL) algorithms. While exact NE can be computed for specific LQ-game formulations using the Riccati-based algorithm introduced in Chapter 2, their practical utility as an evaluation baseline proves insufficient in many cases. Comparing NE-policies and MARL-policies in two coordination scenarios showed that they on average achieved equivalent performance. In a coupled scenario, the MARL-policies frequently outperformed the NE-policies by utilizing a joint policy which was more cooperatively efficient than the NE-policies. This result shows that using an exact NE to benchmark cooperative tasks may obscure more efficient cooperative policies. Some broader structural limitations of LQ-games, as defined in this thesis, are the restrictive assumptions they impose, such as continuous action space and full observability. These assumptions do not always reflect the complex environments where MARL is usually deployed, rendering LQ-games limited as a comprehensive benchmarking tool. To address these limitations, future work should investigate the convergence and solvability of the Riccati iteration algorithm under relaxed assumptions. Additionally, extending the comparisons to other MARL-policies, as well as evaluating more complex scenarios would provide a better understanding of when NE-policies are relevant to use as a baseline for evaluating MARL-algorithms.

This thesis also sought to investigate the effectiveness of simulation-based learning as a form of knowledge sharing for engineers as well as gather their general attitudes towards simulation-based learning. Two lectures were held, one traditional didactic lecture, and one using a simulation tool, produced specifically for this purpose. Answers to a questionnaire, provided to participants of both lectures, indicated that the simulation tool was perceived as highly engaging due to the interactive and visual elements present as well as the ability to freely explore the material. The answers also indicated that the participants considered sufficient context and examples of applicability as important aspects to include when designing learning sessions in the context of knowledge sharing for engineers. The answers indicated that the effectiveness of simulation on learning was comparable to that of traditional lectures. There is however reason to believe that simulation is more conducive to higher order cognitive function than the traditional lecture due to its interactive and exploratory nature. In conclusion, these results indicate that, if it is feasible and appropriate to implement, a simulation-based approach is an engaging and suitable method for knowledge sharing for engineers. Further research is needed to strengthen the link between simulation-based learning and its effect on innovation.

Bibliography

[1] L. Gupta, R. Jain, and G. Vaszkun, "Survey of important issues in uav communication networks," *IEEE communications surveys & tutorials*, vol. 18, no. 2, pp. 1123–1152, 2015.

[2] Y. Bu, Y. Yan, and Y. Yang, "Advancement challenges in uav swarm formation control: A comprehensive review," *Drones*, vol. 8, no. 7, p. 320, 2024.

[3] S. V. Albrecht, F. Christianos, and L. Schäfer, *Multi-Agent Reinforcement Learning: Foundations and Modern Approaches*. MIT Press, 2024.

[4] K. Zhang, Z. Yang, and T. Basar, "Policy optimization provably converges to nash equilibria in zero-sum linear quadratic games," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[5] E. Mazumdar, L. J. Ratliff, M. I. Jordan, and S. S. Sastry, *Policy-gradient algorithms have no guarantees of convergence in linear quadratic games*, 2019. arXiv: 1907.03712 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1907.03712.

[6] Z. Wang and N. Wang, "Knowledge sharing, innovation and firm performance," *Expert Systems with Applications*, vol. 39, no. 10, pp. 8899–8908, 2012, ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2012.02.017.

[7] S. Harkema, "A complex adaptive perspective on learning within innovation projects," *The Learning Organization: An International Journal*, vol. 10, no. 6, pp. 340–346, Dec. 2003, ISSN: 0969-6474. DOI: 10.1108/09696470310497177.

[8] U. Zander and B. Kogut, "Knowledge and the speed of the transfer and imitation of organizational capabilities: An empirical test," *Organization Science*, vol. 6, no. 1, pp. 76–92, 1995. DOI: 10.1287/orsc.6.1.76.

[9] X. Fan, D. Geelan, and R. Gillies, "Evaluating a novel instructional sequence for conceptual change in physics using interactive simulations," *Education Sciences*, vol. 8, no. 1, 2018, ISSN: 2227-7102. DOI: 10.3390/educsci8010029. [Online]. Available: https://www.mdpi.com/2227-7102/8/1/29.

[10] E. Solan, *A Course in Stochastic Game Theory*. Cambridge: Cambridge University Press, 2022.

[11] T. Başar and G. J. Olsder, *Dynamic Noncooperative Game Theory* (Mathematics in Science and Engineering). New York, NY: Academic Press, 1982, vol. 160, pp. iii–viii, 1–430, ISBN: 978-0-12-080220-3.

[12] P. Van Dooren, "A generalized eigenvalue approach for solving riccati equations," *SIAM Journal on Scientific and Statistical Computing*, vol. 2, no. 2, pp. 121–135, 1981. DOI: 10.1137/0902010.

[13] Jeffrey Armstrong and Chad Fulton and Ilhan Polat, *SciPy: Fundamental Library for Scientific Computing in Python*, https://docs.scipy.org/doc/scipy/reference/generated/scipy.linalg.solve_discrete_are.html, Accessed: 2026-03-11, 2016.

[14] B. Nortmann, A. Monti, M. Sassano, and T. Mylvaganam, "Nash equilibria for linear quadratic discrete-time dynamic games via iterative and data-driven algorithms," *IEEE Transactions on Automatic Control*, vol. 69, no. 10, pp. 6561–6575, 2024. DOI: 10.1109/TAC.2024.3375249.

[15] G. Salizzoni, R. Ouhamma, and M. Kamgarpour, "Nash equilibria in scalar discrete-time linear quadratic games," *arXiv preprint arXiv:2410.12544*, 2025. DOI: 10.48550/arXiv.2410.12544.

[16] M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning," in *Machine learning proceedings 1994*, Elsevier, 1994, pp. 157–163.

[17] C. Yu et al., "The surprising effectiveness of ppo in cooperative multi-agent games," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24 611–24 624, 2022.

[18] S. R. Dubey, S. K. Singh, and B. B. Chaudhuri, "Activation functions in deep learning: A comprehensive survey and benchmark," *Neurocomputing*, vol. 503, pp. 92–108, 2022, ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2022.06.111. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231222008426.

[19] W. Geng, B. Xiao, R. Li, N. Wei, D. Wang, and Z. Zhao, "Noise distribution decomposition based multi-agent distributional reinforcement learning," *IEEE Transactions on Mobile Computing*, vol. 24, pp. 2301–2314, 2025. DOI: 10.1109/tmc.2024.3492272.

[20] R. E. Miles, C. C. Snow, A. D. Meyer, and H. J. Coleman, "Organizational strategy, structure, and process," *The Academy of Management Review*, vol. 3, no. 3, pp. 546–562, 1978. DOI: 10.2307/257544.

[21] I. Nonaka and H. Takeuchi, *The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation*. New York: Oxford University Press, 1995.

[22] M. Caccamo, D. Pittino, and F. Tell, "Boundary objects, knowledge integration, and innovation management: A systematic review of the literature," *Technovation*, vol. 122, p. 102 645, 2023, ISSN: 0166-4972. DOI: https://doi.org/10.1016/j.technovation.2022.102645. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0166497222001961.

[23] S. L. Star and J. R. Griesemer, "Institutional ecology, 'translations' and boundary objects: Amateurs and professionals in berkeley's museum of vertebrate zoology, 1907-39," *Social Studies of Science*, vol. 19, no. 3, pp. 387–420, 1989. DOI: 10.1177/030631289019003001.

[24] B. S. Bloom, M. D. Engelhart, E. J. Furst, W. H. Hill, and D. R. Krathwohl, *Taxonomy of Educational Objectives: The Classification of Educational Goals. Handbook I: Cognitive Domain*. New York, NY: Longmans, Green and Co., 1956.

[25] L. W. Anderson et al., *A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives*. New York, NY: Longman, 2001.

[26] R. M. Felder and R. Brent, *Teaching and Learning STEM: A Practical Guide*, 2nd. John Wiley & Sons, Incorporated, 2024, ISBN: 9781394196357.

[27] M. S. Knowles, *The Modern Practice of Adult Education: Andragogy versus Pedagogy*. New York: Association Press, 1970.

[28] D. Kolb, *Experiential Learning: Experience As The Source Of Learning And Development*. Prentice Hall, Jan. 1984, vol. 1, ISBN: 0132952610.

[29] M. Kharasch, P. Aitchison, and S. Leikin, "Simulation applications in emergency medical services," *Disease-a-Month*, 2011. DOI: 10.1016/J.DISAMONTH.2011.08.012.

[30] T. de Jong and W. R. van Joolingen, "Scientific discovery learning with computer simulations of conceptual domains," *Review of Educational Research*, vol. 68, pp. 179–201, 1998. [Online]. Available: https://api.semanticscholar.org/CorpusID:53410122.

[31] R. C. Clark and R. E. Mayer, *E-learning and the Science of Instruction: Proven Guidelines for Consumers and Designers of Multimedia Learning*, 4th. Hoboken, New Jersey: John Wiley & Sons, Inc., 2016, Revised edition of the 2011 version, ISBN: 978-1119158660.

[32] S. Asadi, J. Allison, M. Khurana, and M. Nilashi, "Simulation-based learning for computer and networking teaching: A systematic literature review and bibliometric analysis," *Education and Information Technologies*, vol. 29, pp. 15 655–15 690, 2024. DOI: 10.1007/s10639-024-12476-7.

[33] V. Braun and V. Clarke, "Using thematic analysis in psychology," *Qualitative Research in Psychology*, vol. 3, no. 2, pp. 77–101, 2006. DOI: 10.1191/1478088706qp063oa.

[34] A. Hansson and P. Hagander, "How to solve singular discrete-time riccati-equations," English, 13th IFAC World Congress ; Conference date: 30-06-1996 Through 05-07-1996, 1996.

A

Appendix

A.1 Solutions to the Discrete Algebraic Riccati Equations

We wish to say something about the solutions to the discrete algebraic Riccati equation (DARE) (2.18). If the DARE is of the form

$$P = Q + A^\top PA - A^\top PB(R + B^\top PB)^{-1}B^\top PA, \tag{A.1}$$

then if (A, B) is a stabilizable pair, $Q \preceq 0$, $R \prec 0$ where Q, R are symmetric, then there exists a symmetric matrix $P \prec 0$ which solves Equation (A.1) [34, Theorem 1]. One way to numerically compute P is by forming the following so-called matrix pencil

$$H - \lambda J = \begin{bmatrix} A & 0 & B \\ -Q & I & 0 \\ 0 & 0 & R \end{bmatrix} - \lambda \begin{bmatrix} I & 0 & 0 \\ 0 & A^\top & 0 \\ 0 & -B^\top & 0 \end{bmatrix} \tag{A.2}$$

and using generalized Schur decomposition to construct P, as described in [12, Equation (58)]. When comparing (A.1) and (2.18), we see that they are the same if we make the substitution $A = \tilde{A}$ and $R = \tilde{R}$.

A.2 Derivation of Newtonian Motion

We can derive the time discrete Newtonian motion by integrating the continuous case over the discretized interval. Assume we have the acceleration $a(t)$ at some time $t \in \mathbb{R}$, and assume it is constant in the interval $[t, t + h]$ for a time step $h \in \mathbb{R}$. Then we get the velocity in the next time step as

$$v(t + h) = v(t) + \int_t^{t+h} a(\tau)\mathrm{d}\tau, \tag{A.3}$$

where $v(t)$ is known. Since $a(t)$ is assumed to be constant during this interval, the equation becomes

$$v(t + h) = v(t) + ha(t). \tag{A.4}$$

The equation for the position is derived similarly as

$$p(t+h) = p(t) + \int_t^{t+h} v(\tau)\mathrm{d}\tau. \tag{A.5}$$

Using Equation (A.4) we get that

$$p(t+h) = p(t) + \int_t^{t+h} v(t) + (\tau - t)a(t)\mathrm{d}\tau, \tag{A.6}$$

since $h = \tau - t$ when τ goes from t to $t+h$. With Equation (A.6) and using the substitution $s = \tau - t$ when integrating the acceleration term we get that

$$p(t+h) = p(t) + hv(t) + a(t)\int_0^h s\mathrm{d}s, \tag{A.7}$$

$$= p(t) + hv(t) + \frac{h^2}{2}a(t). \tag{A.8}$$

Figure A.1: Landing page for the simulation website described in Chapter 4.

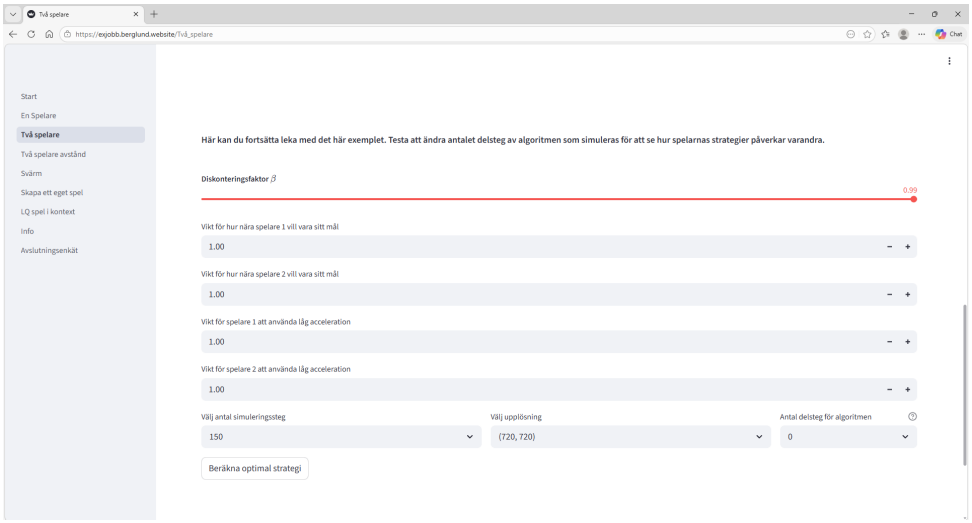


Figure A.2: Simulation settings for a two-player game.

A.3 Screenshots of the Website

A.4 Codes for Thematic Analysis

Symbol Key

- → Leads to
- + Increases / positive impact
- − Decreases / negative impact
- ? Uncertain relationship

- **Theme 1: Generating Curiosity and Engagement Through Interactivity and Visuals**
  - Freedom to explore → + Curiosity
  - Experimentation → + Curiosity
  - Experimentation → + Personal interest
  - Visuals → + Curiosity
  - Interactivity → + Curiosity
  - Interactivity → + Personal interest
  - Prior knowledge → ? Curiosity
- **Theme 2: Balance Between Mathematical Rigor and Concepts**
  - Too many equations → + Boredom
  - Low level of detail in theory → + Focus
  - Low level of detail in theory → + Confusion
  - Lecture → Good for inspiration, shallow in knowledge sharing
- **Theme 3: The Role of Context**
  - Lack of applicability → − Curiosity
  - Lack of context for simulation → +Confusion
  - Simulation as a complement to lecture
- **Theme 4: The Balance Between Free Exploration and Direction**
  - Freedom to explore: Good with structure
  - Progression → + Understanding
  - Repetitive or low progression → + Boredom and decreased inspiration

DEPARTMENT OF COMMUNICATION AND LEARNING IN SCIENCE
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY