Is Cooperation the Key? Untersuchung kooperativer und individueller Belohnungsstrukturen für NPC-Verhalten im Multi-Agent Reinforcement Learning

BACHELORARBEIT

Methode

Experimenteller Vergleich / Simulation

Bachelorstudium

„Digital Business & Software Engineering“

MCI | DIE UNTERNEHMERISCHE HOCHSCHULE®

Betreuer*in

Prof. Dr. Michael Kohlegger

Verfasser*in

Manuel Junker

Jahrgang 2023

52312278

16. Mai 2026

Zusammenfassung

Diese Bachelorarbeit untersucht empirisch, welche Auswirkungen unterschiedliche Belohnungsstrukturen auf das emergente Verhalten autonomer Agenten in einem dezentralen Multi-Agent-Reinforcement-Learning-System haben. Im Mittelpunkt steht die Frage, ob der Einsatz gemeinsamer Teambelohnungen dazu führt, dass computergesteuerte Agenten messbar effizientere Strategien entwickeln als bei individuellem, egoistischem Training. Als experimentelle Grundlage diente eine modifizierte Push-Block-Umgebung in der Unity-ML-Agents-Plattform, in der drei Agenten simultan physische Objekte unterschiedlicher Masse in eine Zielzone befördern mussten. Es wurden drei Trainingsläufe unter identischen algorithmischen Bedingungen verglichen: eine kooperative Konfiguration mit geteilter Teambelohnung ohne Aufgabenprogression, eine kooperative Konfiguration mit geteilter Teambelohnung unter Einsatz von Curriculum Learning sowie eine kompetitive Konfiguration mit individueller Belohnung und identischer Aufgabenprogression. Als Trainingsalgorithmus kam Independent Proximal Policy Optimization in dezentraler Anwendung zum Einsatz. Die quantitative Auswertung der Trainingsmetriken sowie eine qualitative Verhaltensanalyse der trainierten Modelle ergaben ein kontraintuitives Ergebnis: Im durchgeführten Experiment erzielte die egoistische Belohnungsstruktur die höchste Lerneffektivität und die schnellste Netzwerkkonvergenz, während die unstrukturierte Teambelohnung am Credit Assignment Problem zu scheitern schien und passives Lazy-Agent-Verhalten hervorrief. Der kooperative Ansatz mit Curriculum Learning schnitt trotz strukturierter Aufgabenprogression schlechter ab als die unstrukturierte Team-Baseline, was als möglicher Curriculum-Overfitting-Effekt interpretiert werden kann. Besonders relevant ist der Befund, dass die egoistische Belohnungsstruktur bei physikalisch anspruchsvollen Objekten emergente Kooperation aus purem Eigennutz hervorbrachte. Die Ergebnisse legen nahe, dass in dezentralen Architekturen ohne zentralen Critic eine scharfe individuelle Belohnungsstruktur einer unstrukturierten Teambelohnung überlegen ist, sofern die resultierende Ressourcenkonkurrenz akzeptiert wird. Der Befund, dass physikalische Umgebungsrestriktionen emergente Kooperation zuverlässiger erzwingen können als geteilte Teambelohnungen, hat unmittelbare Implikationen für das Design zukünftiger Multi-Agenten-Systeme.

Abstract

This bachelor's thesis empirically investigates the effects of different reward structures on the emergent behaviour of autonomous agents in a decentralised multi-agent reinforcement learning system. The central question is whether shared team rewards lead to measurably more efficient strategies compared to individual, egocentric training. The experimental foundation was a modified Push-Block environment in Unity ML-Agents, in which three agents simultaneously transported physical objects of varying mass into a target zone. Three training configurations were compared: a cooperative configuration with shared team reward and no task progression, a cooperative configuration with shared team reward and curriculum learning, and a competitive configuration with individual reward and an identical curriculum. Independent Proximal Policy Optimization was applied as the training algorithm. Quantitative and qualitative analysis of the trained models yielded a counterintuitive result: within the conducted experiment, the egocentric reward structure achieved the highest learning effectiveness and fastest network convergence, while the unstructured team reward appeared to fail due to the credit assignment problem, inducing passive lazy-agent behaviour. Despite structured task progression, the cooperative curriculum approach performed worse than the unstructured team baseline, suggesting a curriculum overfitting effect. Notably, the egocentric reward structure brought about emergent cooperation from pure self-interest when agents encountered physically demanding objects. The results suggest that in decentralised architectures without a centralised critic, a sharp individual reward structure outperforms an unstructured shared reward, provided resource competition is accepted as a trade-off. The finding that physical environmental constraints can enforce emergent cooperation more reliably than shared team rewards has direct implications for the design of future multi-agent systems.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Die Gestaltung von Belohnungssystemen zählt zu den zentralen Designentscheidungen im Multi-Agent Reinforcement Learning. Sie bestimmt maßgeblich, ob autonome Agenten kooperieren, konkurrieren oder dysfunktionales Verhalten entwickeln. Diese Arbeit untersucht, wie die Wahl der Belohnungsstruktur das emergente Verhalten autonomer Agenten in dezentralen Multi-Agent-Reinforcement-Learning-Systemen beeinflusst. Ausgehend von der Problemstellung und der zentralen Forschungsfrage wird ein empirischer Vergleich dreier Trainingsansätze in einer physikbasierten Simulationsumgebung durchgeführt.

1.1 Motivation und Problemstellung

Die Fähigkeit mehrerer autonomer Agenten, in einer gemeinsamen Umgebung koordiniert zu handeln, zählt zu den anspruchsvollsten und gleichzeitig folgenreichsten Problemen der modernen Informatik. Während klassische Ansätze zur Steuerung von *Non-Player Characters* (NPCs), also computergesteuerten Akteuren in simulierten Umgebungen, auf regelbasierten Systemen und handcodierten Entscheidungsbäumen beruhen, eröffnet das *Reinforcement Learning* (RL) einen grundlegend anderen Weg: Agenten erlernen eigenständig Verhaltensstrategien durch wiederholte Interaktion mit ihrer Umgebung, ohne dass das Zielverhalten explizit vorgeschrieben werden muss [1]. Die wissenschaftliche und öffentliche Aufmerksamkeit für diesen Ansatz ist spätestens seit den spektakulären Erfolgen von Systemen wie AlphaGo [2] und AlphaZero [3] dauerhaft gestiegen, die menschliche Expert*innen in komplexen strategischen Spielen übertrafen.

Diese Erfolge beschränken sich nicht auf Einzelspielerszenarien. So wurde etwa im komplexen Echtzeit-Strategiespiel *StarCraft II* Grandmaster-Niveau durch ein MARL-basiertes System erreicht, das hochgradig koordiniertes Mehrfachagenten-Verhalten in einer partiell beobachtbaren Umgebung demonstrierte [4]. Ein ähnliches Ergebnis wurde für das Mehrspielerspiel *Dota 2* erzielt, in dem komplexe Teamstrategien emergent aus dem Trainingsprozess hervorgingen [5]. Diese Systeme belegen eindrücklich, dass MARL in der Lage ist, koordiniertes Gruppenverhalten zu erzeugen, das weder explizit programmiert noch manuell annotiert wurde.

Die wissenschaftliche Herausforderung bei der Übertragung dieser Erfolge auf allgemeinere Anwendungsszenarien liegt jedoch in einer Grundspannung, die jedes kooperative

MARL-System durchzieht: der Wahl der Belohnungsstruktur. Wird eine gemeinsame Teambelohnung (*Shared Reward*) eingesetzt, die alle Agenten gleichermaßen für den kollektiven Erfolg entlohnt, entsteht das *Credit Assignment Problem*, also die algorithmische Schwierigkeit, den individuellen Beitrag eines einzelnen Agenten an einem gemeinsamen Ergebnis zu quantifizieren [6]. Das Belohnungssignal wird durch das Rauschen der kollektiven Gruppenleistung verwässert, sodass individuelle Handlungen und ihre Konsequenzen nicht zuverlässig miteinander verknüpft werden. Eine individuelle Belohnungsstruktur löst dieses Zuordnungsproblem auf, indem sie jeden Agenten ausschließlich für den eigenen Beitrag entlohnt. Dies kann jedoch kompetitives Verhalten und Ressourcenkonkurrenz zwischen den Agenten hervorrufen, die einer effizienten Aufgabenlösung entgegenwirken.

Der vorliegenden Arbeit liegt die Beobachtung zugrunde, dass diese Spannung zwischen kooperativen und individuellen Belohnungsstrukturen in der Literatur zwar theoretisch umfassend diskutiert, jedoch im Kontext leichtgewichtiger, reproduzierbarer Simulationsumgebungen selten systematisch und isoliert empirisch untersucht wird. Viele prominente Arbeiten in diesem Bereich operieren auf hochskalierten, rechenintensiven Umgebungen, die den experimentellen Zugang für unabhängige Reproduktionen erheblich erschweren [4], [5]. Die vorliegende Arbeit begegnet diesem Desiderat durch den Einsatz der Unity-ML-Agents-Plattform als zugängliche und reproduzierbare Experimentalbasis, auf der die Auswirkungen der Belohnungsstruktur unter kontrollierten Bedingungen isoliert untersucht werden können.

1.2 Forschungsfrage und Zielsetzung

Aus der beschriebenen Problemstellung ergibt sich die folgende zentrale Forschungsfrage, die den Rahmen dieser Arbeit definiert:

> *Führt der Einsatz von gemeinsamen Team-Belohnungen (Shared Rewards) dazu, dass NPCs messbar effizientere Strategien entwickeln und eine höhere Erfolgsrate erzielen als bei individuellem Training?*

Zur Beantwortung dieser Frage wird ein vergleichendes Experiment in einer modifizierten *Push-Block*-Umgebung durchgeführt, in der drei simultan agierende Agenten physische Objekte unterschiedlicher Masse in eine Zielzone befördern müssen. Das Experiment kontrastiert dabei drei klar abgegrenzte Trainingsansätze unter sonst identischen Bedin-

gungen: eine kooperative Konfiguration mit geteilter Teambelohnung ohne strukturierte Aufgabenprogression (*Team-Baseline*), eine kooperative Konfiguration mit geteilter Teambelohnung unter Einsatz von *Curriculum Learning* (*Team-Curriculum*) sowie eine kompetitive Konfiguration mit individueller Belohnung und identischer Aufgabenprogression (*Ego-Ansatz*). Als Trainingsalgorithmus kommt *Independent Proximal Policy Optimization* (IPPO) zum Einsatz, das heißt PPO in dezentraler Anwendung auf mehrere unabhängig lernende Agenten [7]. Diese Architektur wurde bewusst gewählt, da sie die algorithmischen Auswirkungen der Belohnungsstruktur isoliert erfahrbar macht, ohne dass ein zentraler Critic-Mechanismus das Zuordnungsproblem während des Trainings algorithmisch teilweise kompensiert.

Die Zielsetzung der Arbeit ist damit dreifach: Erstens soll empirisch untersucht werden, welche der drei Trainingskonfigurationen in der gewählten Umgebung die höchste Lerneffektivität, gemessen am akkumulierten extrinsischen Reward, sowie die größte Aufgabeneffizienz, gemessen an der durchschnittlichen Episodenlänge, erzielt. Zweitens soll das qualitative Verhalten der trainierten Modelle analysiert werden, um emergente Verhaltensmuster, darunter unerwartete Kooperationsformen oder dysfunktionale Wettbewerbsdynamiken, zu identifizieren und zu interpretieren. Drittens sollen die Ergebnisse in den bestehenden wissenschaftlichen Diskurs zu kooperativem und kompetitivem MARL eingebettet werden, um die Generalisierbarkeit der Befunde und konkrete Anknüpfungspunkte für weiterführende Forschungen zu benennen.

Es wird dabei explizit keine Aussage über die absolute Überlegenheit eines Belohnungsparadigmas in beliebigen Umgebungen angestrebt. Die Ergebnisse sind an die spezifischen Eigenschaften der gewählten Simulationsumgebung gebunden und in diesem Kontext zu interpretieren. Dazu zählen die physikalische Massedifferenzierung der Blöcke, die partielle Beobachtbarkeit durch Raycast-Sensoren sowie die dezentrale IPPO-Architektur.

1.3 Aufbau der Arbeit

Die Arbeit gliedert sich in sechs Kapitel, die aufeinander aufbauen und gemeinsam einen vollständigen empirischen Forschungszyklus abbilden. Das vorliegende erste Kapitel hat die Problemstellung motiviert, die Forschungsfrage präzisiert und den Rahmen der Zielsetzung abgesteckt.

Kapitel 2 legt die theoretischen Grundlagen dar, die für das Verständnis des Experiments und seiner Ergebnisse erforderlich sind. Es führt in die Grundprinzipien des Reinforcement Learning ein, erläutert die spezifischen Herausforderungen des Multi-Agent Reinforcement Learning, beschreibt den eingesetzten PPO-Algorithmus in seinen konzeptionellen Grundzügen und ordnet die Dichotomie kooperativer und kompetitiver Belohnungsparadigmen in den wissenschaftlichen Diskurs ein.

Kapitel 3 beschreibt den Versuchsaufbau in seiner vollen technischen Tiefe. Es dokumentiert die Simulationsumgebung, das modifizierte Push-Block-Szenario mit seinen Wahrnehmungs- und Handlungsräumen, die konkrete Gestaltung der drei Belohnungsfunktionen sowie die Implementierung des Curriculum-Learning-Ansatzes. Abschließend werden die Trainingskonfiguration und die verwendete Hardware spezifiziert.

Kapitel 4 präsentiert die Resultate der drei Trainingsläufe. Die quantitative Auswertung umfasst die Metriken extrinsischer Reward, Episodenlänge und Policy-Entropie; ergänzend erfolgt eine qualitative Verhaltensanalyse der trainierten Modelle in der Simulationsumgebung.

Kapitel 5 interpretiert die Ergebnisse vor dem Hintergrund der Forschungsfrage und der einschlägigen Literatur, diskutiert die methodischen Limitationen des Versuchsaufbaus und ordnet die zentralen Befunde in den wissenschaftlichen Diskurs ein.

Kapitel 6 fasst die Erkenntnisse der Arbeit zusammen, beantwortet die Forschungsfrage komprimiert und leitet daraus konkrete Empfehlungen für zukünftige Forschungsarbeiten ab.

2 Theoretischer Hintergrund

Die Frage, ob kooperative Belohnungsstrukturen zu effizienteren Agenten führen als individuelle, lässt sich nicht isoliert von den algorithmischen und formalen Grundlagen beurteilen, auf denen das Experiment aufbaut. Zunächst werden die lerntheoretischen Prinzipien des *Reinforcement Learning* eingeführt, da sie das Fundament für alle weiteren Konzepte bilden. Darauf aufbauend werden die spezifischen Herausforderungen des *Multi-Agent Reinforcement Learning* herausgearbeitet, insbesondere jene, die unmittelbar mit der Wahl der Belohnungsstruktur zusammenhängen. Anschließend wird der eingesetzte

Proximal Policy Optimization-Algorithmus (PPO) in seinen konzeptionellen Grundzügen beschrieben. Den Abschluss bildet eine Einordnung der Dichotomie kooperativer und kompetitiver Belohnungsparadigmen, die den konzeptionellen Kern der Forschungsfrage bildet.

2.1 Grundlagen des Reinforcement Learning (RL)

Reinforcement Learning bezeichnet einen Teilbereich des maschinellen Lernens, in dem ein autonomer Agent durch wiederholte Interaktion mit einer Umgebung eine optimale Verhaltensstrategie erlernt [1]. Im Gegensatz zu überwachten Lernverfahren, bei denen korrekte Ein- und Ausgabepaare als Trainingsgrundlage vorgegeben werden, erhält der Agent ausschließlich ein skalares Belohnungssignal (*Reward*), anhand dessen er die Güte seiner ausgeführten Aktionen beurteilen kann. Das übergeordnete Lernziel besteht darin, eine Policy zu erlernen, die die kumulative, langfristige Belohnung über die gesamte Interaktionsdauer maximiert [8].

Die formale Grundlage des RL bildet das Framework des *Markov-Entscheidungsprozesses* (*Markov Decision Process*, MDP), das die Umgebung als ein Tupel (S, A, P, R, γ) beschreibt [1]. Hierbei bezeichnet S die Menge aller möglichen Umgebungszustände, A die Menge aller verfügbaren Aktionen, $P : S \times A \times S \to [0, 1]$ die stochastische Übergangswahrscheinlichkeit zwischen Zuständen und $R : S \times A \to \mathbb{R}$ die Belohnungsfunktion. Der *Diskontierungsfaktor* $\gamma \in [0, 1]$ gewichtet zukünftige Belohnungen relativ zu unmittelbaren Belohnungen und spiegelt damit den zeitlichen Planungshorizont des Agenten wider. Die mathematische Grundlage für die rekursive Berechnung optimaler Wertfunktionen bildet das sogenannte *Optimalitätsprinzip* [9]. Dieses besagt, dass eine optimale Policy die fundamentale Eigenschaft besitzt, dass alle verbleibenden Entscheidungen, unabhängig vom Ausgangszustand und der ersten getroffenen Entscheidung, eine optimale Policy für den Zustand bilden müssen, der aus dieser ersten Entscheidung resultiert. Die Pfadoptimierung lässt sich somit in eine Abfolge rekursiver Teilprobleme zerlegen, bei der die optimale Entscheidung für den Rest eines Problems unabhängig von den Entscheidungen vor Erreichen des aktuellen Zustands ist.

Die erlernte Verhaltensstrategie des Agenten wird formal durch seine *Policy* $\pi : S \to A$ beschrieben, die jedem wahrgenommenen Zustand entweder eine deterministische Aktion

oder eine Wahrscheinlichkeitsverteilung über mögliche Aktionen zuordnet. Zur Bewertung der Qualität einer Policy werden *Wertfunktionen* herangezogen: Die *Zustandswertfunktion* $V^{\pi}(s)$ quantifiziert den erwarteten kumulierten Reward, der erzielt wird, wenn der Agent im Zustand s gemäß Policy π agiert, während die *Aktionswertfunktion* $Q^{\pi}(s, a)$ zusätzlich die im Zustand s gewählte Aktion a in die Bewertung einbezieht [10]. Auf Basis dieser Funktionen ermöglicht das klassische *Q-Learning*-Verfahren die modellfreie Approximation einer optimalen Policy aus gesammelten Erfahrungsdaten, ohne eine explizite Repräsentation der Umgebungsdynamik vorauszusetzen [10].

Mit zunehmender Dimension der Zustands- und Aktionsräume stoßen tabellarische Methoden, die jeden Zustand diskret repräsentieren, an ihre praktischen Grenzen. Der Übergang zu *Deep Reinforcement Learning* (DRL) adressiert dieses Problem, indem tiefe neuronale Netze als universelle Funktionsapproximatoren für Wert- und Policyfunktionen eingesetzt werden [11]. Den wegweisenden Nachweis für die Leistungsfähigkeit dieses Ansatzes lieferte der *Deep Q-Network*-Algorithmus (DQN), der auf einer Menge von Atari-Spielen erstmals menschenvergleichbare Leistungen erzielte, indem ausschließlich Rohpixel als Eingabe und der Spielstand als Belohnungssignal verwendet wurden [12]. Zwei entscheidende Stabilisierungsmechanismen ermöglichten diesen Erfolg: das *Experience Replay* [13], bei dem vergangene Erfahrungen in einem Pufferspeicher zwischengelagert und in zufälliger Reihenfolge erneut für das Training genutzt werden, um Korrelationen in den Trainingsdaten aufzubrechen, sowie ein separates Zielnetzwerk, das zeitlich verzögert aktualisiert wird, um die Lernziele zu stabilisieren.

Eine konzeptionell grundlegend andere Klasse von Algorithmen optimiert die Policy nicht indirekt über Wertfunktionen, sondern passt deren Parameter direkt in Richtung des Gradienten der erwarteten Gesamtbelohnung an. Das *REINFORCE*-Verfahren legte hierfür mit dem *Policy-Gradienten-Theorem* die theoretische Grundlage, indem gezeigt wurde, dass der Gradient der erwarteten Gesamtbelohnung bezüglich der Policy-Parameter aus gesammelten Stichproben schätzbar ist [14]. Das *Actor-Critic*-Paradigma [15], [16] verbindet die Stärken beider Ansätze: Ein Policy-basierter *Actor* wählt Aktionen aus, während ein wertfunktionsbasierter *Critic* die Güte dieser Aktionen bewertet und als Baseline zur Reduktion der Varianz des Gradientenschätzers dient. Diese hybride Architektur bildet das konzeptionelle Fundament für moderne Policy-Gradienten-Algorithmen, darunter auch den in dieser Arbeit eingesetzten PPO-Algorithmus [11].

Eine grundlegende Spannung, die alle RL-Algorithmen durchzieht und deren praktische Leistungsfähigkeit maßgeblich beeinflusst, ist der sogenannte *Exploration-Exploitation-Trade-off*: Der Agent muss kontinuierlich abwägen, ob er bekannte, bereits ertragreiche Zustände gezielt aufsucht (*Exploitation*) oder unbekannte Bereiche des Zustandsraums erkundet, um möglicherweise höherwertige Strategien zu entdecken (*Exploration*) [1]. Zu starke Exploitation führt zu einer vorzeitigen Konvergenz auf suboptimale Policies, während zu starke Exploration die Konsolidierung erlernter Strategien verhindert. In Policy-Gradienten-Methoden wird dieses Gleichgewicht häufig durch den Entropieterm in der Zielfunktion reguliert: Ein expliziter Anreiz für stochastische Aktionswahl hält die Policy in frühen Trainingsphasen ausreichend explorativ [11]. In MARL-Systemen verschärft sich dieses Problem, da die simultane Exploration mehrerer Agenten die Umgebungsdynamik zusätzlich destabilisiert und die Bewertung einzelner Explorationsstrategien erschwert [6]. Im vorliegenden Experiment wurde die Exploration zusätzlich durch ein intrinsisches Curiosity-Modul gefördert, das einen internen Vorhersagefehler als exploratives Zusatzsignal nutzte und damit die Abhängigkeit vom extrinsischen Belohnungssignal in frühen Trainingsphasen reduzierte [17].

2.2 Multi-Agent Reinforcement Learning (MARL)

Das klassische RL-Framework setzt einen einzelnen Agenten voraus, der mit einer als stationär angenommenen Umgebung interagiert. Im *Multi-Agent Reinforcement Learning* (MARL) agieren hingegen mehrere lernende Agenten simultan in einer gemeinsamen Umgebung und beeinflussen sich dabei wechselseitig durch ihre Aktionen [18]. Diese Erweiterung bringt eine Reihe fundamentaler theoretischer Herausforderungen mit sich, die im Single-Agent-Setting strukturell nicht auftreten und eigene algorithmische Lösungsansätze erfordern.

Eine der zentralen Herausforderungen ist die sogenannte *Nicht-Stationarität der Umgebung* (*Non-Stationarity*): Da alle Agenten ihre Policies gleichzeitig und unabhängig voneinander aktualisieren, verändert sich aus der Perspektive eines einzelnen Agenten die effektive Umgebungsdynamik kontinuierlich [6], [19]. Aus der Perspektive eines einzelnen Agenten erscheint die Umgebungsdynamik dadurch nicht-stationär, da sich die effektive Übergangswahrscheinlichkeit mit den sich verändernden Policies der Mitagenten kontinuierlich ändert. Die Konvergenzgarantien klassischer RL-Algorithmen gelten unter dieser Bedingung nicht

mehr ohne weiteres. Eine formale Erweiterung des MDP auf den Mehrfachagenten-Fall bietet das dezentrale, partiell beobachtbare Markov-Entscheidungsprozess-Framework (Dec-POMDP) [20], [21]. Ein Dec-POMDP wird formal als Tupel $\langle I, S, \{A_i\}, P, R, \{\Omega_i\}, O, \gamma \rangle$ definiert. Dabei repräsentiert I die endliche Menge der Agenten und S den globalen Zustandsraum. Jeder Agent $i \in I$ verfügt über eine eigene Menge möglicher Aktionen A_i. Die Zustandsübergänge werden durch die Wahrscheinlichkeitsfunktion P bestimmt, während R die kollektive Belohnungsfunktion darstellt. Der entscheidende Unterschied zum klassischen MDP liegt in der Beobachtungsfunktion: Anstatt den globalen Zustand S zu sehen, erhält jeder Agent i lediglich eine lokale Beobachtung $\omega \in \Omega_i$, gesteuert durch die Beobachtungswahrscheinlichkeit O. Dieses Modell erfasst die in der Praxis typische Situation, in der ein Agent (wie im vorliegenden Experiment durch Raycast-Sensoren simuliert) nur einen lokalen, unvollständigen Ausschnitt seiner Umgebung wahrnimmt. Die Entscheidungsfindung unter dieser partiellen Beobachtbarkeit stellt ein nachweislich komplexes algorithmisches Problem dar [20]. Für vollständig beobachtbare kooperative Szenarien bieten *Markov-Spiele* einen alternativen formalen Rahmen, der die spieltheoretische Perspektive auf Mehrfachagenten-Interaktionen explizit modelliert [22].

Ein zentrales Entwurfsprinzip im MARL-Bereich ist die Unterscheidung zwischen zentralisierten und dezentralisierten Trainings- und Ausführungsarchitekturen. Im vollständig dezentralisierten Ansatz, dem sogenannten *Independent Learning*, behandelt jeder Agent die übrigen Agenten als nicht gesondert modellierte Bestandteile seiner Umgebung und trainiert unabhängig auf Basis seiner lokalen Beobachtungen [23]. Der wesentliche Vorteil dieser Architektur liegt in ihrer Skalierbarkeit und der einfachen Übertragbarkeit bestehender Single-Agent-Algorithmen; ihr zentrales Problem besteht in der beschriebenen Nicht-Stationarität sowie in der häufig unzureichenden Koordination der entstehenden Policies [24]. Ein früher algorithmischer Ansatz zur Verbesserung des Independent Learning in kooperativen Szenarien setzt auf die selektive Nutzung ausschließlich positiver Lernerfahrungen: Ungünstige Ergebnisse werden dabei als Folge der Exploration von Mitagenten interpretiert, sodass die wahrgenommene Nicht-Stationarität der Umgebung in bestimmten kooperativen Aufgaben abgeschwächt werden kann [25]. Dieser Ansatz verdeutlicht, dass das Problem der Nicht-Stationarität nicht zwingend eine vollständige Zentralisierung des Trainings erfordert, sondern durch gezielte Anpassungen der Lernregel auch in dezentralen Architekturen gemildert werden kann. Die praktische Wirksamkeit solcher Ansätze ist

jedoch stark aufgabenabhängig und nimmt mit wachsender Agentenzahl und zunehmender Umgebungskomplexität ab [24]. Für das in dieser Arbeit verwendete Drei-Agenten-System und die physisch eng gekoppelte Push-Block-Aufgabe wurde bewusst auf solche Anpassungen verzichtet, um die isolierte Wirkung der Belohnungsstruktur ohne zusätzliche algorithmische Eingriffe zu untersuchen. Dem gegenüber steht das Paradigma des *Centralized Training with Decentralized Execution* (CTDE) [26]. Während der Trainingsphase greift ein zentraler *Critic* auf globale Zustandsinformationen aller Agenten zu, um stabilere und informiertere Gradientenschätzungen zu ermöglichen. In der Ausführungsphase agiert jeder Agent hingegen ausschließlich auf Basis seiner lokalen Beobachtungen, wodurch das Verfahren in der Praxis ohne zentrale Kommunikationsinfrastruktur auskommt und auf reale, dezentrale Systeme übertragbar bleibt.

Eine weitere zentrale Problemstellung in kooperativen MARL-Szenarien mit globaler Belohnungsverteilung ist das *Credit Assignment Problem*: Wird eine Teambelohnung an alle Agenten gemeinsam ausgeschüttet, ist es für den Lernalgorithmus mathematisch schwer zu quantifizieren, welcher einzelne Agent durch welche spezifische Aktion zum kollektiven Erfolg beigetragen hat [6], [27]. Das Belohnungssignal wird durch das Rauschen der kollektiven Gruppenleistung verfälscht, was dazu führen kann, dass Agenten erlernen, passiv zu verharren. Ein Phänomen, das in der Literatur als *Lazy-Agent*-Problem beschrieben wird [6], [27]. Aktive Beiträge einzelner Agenten zur Zielerreichung werden unter diesen Bedingungen nicht zuverlässig durch das Belohnungssignal verstärkt. Lösungsansätze hierfür umfassen unter anderem *Value Decomposition*-Methoden, die die gemeinsame Teamwertfunktion in individuelle Agentenwertfunktionen dekomponieren [28], sowie kontrafaktische Baseline-Ansätze wie *Counterfactual Multi-Agent Policy Gradients* (COMA), die den marginalen Beitrag jedes Agenten durch kontrafaktische Vergleiche isolieren [29].

Die Forschungslandschaft im MARL-Bereich hat sich in den letzten Jahren erheblich ausgeweitet, angetrieben unter anderem durch anspruchsvolle Benchmarkumgebungen wie die *StarCraft Multi-Agent Challenge* (SMAC) [30], die komplexe kooperative Szenarien mit partieller Beobachtbarkeit und hochdimensionalen Aktionsräumen kombiniert. Als standardisiertes Benchmarking-Framework für die Entwicklung und den Vergleich von MARL-Algorithmen hat sich zudem die *PettingZoo*-Bibliothek etabliert, die eine einheitliche Schnittstelle für eine Vielzahl von Mehrfachagenten-Umgebungen bereitstellt [31]. Umfassende Überblicksstudien dokumentieren die theoretische Breite der entwickelten An-

sätze sowie die Herausforderungen, die mit der Skalierung auf reale Anwendungsszenarien verbunden sind [6], [32].

Die praktische Relevanz von MARL-Forschung zeigt sich besonders deutlich in der Breite ihrer Anwendungsfelder. Während frühe Arbeiten vorwiegend auf spieltheoretischen Modellumgebungen operierten [33], hat sich das Feld in den letzten Jahren zunehmend realen Anwendungsszenarien zugewandt. Großskalige Experimente wie AlphaStar [4] und OpenAI Five [5] demonstrierten, dass MARL-Systeme in hochdimensionalen, echtzeitdynamischen Umgebungen koordiniertes Gruppenverhalten entwickeln können, das menschliche Experten übertrifft. Diese Erfolge wurden jedoch unter massivem Ressourceneinsatz und mit zentralisierten Trainingsinfrastrukturen erzielt, die für die überwiegende Mehrheit der Forschungseinrichtungen nicht reproduzierbar sind.

Für leichtgewichtige, reproduzierbare Forschungsumgebungen hat sich in den letzten Jahren ein eigenes Ökosystem an Benchmarking-Plattformen etabliert. Neben der *StarCraft Multi-Agent Challenge* [30] und *PettingZoo* [31] haben sich spielbasierte Simulationsumgebungen wie die Unity-ML-Agents-Plattform [34] als zugängliche Alternative für empirische MARL-Experimente bewährt. Sie erlauben die kontrollierte Variation einzelner Designparameter, wie der Belohnungsstruktur oder der Agentenzahl, unter reproduzierbaren Bedingungen, ohne die Hardwareanforderungen großskaliger Systeme vorauszusetzen. Diese Zugänglichkeit macht sie besonders geeignet für isolierte Kausalanalysen, wie sie das vorliegende Experiment anstrebt.

2.3 Proximal Policy Optimization (PPO)

Der im Experiment dieser Arbeit eingesetzte Algorithmus ist der *Proximal Policy Optimization*-Algorithmus (PPO) [35], der zur Klasse der Actor-Critic-basierten Policy-Gradienten-Verfahren gehört. Ein wesentliches Problem früher Policy-Gradienten-Methoden besteht darin, dass zu große Aktualisierungsschritte in der Parameteroptimierung die Policy destabilisieren und den Lernprozess irreversibel schädigen können. Das *Trust Region Policy Optimization*-Verfahren (TRPO) [36] adressierte dieses Problem erstmals systematisch, indem die Policy-Aktualisierung durch eine explizite Nebenbedingung auf eine sogenannte *Trust Region* beschränkt wird, ein Gebiet im Parameterraum, in dem die aktualisierte Policy hinreichend ähnlich zur vorherigen bleibt. Als Ähnlichkeitsmaß dient dabei die Kullback-

Leibler-Divergenz (KL-Divergenz), durch die TRPO theoretisch robuste Lerngarantien bietet. Die Lösung dieser Nebenbedingung ist jedoch in jedem Aktualisierungsschritt rechnerisch aufwendig.

PPO approximiert die Sicherheitseigenschaften von TRPO durch einen wesentlich einfacheren Mechanismus: die *Clipped Surrogate Objective*-Funktion [35]. Anstatt die Policy-Aktualisierung über eine formale Optimierungsnebenbedingung abzusichern, wird das Verhältnis zwischen der Wahrscheinlichkeit einer Aktion unter der neuen und der alten Policy, das sogenannte *Importance Ratio*, direkt geclipt. Die resultierende Zielfunktion ist formal definiert als [35]:

$$L^{\text{CLIP}}(\theta) = \mathbb{E}_t \left[\min\Big(r_t(\theta)\, \hat{A}_t,\ \text{clip}(r_t(\theta),\, 1{-}\varepsilon,\, 1{+}\varepsilon)\, \hat{A}_t \Big) \right] \tag{1}$$

wobei $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{alt}}}(a_t|s_t)}$ das Importance Ratio, $\hat{A}_t$ den Advantage-Schätzer und ε den Clipping-Schwellenwert bezeichnet. Im vorliegenden Experiment wurde $\varepsilon = 0{,}2$ gewählt (vgl. Anhang B). Übersteigt das Importance Ratio diesen Schwellenwert, wird der entsprechende Gradientenbeitrag abgeschnitten. Durch diesen Mechanismus werden destabilisierende Aktualisierungen verhindert, ohne die rechenintensive Nebenbedingungsoptimierung von TRPO zu erfordern. In der Praxis liefert PPO dabei vergleichbare oder überlegene Ergebnisse bei deutlich geringerem Implementierungsaufwand [35]. Die Parallelisierung des Trainings über mehrere simultane Umgebungsinstanzen verstärkt die Stabilität des PPO-Lernprozesses zusätzlich, da sie eine diversifiziertere Stichprobengrundlage für die Gradientenschätzung bereitstellt. Diese Technik wurde bereits bei asynchronen Actor-Critic-Methoden zur Dekorrelation der Trainingsdaten eingesetzt und ist auch für die vorliegende Implementierung zentral [37].

Ein weiterer zentraler Bestandteil der PPO-Implementierung, der die Qualität des Lernprozesses maßgeblich beeinflusst, ist die Schätzung des *Advantage*-Wertes $\hat{A}_t$ in der Zielfunktion. In der Praxis wird hierfür die *Generalized Advantage Estimation* (GAE) eingesetzt [38], die einen gewichteten Kompromiss zwischen der unverzerrten, aber hochvarianten Monte-Carlo-Schätzung und der verzerrten, aber niedrigvarianten Ein-Schritt-TD-Schätzung herstellt. Formal ergibt sich der GAE-Schätzer als exponentiell gewichtete Summe der zeitlichen Differenzfehler $\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$:

$$\hat{A}_t^{\mathrm{GAE}(\gamma,\lambda)} = \sum_{l=0}^{\infty} (\gamma\lambda)^l \, \delta_{t+l} \tag{2}$$

Der Parameter $\lambda \in [0, 1]$ steuert dabei das Gleichgewicht zwischen Varianz und Bias: Ein Wert von $\lambda = 0$ reduziert den Schätzer auf den Ein-Schritt-TD-Fehler, während $\lambda = 1$ der vollständigen Monte-Carlo-Rückkehr entspricht. Im vorliegenden Experiment wurde $\lambda = 0{,}95$ gewählt (vgl. Anhang B), ein in der Literatur etablierter Wert, der eine geringe Verzerrung bei gleichzeitig deutlich reduzierter Varianz gegenüber der reinen Monte-Carlo-Schätzung erzielt [38]. Die Stabilität des Lernprozesses, wie sie in der Entropieanalyse in Abschnitt 4.1.3 beobachtet wurde, ist unter anderem auf diese Kalibrierung des Advantage-Schätzers zurückzuführen.

Im Kontext von MARL-Szenarien wird PPO durch dezentrale Anwendung auf mehrere unabhängig trainierende Agenten zum *Independent Proximal Policy Optimization*-Verfahren (IPPO) [7]. Jeder Agent trainiert dabei eine eigene, separate Policy auf Basis seiner lokalen Beobachtungen, ohne direkten Zugriff auf die Policies oder Beobachtungen der Mitagenten zu haben. Obwohl dieser Ansatz die Nicht-Stationarität des Independent Learning nicht auflöst, konnte in jüngeren empirischen Arbeiten gezeigt werden, dass IPPO unter bestimmten Bedingungen mit vollständig zentralisierten Trainingsansätzen wie *Multi-Agent PPO* (MAPPO) konkurrenzfähig ist [39]. Dieser Befund motiviert den Einsatz von IPPO als Forschungsinstrument: Die Architektur erlaubt die isolierte Analyse der Auswirkungen der Belohnungsstruktur auf das emergente Agentenverhalten, ohne dass algorithmische Kompensationsmechanismen, etwa ein zentraler Critic, die beobachteten Effekte überlagern.

2.4 Kooperative und kompetitive Belohnungsparadigmen

Die Frage, unter welchen Bedingungen rationale Akteure zur Kooperation bereit sind oder Wettbewerb bevorzugen, zählt zu den grundlegenden Problemen der Spieltheorie und der Evolutionsbiologie. Kooperative Strategien können selbst in einem Umfeld individuellen Eigennutzes emergieren, sofern die Interaktionsstruktur wiederholte Begegnungen und eine hinreichende Gewichtung zukünftiger Erträge ermöglicht [40]. Diese Erkenntnis findet ihre direkte Entsprechung in modernen MARL-Systemen: Komplexe, nicht explizit programmierte Verhaltensweisen gehen häufig allein aus der Struktur der Belohnungsfunktion

und der physischen Umgebungsbeschaffenheit hervor.

Im Bereich des MARL lassen sich Trainingsszenarien grundsätzlich entlang einer Kooperations-Wettbewerbs-Achse einordnen [41]. Vollständig kooperative Szenarien bilden dabei eine Extremposition, in der alle Agenten ein gemeinsames Ziel verfolgen und eine geteilte Belohnung erhalten. Die entgegengesetzte Extremposition bilden vollständig kompetitive Nullsummenszenarien, in denen der Gewinn eines Agenten dem Verlust eines anderen entspricht. Zwischen diesen Extremen existiert eine breite Klasse gemischter Szenarien, die Elemente beider Paradigmen kombinieren. Experimentell wurde nachgewiesen, dass dasselbe Deep-RL-System, abhängig ausschließlich von der Struktur der Belohnungsfunktion, entweder kooperatives oder kompetitives Verhalten in einer gemeinsamen Umgebung ausbildet [33]. Die Belohnungsfunktion fungiert damit als zentrales Designinstrument, das die emergente soziale Dynamik eines MARL-Systems in fundamentaler Weise determiniert.

Eine besonders relevante Klasse von Problemen stellen die sogenannten *Sequential Social Dilemmas* dar [42]: Strukturen, in denen rational eigennützige Agenten in eine Situation geraten, die einem iterativen Gefangenendilemma ähnelt – individuelle Nutzenmaximierung führt zu kollektiv suboptimalen Ergebnissen. Derartige Dilemmata entstehen in MARL-Umgebungen, wenn Ressourcen geteilt werden müssen und kurzfristig vorteilhaftes Wettbewerbsverhalten die kollektive Handlungseffizienz dauerhaft schädigt. Interventionen auf Ebene der Belohnungsfunktion, etwa die Modellierung einer Präferenz für faire Ergebnisverteilungen, können in solchen Szenarien die emergente Kooperation substanziell verbessern [43]. Empirisch wurde zudem nachgewiesen, dass das unreflektierte Teilen von Belohnungen in koordinierten Gruppenaufgaben die Effizienz der kollektiven Handlungsausführung signifikant untergraben kann [44], da individuelle Beiträge im Rauschen des gemeinsamen Signals nicht mehr zuverlässig verstärkt werden.

Großskalige MARL-Experimente haben darüber hinaus gezeigt, dass aus der Interaktion vieler Agenten hochkomplexe, emergente Verhaltensweisen entstehen können, die weit über das explizit Trainierte hinausgehen. In einer Versteck-und-Suche-Simulation wurde die Emergenz mehrerer aufeinander aufbauender Strategien und Gegenstrategien beobachtet, die ausschließlich durch den wechselseitigen Wettbewerbsdruck zwischen Agenten hervorgebracht wurden [45]. Unter geeigneten kooperativen Bedingungen entwickeln Agenten emergente Kommunikationsformen, ohne dass eine explizite Kommuni-

kationsstruktur vorgegeben wird [46]. Diese Befunde unterstreichen, dass die Wahl des Belohnungsparadigmas, ob kooperativ, kompetitiv oder hybrid, eine der einflussreichsten Designentscheidungen im gesamten MARL-System darstellt und das emergente Verhalten der Agenten in einer Weise prägt, die aus rein theoretischen Überlegungen heraus schwer vorherzusagen ist [47]. Die im empirischen Teil dieser Arbeit untersuchten Belohnungsstrukturen (individuelle Belohnung versus geteilte Teambelohnung) repräsentieren damit zwei klar abgrenzbare Positionen innerhalb dieses konzeptionellen Spektrums.

Für das Design von MARL-Systemen in der Praxis ergibt sich aus diesen Befunden eine zentrale Designfrage: Unter welchen Bedingungen ist eine kooperative Belohnungsstruktur einer kompetitiven vorzuziehen, und wann kippt der Vorteil in die entgegengesetzte Richtung? Die Gestaltung von Belohnungssystemen gilt als eines der fundamentalen offenen Probleme der kooperativen KI-Forschung [47]. Dabei hängt die optimale Wahl nicht allein von der abstrakten Aufgabenstruktur ab, sondern auch von der konkreten algorithmischen Architektur: Eine globale Teambelohnung, die in einem zentralisierten Trainingsrahmen mit vollständiger Zustandsbeobachtung funktioniert, kann in einer dezentralen Architektur mit partieller Beobachtbarkeit zu genau den Dysfunktionen führen, die in dieser Arbeit empirisch untersucht werden.

Die Literatur zeigt zudem, dass hybride Belohnungsstrukturen, die individuelle und kollektive Anreize kombinieren, in bestimmten Szenarien die Nachteile beider Extrempositionen abmildern können [43]. Dabei ist die Kalibrierung des relativen Gewichts zwischen individuellem und kollektivem Belohnungsanteil ein nicht-triviales Optimierungsproblem, das erheblichen Einfluss auf das emergente Verhalten haben kann. Das vorliegende Experiment setzt an dieser Stelle an und untersucht bewusst die beiden reinen Extrempositionen (vollständig individuelle versus vollständig geteilte Belohnung) in einer kontrollierten Simulationsumgebung.

2.5 Verwandte empirische Arbeiten

Die theoretischen Grundlagen aus den vorangegangenen Abschnitten spiegeln sich in einer wachsenden Zahl empirischer Studien wider, die den Einfluss von Belohnungsstrukturen auf das Agentenverhalten in konkreten Simulationsszenarien untersuchen. Eine kritische Einordnung dieser Arbeiten erlaubt es, den spezifischen Beitrag der vorliegenden

Untersuchung zu verorten.

Die systematische Untersuchung von Belohnungsstrukturen in physikbasierten Kooperationsaufgaben wurde maßgeblich durch Arbeiten vorangetrieben, die zeigen, dass dasselbe tiefe Reinforcement-Learning-System durch ausschließliche Variation der Belohnungsfunktion entweder kooperatives oder kompetitives Verhalten ausbildet [33]. Dieser Befund legte den konzeptionellen Grundstein für das vorliegende Experiment, da er die Belohnungsfunktion als primäre Steuerungsvariable für emergentes Gruppenverhalten etabliert, unabhängig von der zugrundeliegenden Algorithmusarchitektur.

Im Bereich physikalisch gekoppelter Kooperationsaufgaben liefern großskalige Experimente wie OpenAI Hide-and-Seek [45] wichtige Referenzpunkte: In einer Versteck-und-Suche-Simulation mit kompetitiver Belohnungsstruktur entstanden mehrere aufeinander aufbauende Strategiegenerationen allein durch wechselseitigen Wettbewerbsdruck. Diese Arbeit belegt, dass kompetitive Belohnungsstrukturen auch in Mehrfachagenten-Szenarien zu hochkomplexem, emergenten Gruppenverhalten führen können. Sie wurde jedoch unter einem Ressourcenaufwand durchgeführt, der für unabhängige Replikationen meist nicht verfügbar ist [4].

Für den spezifischen Kontext kooperativer Aufgaben mit geteilter Belohnung ist ein weiterer empirischer Befund besonders relevant: In einer agentenbasierten Jagdsimulation wurde nachgewiesen, dass das Teilen von Belohnungen unter bestimmten Umgebungsbedingungen die kollektive Koordination nicht verbessert, sondern messbar verschlechtert [44]. Dieser Befund steht in direktem Widerspruch zur intuitiven Erwartung, dass geteilte Belohnungen Kooperation fördern, und motiviert die vergleichende Untersuchung beider Paradigmen im vorliegenden Experiment.

Im Bereich dezentraler Algorithmusarchitekturen zeigen jüngere Vergleichsstudien, dass *Independent PPO* (IPPO) unter kooperativen Bedingungen mit vollständig zentralisierten Ansätzen wie MAPPO konkurrenzfähig sein kann [39]. Gleichzeitig belegen Analysen, dass IPPO mit globalen Belohnungsstrukturen strukturell benachteiligt ist, da die Nicht-Stationarität der Umgebung das Belohnungssignal zusätzlich verrauscht [7]. Diese Befundlage motiviert die Wahl von IPPO als analytisches Werkzeug im vorliegenden Experiment: Die Architektur legt die spezifischen Schwächen unstrukturierter Teambelohnungen in ihrer reinsten Form frei, ohne dass algorithmische Kompensationsmechanismen die Wirkung

der Belohnungsstruktur überlagern.

Gegenüber den zitierten Arbeiten unterscheidet sich das vorliegende Experiment in einem zentralen Aspekt: Der Vergleich erfolgt in einer physikbasierten Umgebung mit expliziter Massedifferenzierung der Objekte, die einen natürlichen Koordinationszwang für bestimmte Aufgabenkategorien erzeugt. Diese physikalische Einschränkung ist in keiner der genannten Referenzarbeiten in vergleichbarer Form systematisch als Koordinationsmechanismus untersucht worden. Die vorliegende Arbeit schließt damit eine spezifische empirische Lücke: den kontrollierten, isolierten Vergleich kooperativer und kompetitiver Belohnungsstrukturen unter dem Einfluss physikalisch erzwungener Kooperationsnotwendigkeit in einer reproduzierbaren, ressourcenarmen Simulationsumgebung.

3 Methodik und Versuchsaufbau

Das methodische Vorgehen wurde zur empirischen Beantwortung der zentralen Forschungsfrage dieser Arbeit – *„Führt der Einsatz von gemeinsamen Team-Belohnungen (Shared Rewards) dazu, dass NPCs messbar effizientere Strategien entwickeln und eine höhere Erfolgsrate erzielen, als bei individuellem Training?“* – detailliert dargelegt. Um belastbare Aussagen über das emergente Verhalten unter verschiedenen Belohnungsstrukturen treffen zu können, war ein kontrollierter, reproduzierbarer und isolierter Versuchsaufbau essenziell.

Die Darstellung der Methodik gliederte sich hierfür in mehrere logische Phasen: Dies umfasste zunächst die Beschreibung der technischen Simulationsumgebung sowie des modifizierten *Push-Block*-Szenarios, welches als physische Grundlage für die Experimente diente. Darauf aufbauend wurde die Architektur des vergleichenden Versuchsaufbaus für die beiden kontrastierenden Belohnungsstrukturen (Team versus Ego) spezifiziert, um die unabhängige Variable des Experiments klar abzugrenzen. Abschließend wurden der angewandte *Curriculum Learning*-Ansatz zur Reduktion der initialen Komplexität sowie die gewählten Hyperparameter für das Training der neuronalen Netze erläutert.

3.1 Die Simulationsumgebung (Unity & ML-Agents)

Für die praktische Durchführung der Experimente und das Training der Agenten wurde die Game-Engine Unity (Version 6000.3.8f1 LTS) unter dem Betriebssystem Windows 11 eingesetzt. Die Entwicklung der spezifischen Anpassungen in C# erfolgte in Visual Studio 2026. Die zentrale Schnittstelle zwischen der 3D-Simulation und den Machine-Learning-Algorithmen bildete das Open-Source-Toolkit *Unity ML-Agents* [34], [48]. Das Training der neuronalen Netze wurde rein CPU-basiert über eine isolierte virtuelle Umgebung (venv) in Python (Version 3.9.13) unter Verwendung des PPO-Algorithmus [35] ausgeführt, welcher durch die dezentrale Anwendung auf mehrere Agenten methodisch einem *Independent Proximal Policy Optimization* (IPPO) [7] entsprach.

3.2 Das Push-Block-Szenario

Als Grundlage für die Evaluation diente das etablierte *Push-Block*-Szenario aus dem offiziellen Unity ML-Agents GitHub-Repository, welches die Grundmechanik für die physischen Interaktionen stellte (Fig. 1) [48].

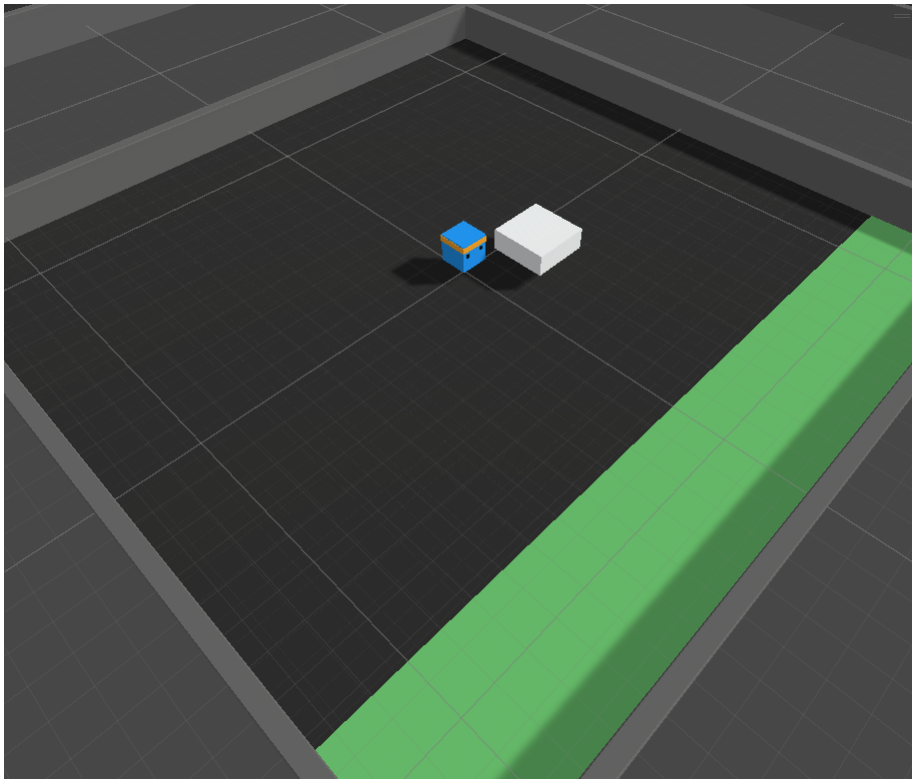

Fig. 1: Grundlegende Mechanik des Push-Block-Szenarios in der Unity-Engine. Eigene Darstellung

Um die Forschungsfrage empirisch fundiert zu beantworten, wurde das Basis-Szenario für ein vergleichendes Versuchsdesign signifikant modifiziert. Beide Experimente fanden in exakt derselben Multi-Agenten-Arena statt (Fig. 2), in der drei Agenten gleichzeitig agierten und insgesamt sechs Blöcke unterschiedlicher Gewichtsklassen, die teilweise die Kraft mehrerer Agenten erforderten, in das Ziel befördern mussten.

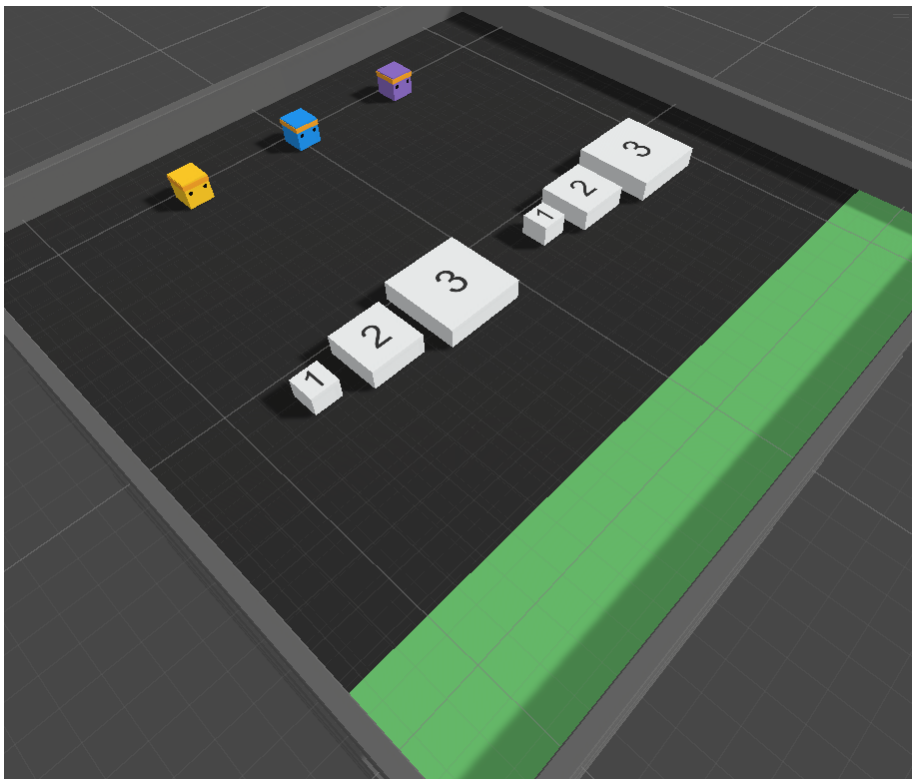

Fig. 2: Modifizierte Multi-Agenten-Arena mit drei Agenten und sechs Blöcken unterschiedlicher Masse. Eigene Darstellung.

3.2.1 Observation und Action Space

Für die Interaktion mit der Umgebung wurde für jeden Agenten ein spezifischer Wahrnehmungs- und Handlungsraum definiert. Der *Observation Space* (Eingabevektor des neuronalen Netzes) basierte primär auf räumlicher Wahrnehmung durch Raycasts (*Ray Perception Sensor 3D*) [34]. Jeder Agent emittierte dabei insgesamt sieben Strahlen (drei pro Seite sowie einen Frontalstrahl) in einem maximalen Sichtfeld von 70 Grad und einer Reichweite von 20 virtuellen Einheiten, um die relativen Distanzen und Objekttypen (Begrenzungswände, verschiebbare Blöcke, die Zielzone sowie Mitagenten) in seiner unmittelbaren Umgebung zu detektieren. Diese visuelle Wahrnehmung wurde durch die eigene lokale Rotation und Geschwindigkeit ergänzt.

Der *Action Space* (Ausgabevektor) wurde als diskreter Raum mit einem einzigen Aktionszweig (*Branch*) definiert, welcher sieben distinkte Handlungsmöglichkeiten umfasste: Untätigkeit (Action 0), Bewegung entlang der Längsachse nach vorne (Action 1) und hinten (Action 2), Rotation um die Hochachse nach rechts (Action 3) und links (Action 4) sowie laterale Seitwärtsbewegungen nach links (Action 5) und rechts (Action 6). Die physikalische Umsetzung in der Game-Engine erfolgte durch die Applikation direkter Kraftvektoren (*ForceMode.VelocityChange*) auf den *Rigidbody* des Agenten.

3.2.2 Physikalische Objektklassen und Massedifferenzierung

Die Arena enthielt insgesamt sechs Blöcke, die in drei Gewichtsklassen unterteilt waren. Diese Differenzierung war keine rein ästhetische Designentscheidung, sondern die zentrale mechanische Variable des Experiments: Sie legte fest, wie viele Agenten für das Bewegen eines Blocks physikalisch erforderlich waren, und erzeugte damit eine gestufte Koordinationsanforderung.

Blöcke der Klasse 0 (leicht) ließen sich von einem einzelnen Agenten ohne nennenswerten Kraftaufwand in die Zielzone manövrieren und stellten damit die Grundaufgabe des Experiments dar. Blöcke der Klasse 1 (mittel) überschritten die physikalisch sinnvolle Einzelkraft eines Agenten und erforderten das koordinierte Zusammenwirken von zwei Agenten, um innerhalb des Episodenzeitlimits in die Zielzone befördert zu werden. Blöcke der Klasse 2 (schwer) konnten ausschließlich durch das gleichzeitige Anschieben aller drei Agenten innerhalb des Episodenzeitlimits von 1000 Entscheidungsschritten in die Zielzone befördert werden. Diese Klasse stellte damit die höchste Koordinationsanforderung des Experiments dar.

Die Belohnungswerte waren entsprechend gestaffelt: Für Klasse 0 wurden +1,0, für Klasse 1 +2,0 und für Klasse 2 +3,0 Punkte vergeben. Diese Skalierung setzte einen expliziten Anreiz, physikalisch anspruchsvollere Objekte zu priorisieren, und verstärkte damit den Selektionsdruck in Richtung kooperativer Interaktionen bei mittleren und schweren Blöcken.

3.3 Design der Belohnungsfunktionen

Um die primäre Forschungsfrage dieser Arbeit empirisch fundiert beantworten zu können, wurde ein vergleichendes Drei-Gruppen-Design entworfen. Das Versuchsdesign umfasste explizit drei Trainingsläufe: die *Team-Baseline* als kooperativen Ansatz mit geteilter Belohnung ohne Curriculum, den *Team-Curriculum*-Ansatz als kooperativen Ansatz mit geteilter Belohnung unter Einsatz eines Curriculums sowie den *Ego-Ansatz* als kompetitive Variante mit individueller Belohnung und identischem Curriculum.

Auf die Evaluation einer vierten Kondition (Ego ohne Curriculum) wurde methodisch bewusst verzichtet. Da die Architektur der geteilten Belohnung bei der Team-Baseline aufgrund des sogenannten *Credit Assignment Problems* [6] erwartungsgemäß zu keiner zielführenden Strategie führte, war die Einführung des Curriculums die zwingende Voraussetzung für eine erfolgreiche Aufgabenerfüllung. Dies reduziert allerdings die Möglichkeit, den isolierten Curriculum-Effekt vollständig zu bestimmen. Um das Ceteris-paribus-Prinzip für den eigentlichen Vergleich der Belohnungsstrukturen (Team versus Ego) zu wahren, musste der Ego-Ansatz exakt denselben Curriculum-Bedingungen unterworfen werden. Ein unstrukturierter Ego-Lauf hätte zwar zusätzliche Informationen geliefert, lag aber außerhalb des fokussierten Vergleichs zwischen Team-Curriculum und Ego-Curriculum.

Die Wahl von IPPO als gemeinsame algorithmische Basis für alle drei Trainingsgruppen folgte dem Prinzip der maximalen Isolierung der unabhängigen Variable. Hätten die kooperativen Gruppen einen zentralisierten Trainingsalgorithmus und der Ego-Ansatz einen dezentralisierten Algorithmus verwendet, wäre es nicht möglich gewesen, die Auswirkungen der Belohnungsstruktur von den Auswirkungen der Algorithmusarchitektur zu trennen. Durch die einheitliche Verwendung von IPPO wurde sichergestellt, dass beobachtete Unterschiede im Lernverhalten auf die Gestaltung der Belohnungsfunktion zurückzuführen sind und nicht auf algorithmische Unterschiede. Diese Entscheidung ging bewusst zu Lasten der absoluten Leistungsfähigkeit der kooperativen Gruppen, da bekannt ist, dass IPPO mit globalen Belohnungen strukturell benachteiligt ist [7]. Sie stellte jedoch eine notwendige Bedingung für die interne Validität des Vergleichs dar. Die Ergebnisse dürfen daher nicht als endgültiges Urteil über kooperative Belohnungsstrukturen im Allgemeinen verstanden werden, sondern beziehen sich explizit auf dezentrale IPPO-basierte Architekturen ohne zentralen Critic.

3.3.1 Ansatz 1: Egoistische Belohnung (Kompetitiv)

Im ersten Experiment, der Individual-Gruppe (Ego), wurde ein striktes *Individual Reward System* implementiert. Das C#-Skript der Umgebung berechnete präzise im Moment der Zielerreichung die euklidische Distanz aller Agenten zum versenkten Block. Ausschließlich der Agent mit der geringsten Distanz (also jener, der den Block physisch in das Ziel geschoben hatte) erhielt die positive Belohnung (die programmtechnische Umsetzung dieser Zuweisung in C# ist in Anhang A dokumentiert). Dies erzeugte ein scharfes Belohnungssignal ohne Rauschen, beförderte jedoch einen kompetitiven Wettbewerb um die Blöcke. Die individuelle Zielbelohnung betrug $+1{,}0$ für kleine, $+2{,}0$ für mittlere und $+3{,}0$ für schwere Blöcke, wobei ausschließlich der nächststehende Agent diese Belohnung erhielt. Ergänzend wirkte eine kontinuierliche Zeitdruckstrafe (*Hurry Up Penalty*) von $-0{,}5$ (skaliert) pro Zeitschritt auf jeden aktiven Agenten. Jede Kollision mit den Begrenzungswänden der Arena zog einen zusätzlichen Abzug von $-0{,}1$ nach sich.

Die Gestaltung dieser Belohnungsfunktion folgte dem Prinzip der *Policy Invariance under Reward Transformations* [49]: Entscheidend für das erlernte Verhalten ist nicht die absolute Höhe der Belohnung, sondern die relative Differenz zwischen belohnten und unbelohnten Aktionen sowie die zeitliche Nähe zwischen Aktion und Belohnungssignal. Durch die distanzbasierte Zuweisung (ausschließlich der physisch nächststehende Agent wurde belohnt), wurde ein unmittelbares, kausales Belohnungssignal erzeugt, das die räumliche Interaktion mit dem Block direkt verstärkte. Die Hurry-Up-Penalty erfüllte dabei eine doppelte Funktion: Sie verhinderte passives Verharren als suboptimale Ausweichstrategie und erhöhte den Selektionsdruck zugunsten schneller, zielgerichteter Aktionssequenzen. Die Wandkollisionsstrafe wiederum zielte darauf ab, destruktive Ausweichmanöver bei der Ressourcenkonkurrenz zu begrenzen, indem sie physikalisch riskantes Verhalten mit einem negativen Signal assoziierte.

3.3.2 Ansatz 2: Geteilte Belohnung (Kooperativ)

Im zweiten Experiment wurde die völlig gleiche Umgebung genutzt, jedoch zielte die Belohnungsfunktion nun auf die Maximierung des globalen Team-Erfolgs ab (*Shared Rewards*). Erreichte ein Block das Ziel, wurde eine globale, positive Belohnung an das gesamte Team ausgeschüttet, unabhängig vom individuellen Beitrag eines einzelnen Agen-

ten (die programmtechnische Einbindung in die *SimpleMultiAgentGroup* ist in Anhang A aufgeführt). Dieses Setup sollte theoretisch Teamwork fördern, barg jedoch das Risiko des *Credit Assignment Problems* [6], da auch unbeteiligte Agenten belohnt wurden. Analog zum Zeitdruck des Ego-Modells erhielt das gesamte Team pro Zeiteinheit einen gemeinsamen Punkteabzug (Hurry Up Penalty), um die kollektive Aufgabenlösung zu beschleunigen.

Zur formalen Präzisierung lassen sich beide Belohnungsfunktionen kompakt notieren. Sei $\mathcal{A} = \{a_1, a_2, a_3\}$ die Menge der drei Agenten, b_k ein versenkter Block der Gewichtsklasse $k \in \{0, 1, 2\}$ mit Belohnungswert $r_k \in \{1{,}0,\ 2{,}0,\ 3{,}0\}$ und $d(a_i, b_k)$ die euklidische Distanz zwischen Agent a_i und Block b_k im Moment der Zielerreichung. Die egoistische Belohnungsfunktion für Agent a_i lautet dann:

$$R_{\text{Ego}}(a_i, b_k) = \begin{cases} r_k & \text{falls } a_i = \arg\min_{a \in \mathcal{A}} d(a, b_k) \\ 0 & \text{sonst} \end{cases} \tag{3}$$

Die kooperative Belohnungsfunktion verteilt hingegen denselben Betrag an alle Agenten über die *SimpleMultiAgentGroup*:

$$R_{\text{Team}}(a_i, b_k) = r_k \qquad \forall\ a_i \in \mathcal{A} \tag{4}$$

Diese Gegenüberstellung verdeutlicht den strukturellen Unterschied beider Ansätze: Während R_{Ego} ein scharfes, selektives Signal erzeugt, das ausschließlich den physisch beitragenden Agenten verstärkt, verteilt R_{Team} denselben Informationsgehalt gleichmäßig auf alle Agenten und verwässert damit die individuelle Kausalverknüpfung zwischen Aktion und Belohnung.

3.4 Curriculum Learning Setup

Wie die Auswertung der unstrukturierten Team-Baseline (vgl. Kapitel 4.1) empirisch belegte, konvergierte der PPO-Algorithmus bei der direkten Konfrontation mit dem finalen Szenario (sechs Blöcke) unter einer geteilten Belohnungsstruktur nicht. Die theoretische Ursache hierfür lag primär im sogenannten *Credit Assignment Problem* [6]. In Multi-Agenten-Systemen mit globalen Belohnungen war es für das neuronale Netz mathematisch

schwer zu quantifizieren, welcher spezifische Agent durch seine individuelle Aktion zum Teamerfolg beigetragen hatte. Dies erzeugte ein starkes Rauschen (Noise) im Signal und hinderte die Agenten daran, den hochdimensionalen Zustandsraum effektiv zu explorieren.

Um dieser anfänglichen Überforderung entgegenzuwirken und die Komplexität systematisch aufzubauen, wurde als methodische Fallback-Strategie ein *Curriculum Learning*-Ansatz [50], [51] implementiert. Die hierfür definierten Schwellenwerte (*Thresholds*) für den Aufstieg in die nächste Lektion sind in der YAML-Konfiguration in Anhang B hinterlegt. Um das Ceteris-paribus-Prinzip für den Vergleich strikt zu wahren, durchliefen sowohl das Team- als auch das Ego-Modell exakt denselben Curriculum-Plan. Das Training gliederte sich in drei aufeinanderfolgende Lektionen mit steigender Komplexität. Die erste Lektion, *Lesson 0* (*Small Blocks*), beschränkte sich auf zwei aktive kleine Blöcke; der Aufstiegsschwellenwert lag bei einem Reward-Wert von $-0{,}6$. Mit *Lesson 1* (*Medium Blocks*) wurden zwei mittlere Blöcke ergänzt, sodass vier Blöcke aktiv waren, bei einem Schwellenwert von $-0{,}2$. *Lesson 2* (*Large Blocks*) stellte das vollständige End-Szenario mit allen sechs Blöcken dar; ein weiterer Aufstiegsschwellenwert war nicht definiert, sodass das Training bis zum Erreichen der maximalen Schrittanzahl fortgeführt wurde.

3.5 Trainingskonfiguration und Hardware-Setup

Um die Validität der Ergebnisse zu gewährleisten, wurden beide Gruppen (Team und Ego) mit exakt denselben Hyperparametern trainiert. Die vollständige Konfigurationsdatei `trainer_config.yaml` inklusive aller PPO-Parameter ist in Anhang B abgedruckt. Das Training von $10\,000\,000$ Umgebungsschritten beanspruchte pro Durchlauf durchschnittlich sieben Stunden.

Um die Datenerhebung zu beschleunigen und die Diversität der gesammelten Trainingsdaten zu erhöhen, wurden in Unity neun identische Arenen parallel instanziiert. Diese gängige Praxis im Bereich des Deep Reinforcement Learning dekorrelierte die Trainingsdaten und trug maßgeblich zur Stabilität des PPO-Algorithmus bei [37]. Das Training wurde auf einem System mit einem Intel Core i7-1360P Prozessor (12 Kerne) und 16 GB Arbeitsspeicher ausgeführt. Da lediglich eine integrierte Intel Iris Xe Grafikeinheit zur Verfügung stand und somit keine hardwarebeschleunigte Tensor-Verarbeitung (CUDA) möglich war, erfolgte die Berechnung der neuronalen Netze vollständig CPU-basiert.

Diese hardware-bedingte Einschränkung hatte direkte Auswirkungen auf die Trainingseffizienz: CPU-basiertes Training ist für neuronale Netze der verwendeten Größenordnung (zwei Hidden-Layer mit je 256 Einheiten) grundsätzlich geeignet, jedoch signifikant langsamer als GPU-beschleunigtes Training unter Verwendung von CUDA-Tensorkerneln. Die resultierende Trainingsdauer von durchschnittlich sieben Stunden pro Lauf schränkte die Anzahl möglicher Wiederholungsläufe im Rahmen dieser Arbeit praktisch ein. Um dennoch eine ausreichende Diversität in den gesammelten Erfahrungsdaten zu gewährleisten, wurden neun parallele Arenen instanziiert [37], was die effektive Datenmenge pro Zeiteinheit erhöhte, ohne die Rechenanforderungen überproportional zu steigern. Für zukünftige Replikationsstudien, die eine statistische Absicherung der Ergebnisse über mehrere unabhängige Trainingsläufe anstreben, wird der Einsatz einer CUDA-fähigen GPU dringend empfohlen. Die in dieser Arbeit gewählte Hardware-Konfiguration ist damit als bewusste Einschränkung zu verstehen, die den Zugang zur Forschungsplattform für ressourcenarme Umgebungen dokumentiert, gleichzeitig aber die Replikierbarkeit der Ergebnisse unter erweiterten Hardware-Bedingungen als offene Forschungsfrage zurücklässt.

Das neuronale Netz bestand aus zwei Hidden-Layers mit jeweils 256 Einheiten (*hidden_units*). Die Lernrate wurde auf 0.0003 bei einem linearen Abbau festgelegt. Die Batch-Größe betrug 1024 bei einer Puffer-Größe von 10240. Zur Förderung der Exploration in dem hochdimensionalen Zustandsraum wurde zudem ein intrinsisches Curiosity-Modul mit einer Stärke von 0.05 aktiviert. Dieser Mechanismus nutzte einen internen Vorhersagefehler als intrinsisches Belohnungssignal, um die Agenten zur Erkundung noch unbekannter Zustände zu motivieren [17].

Ein entscheidendes technisches Detail für die Evaluierung der Effizienz war die Konfiguration des *Decision Requesters*. Um Rechenkapazität zu schonen, nutzten die Agenten eine *Decision Period* von 5. Da die maximale Episodenlänge im Unity-Agenten-Skript auf 5000 Engine-Schritte festgelegt wurde, resultierten daraus genau 1000 neuronale Entscheidungsschritte pro Episode. Diese 1000 Entscheidungen bildeten das harte Time-Out-Limit des Environments.

Die Aufzeichnung sämtlicher Trainingsmetriken erfolgte über das in ML-Agents integrierte *TensorBoard* [52]. Für die finale Auswertung und visuelle Aufbereitung der exportierten Log-Daten in diesem Dokument wurde *Python* in Verbindung mit den Bibliotheken *Pandas*

[53] (zur Datenstrukturierung) und *Matplotlib* [54] (zur Grafikerstellung) verwendet.

4 Ergebnisse

Zur Beantwortung der Forschungsfrage wurden im Folgenden die Resultate der durchgeführten Simulationen präsentiert und vergleichend analysiert. Grundlage hierfür bildeten die exportierten Log-Daten der drei finalen Trainingsläufe (Team ohne Curriculum, Team mit Curriculum sowie Ego-Ansatz), die jeweils über eine exakte Distanz von zehn Millionen Entscheidungsschritten protokolliert wurden.

Die Auswertung gliederte sich methodisch in zwei Bereiche: Zunächst erfolgte in Abschnitt 4.1 eine quantitative Analyse der etablierten Reinforcement-Learning-Metriken, um die erlernte Effektivität, die operative Effizienz sowie die algorithmische Stabilität der Modelle objektiv zu bewerten. Zur besseren visuellen Beurteilung der stark oszillierenden Rohdaten wurden die Metriken in den grafischen Darstellungen zusätzlich durch einen gleitenden Mittelwert (*Moving Average*) geglättet. Darauf aufbauend schloss sich in Abschnitt 4.2 eine qualitative Betrachtung an, welche das in der 3D-Umgebung beobachtete Verhalten und die emergenten Strategien der Agenten detailliert beschrieb.

4.1 Quantitative Analyse der Trainingsmetriken

Um die Leistung der drei trainierten Modelle objektiv zu evaluieren, wurden im Folgenden die primären Metriken des Reinforcement-Learning-Prozesses isoliert betrachtet. Dies umfasste die akkumulierten Zielbelohnungen, die benötigte Zeit pro Episode sowie die mathematische Stabilität der neuronalen Netze.

4.1.1 Lerneffektivität (Extrinsic Reward)

Die zentrale Metrik zur Beurteilung der Aufgabenlösung ist der extrinsische Reward (*Extrinsic Reward*), der in der RL-Literatur als primärer Indikator für die Aufgabenerfüllung herangezogen wird [1]. In der vorliegenden Push-Block-Umgebung korreliert dieser Wert direkt mit der Anzahl und der Gewichtsklasse der erfolgreich in die Zielzone beförderten Blöcke. Der Lernfortschritt der drei Agenten-Konfigurationen wurde über den gesamten Trainingszeitraum aufgezeichnet und durch einen gleitenden Mittelwert geglättet visualisiert (Fig. 3).

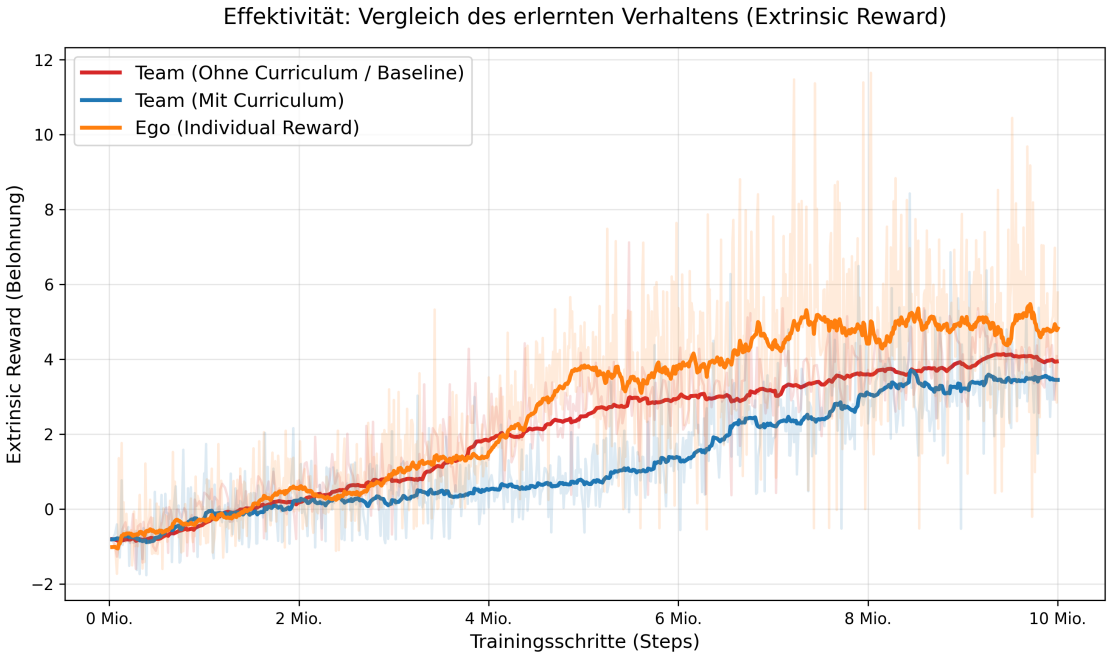

Fig. 3: Verlauf des Extrinsic Rewards über 10 Millionen Trainingsschritte. Eigene Darstellung auf Basis der TensorBoard-Logdaten.

Die deskriptive Betrachtung der Lernkurven sowie der zugrunde liegenden Rohdaten (transparente Graphen im Hintergrund) zeigte deutliche Unterschiede im Kurvenverlauf der drei Ansätze auf:

Das kompetitive Ego-Modell erzielte die höchsten absoluten Punktewerte, verzeichnete jedoch gleichzeitig die stärksten Schwankungen. Nach einem Start im negativen Bereich überschritt die Kurve als erste die Nulllinie und erreichte nach zwei Millionen Schritten einen Wert von ca. 0,5. Im darauffolgenden Intervall bis vier Millionen Schritte folgte ein steiler Anstieg auf ca. 2,0. Zwischen vier und sieben Millionen Schritten verdoppelte sich dieser Wert rasch auf einen Spitzenwert von annähernd 5,0. Im letzten Trainingsdrittel pendelte sich die geglättete Kurve auf einem Plateau zwischen 4,5 und 5,0 ein. Die univariaten Rohdaten zeigten in dieser Phase extreme Ausschläge nach oben (teilweise 10 bis 12 Punkte), aber auch tiefe Einbrüche.

Im Gegensatz dazu wies die kooperative Baseline ohne Curriculum den stetigsten Kurvenverlauf auf. Bis zur Marke von etwa 3,5 Millionen Schritten verlief die Entwicklung nahezu deckungsgleich zum Ego-Ansatz. Ab vier Millionen Schritten (ca. 2,0 Punkte) ging die Baseline in ein moderateres, lineares Wachstum über. Die Kurve erreichte bei

acht Millionen Schritten einen Wert von ca. 3,5 und flachte am Ende des Trainings bei einem Wert von ca. 4,0 ab. Die Rohdaten gruppierten sich hierbei deutlich enger um die geglättete Linie als beim Ego-Ansatz.

Demgegenüber war der Verlauf des Curriculum-Setups durch lange Stagnationsphasen gekennzeichnet. In den ersten vier Millionen Schritten verharrte der gleitende Mittelwert geringfügig oberhalb der Nulllinie (0,0 bis 0,5). Ein langsamer Anstieg über die 1,0-Marke ließ sich erst zwischen vier und sechs Millionen Schritten beobachten. Der stärkste Zuwachs erfolgte im Intervall zwischen sechs und acht Millionen Schritten, in dem die Kurve von ca. 1,5 auf über 3,0 Punkte stieg. Am Ende der 10 Millionen Schritte stagnierte der Wert bei ca. 3,5 und verblieb damit unterhalb der unstrukturierten Baseline.

Die vergleichende Betrachtung der drei Lernkurven verdeutlicht, dass die Belohnungsstruktur einen stärkeren Einfluss auf den Lernverlauf ausübte als der Einsatz von Curriculum Learning. Der Ego-Ansatz überholte die Baseline trotz identischer algorithmischer Grundlage und trotz der durch das Curriculum verkürzten effektiven Trainingszeit in der finalen Aufgabenstufe. Besonders aufschlussreich ist der Verlauf der ersten vier Millionen Schritte: Alle drei Kurven starteten aus ähnlichen Ausgangspositionen und entwickelten sich bis zur Zweimillionen-Marke nahezu parallel. Die Divergenz begann erst ab diesem Punkt, ein Hinweis darauf, dass die Belohnungsstruktur ihre differenzierende Wirkung nicht unmittelbar, sondern erst nach einer initialen Explorationsphase entfaltete, in der alle Agenten zunächst grundlegende Interaktionsmuster mit den Blöcken erlernten. Dieser Befund legt nahe, dass die Wahl der Belohnungsstruktur vor allem in fortgeschrittenen Trainingsphasen, wenn die Agenten beginnen, spezifischere Strategien zu konsolidieren, maßgeblich zur Strategieausbildung beizutragen scheint. Die beobachtete Divergenz nach einer gemeinsamen Explorationsphase ist konsistent mit der in der Literatur beschriebenen zunehmenden Bedeutung der Belohnungsstruktur bei steigender Policy-Komplexität [41].

4.1.2 Aufgabeneffizienz (Episode Length)

Ergänzend zur reinen Punkteausbeute wird die durchschnittliche Dauer bis zum Abschluss einer Episode (*Episode Length*) betrachtet. Das harte Zeitlimit pro Episode ist in der Simulationsumgebung auf exakt 1000 Entscheidungsschritte begrenzt. Ein Erreichen dieses Limits bedeutet, dass die Episode durch ein Time-Out und nicht durch das vorzeitige

Abräumen aller Blöcke beendet wurde. Der Verlauf dieser Metrik ist in Fig. 4 dargestellt.

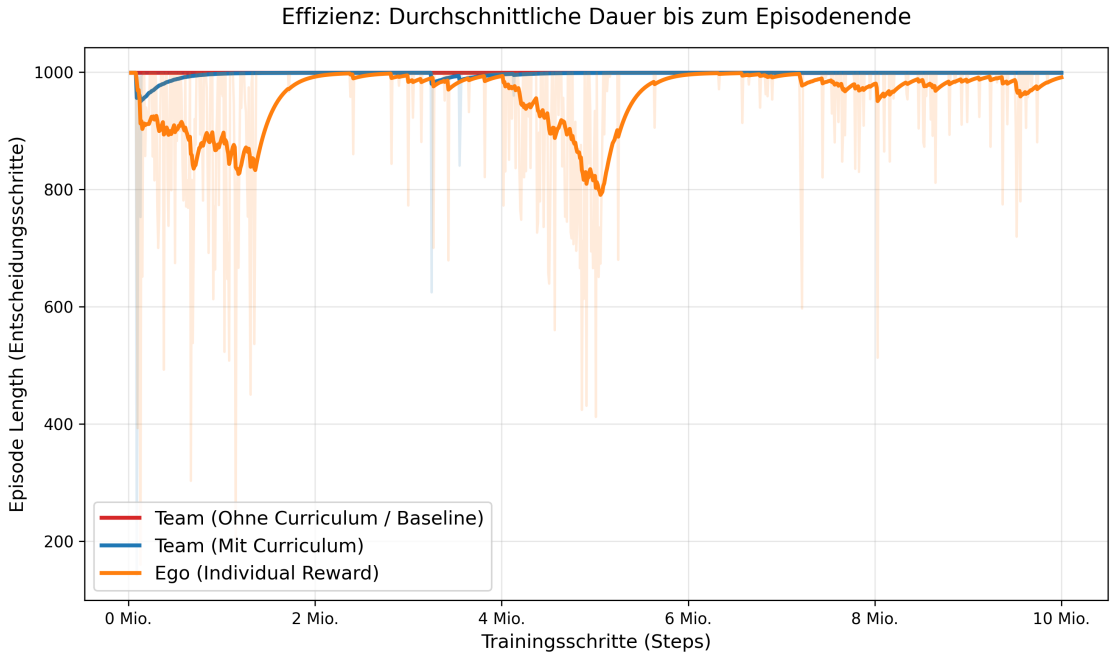

Fig. 4: Entwicklung der durchschnittlichen Episodenlänge über 10 Millionen Trainingsschritte. Eigene Darstellung auf Basis der TensorBoard-Logdaten.

Die Analyse der Episodenlänge zeigte eine klare Zweiteilung im Verhalten der aufgezeichneten Modelle. Die kooperative Baseline verlief vom Start bis zum Ende der zehn Millionen Schritte fast ununterbrochen und konstant bei exakt 1000 Schritten. Auch in den univariaten Rohdaten waren Abweichungen nach unten äußerst selten und von minimaler Ausprägung. Das Modell nutzte die verfügbare Episodenzeit somit durchgehend vollständig aus.

Das Curriculum-Team verhielt sich weitestgehend identisch zur Baseline. Zu Beginn des Trainings (0 bis 1 Million Schritte) war ein kurzer Einbruch in den Rohdaten zu verzeichnen, der die geglättete Kurve temporär auf ca. 950 Schritte sinken ließ. Ab der 1-Millionen-Marke verharrte jedoch auch diese Kurve konstant bei 1000 Schritten. Eine singuläre Ausnahme bildete ein isolierter Spike bei ca. 3,3 Millionen Schritten, bei dem die Rohdaten auf annähernd 600 Schritte fielen. In allen anderen Phasen schlug die Metrik am Maximum an.

Der Ego-Ansatz wies als einziges Modell eine hohe Dynamik in der Episodenlänge auf. Bereits in den ersten zwei Millionen Schritten sank der gleitende Mittelwert auf 850 bis 900 Schritte, wobei die Rohdaten Abschlüsse bei 300 bis 500 Schritten dokumentierten. Nach

einer temporären Rückkehr an die 1000er-Marke verzeichnete die Kurve zwischen vier und sechs Millionen Schritten einen erneuten, starken Einbruch auf ein lokales Minimum von ca. 800 Schritten. Im letzten Drittel des Trainings stieg die Kurve wieder in den Bereich von 950 bis 990 Schritten an. Die konstant hohe Varianz in den Rohdaten zeigte, dass Episoden hier regelmäßig vor dem Erreichen des Time-Outs beendet wurden.

Die Episodenlänge erwies sich damit als komplementäre Metrik zum extrinsischen Reward: Während der Reward die Qualität der erlernten Strategie abbildet, quantifiziert die Episodenlänge deren operative Effizienz. Das dauerhafte Anschlagen der kooperativen Modelle am Zeitlimit von 1000 Schritten war dabei nicht als Zeichen hoher Systemauslastung zu interpretieren, sondern als Symptom einer ineffizienten Aufgabenlösung, die Agenten füllten die verfügbare Zeit aus, ohne die Aufgabe zu einem vorzeitigen Abschluss zu bringen. Im Kontrast dazu dokumentierten die Einbrüche der Episodenlänge im Ego-Modell tatsächliche Aufgabenabschlüsse vor dem Zeitlimit, was eine höhere operative Effizienz in bestimmten Trainingsphasen anzeigte. Die Tatsache, dass auch der Ego-Ansatz gegen Ende des Trainings wieder an das Zeitlimit heranrückte, deutete darauf hin, dass die Agenten in der finalen Trainingsphase zwar eine stabilere, aber auch weniger aggressive Strategie konsolidierten, ein Phänomen, das mit der gleichzeitig beobachteten Reduktion der Policy-Entropie konsistent war.

4.1.3 Netzwerkstabilität und Konvergenz (Policy Entropy)

Zur Evaluation der Netzwerkstabilität wird abschließend die Entropie der Policy (*Policy Entropy*) herangezogen [35]. Ein hoher Wert spricht für eine stochastische Aktionswahl, während ein kontinuierlich sinkender Wert eine deterministischere Verhaltensweise und die Konvergenz des Modells anzeigt. Der Verlauf dieser Metrik ist in Fig. 5 abgebildet.

Alle drei Architekturen starteten initial mit einer ausgeprägten Entropie von geringfügig unter 2,0. Innerhalb der ersten Million Schritte verzeichneten alle Graphen den RL-typischen, steilen Abfall, bevor sich die Verläufe differenzierten.

Der Ego-Ansatz verzeichnete den raschesten Entropieabfall. Nach einem steilen initialen Sinken auf ca. 0,40 flachte die Kurve zunächst ab. Zur Mitte des Trainings zeigte sich ein kurzer, temporärer Anstieg auf annähernd 0,30. Direkt im Anschluss fiel die Entropie auf ein globales Minimum von ca. 0,15 ab und pendelte sich für das verbleibende Trainings-

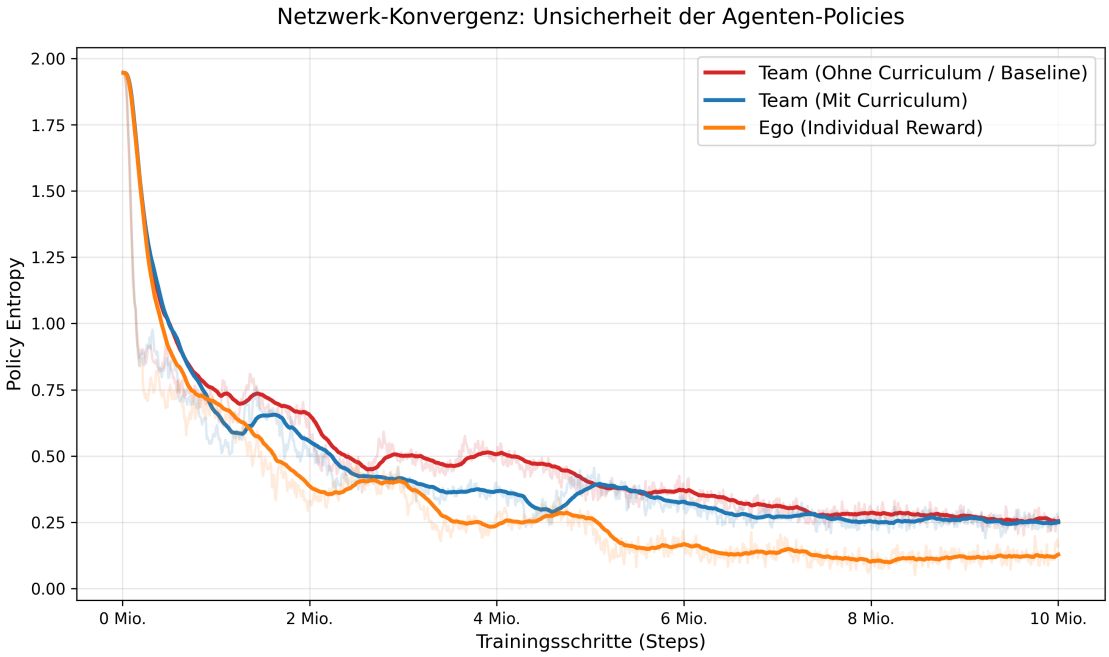

Fig. 5: Verlauf der Policy Entropy zur Beurteilung der Netzwerkstabilität. Eigene Darstellung auf Basis der TensorBoard-Logdaten.

drittel dauerhaft und bemerkenswert stabil auf dem niedrigsten gemessenen Wert zwischen 0,10 und 0,12 ein.

Der Entropieverlauf des Curriculum-Setups sank anfangs langsamer als beim Ego-Ansatz, jedoch stetiger als bei der Baseline. Über das mittlere Drittel des Trainings fiel der Wert kontinuierlich und gleichmäßig von ca. 0,55 auf annähernd 0,30 ab. In der finalen Trainingsphase flachte die Kurve weiter ab und stabilisierte sich letztlich bei einem Endwert von ca. 0,25.

Die unstrukturierte Baseline verzeichnete die am längsten anhaltenden hohen Entropiewerte. Über die gesamte erste Trainingshälfte verharrte die Kurve mit leichten Schwankungen auf einem Plateau um 0,50. Ein signifikanter, kontinuierlicher Abwärtstrend begann erst ab der 6-Millionen-Marke. Gegen Ende der Simulation näherte sich die Baseline der Curriculum-Kurve an und konvergierte auf einem vergleichbaren Niveau zwischen 0,25 und 0,28. Beide kooperativen Modelle schlossen das Training somit mit einer messbar höheren Rest-Entropie ab als der kompetitive Ansatz.

Die Policy-Entropie lieferte damit den algorithmischen Erklärungsrahmen für die in den vorherigen Metriken beobachteten Unterschiede. Die rasche Konvergenz des Ego-Modells

auf niedrige Entropiewerte spiegelte die Tatsache wider, dass das scharfe individuelle Belohnungssignal dem Agenten eine eindeutige Lernrichtung vorgab: Die Kausalität zwischen der eigenen Aktion und der erhaltenen Belohnung war direkt und minimal verrauscht, was die Konsolidierung einer deterministischen Policy begünstigte. Die kooperativen Modelle hingegen operierten unter einem verwässerten Belohnungssignal, das keine eindeutige Präferenz für einzelne Aktionen erzeugte. Das neuronale Netz behielt daher eine höhere Unsicherheit in der Aktionswahl bei, formal ausgedrückt als höhere Restentropie, da unterschiedliche Verhaltensweisen im Hinblick auf das globale Belohnungssignal ähnlich bewertet wurden. Der Befund, dass beide kooperativen Modelle trotz unterschiedlicher Curriculum-Bedingungen auf nahezu identische Entropieniveaus konvergierten, legte nahe, dass die Belohnungsstruktur den dominanten Einflussfaktor auf die Netzwerkstabilität darstellte, nicht die Aufgabenprogression.

Die drei betrachteten Metriken konvergierten damit zu einem konsistenten Gesamtbild. Der extrinsische Reward belegte die Lernüberlegenheit des Ego-Ansatzes hinsichtlich der absoluten Aufgabenerfüllung, die Episodenlänge quantifizierte die operative Effizienz und lokalisierte die Phasen tatsächlicher Aufgabenabschlüsse vor dem Zeitlimit, und die Policy-Entropie lieferte den algorithmischen Erklärungsrahmen für beide Befunde: Ein scharfes, rauschfreies Belohnungssignal erzeugt eine schnellere und tiefere Konvergenz der Policy. Die kooperativen Modelle blieben in allen drei Dimensionen hinter dem Ego-Ansatz zurück, wobei das Team-Curriculum trotz initialer Vorteile durch strukturierte Aufgabenprogression am Ende unter der unstrukturierten Team-Baseline verblieb. Dieses kontraintuitive Ergebnis verlangt eine gesonderte Interpretation, die in Abschnitt 5.1 erfolgt. Die quantitative Analyse liefert damit die metrische Grundlage für die anschließende qualitative Betrachtung, welche das beobachtete Verhalten in der Simulationsumgebung kausal erklärt.

4.2 Qualitative Verhaltensanalyse

Zur Kontextualisierung der quantitativen Trainingsdaten (Kapitel 4.1) erfolgte eine visuelle Analyse des finalen Verhaltens der trainierten Modelle in der Unity-Simulationsumgebung. Ziel der qualitativen Betrachtung war es, die spezifischen Strategien, Ineffizienzen und Gruppendynamiken zu identifizieren, die durch die unterschiedlichen Belohnungsstrukturen (Team vs. Ego) in Kombination mit variierenden Blockmassen hervorgerufen wurden.

4.2.1 Verhalten der kooperativen Modelle (Team Baseline & Curriculum)

Die visuelle Analyse der Modelle mit geteilter Teambelohnung (Shared Reward) stützte die Interpretation der quantitativen Metriken. Obwohl die Agenten grundsätzlich die Fähigkeit erlernten, mit den Blöcken zu interagieren, scheiterten sie an der zielgerichteten Erfüllung der Aufgabe. Es ließen sich zwei primäre Fehlverhaltensmuster beobachten. Das erste ist eine fehlende räumliche Assoziation (*Interference*): Die Agenten zeigten keinen verlässlichen Drang, die Blöcke in die vorgesehene Zielzone zu manövrieren, sondern schoben sie häufig planlos gegen die Begrenzungswände der Arena oder blockierten sich gegenseitig in unkoordinierten Verkehrsstaus (Fig. 6). Dies verdeutlichte, dass das neuronale Netz die grüne Zielzone nicht erfolgreich als Auslöser für die Teambelohnung erkannte. Das zweite Muster war das „Lazy-Agent-Phänomen“: Besonders auffällig war das wiederkehrende passive Verhalten einzelner Agenten. Oftmals begab sich ein Agent ohne Block in die Zielzone und verharrte dort, während die verbleibenden Agenten mit den Blöcken interagierten. Dieses Verhalten war ein klassisches Symptom des *Credit Assignment Problems* [6]. Wie die qualitativen Beobachtungen in der Simulationsumgebung nahelegten, lernte der Agent hierbei, dass passives Verhalten unter dem globalen Belohnungssignal eine validere Strategie darstellte als die aktive, potenziell fehlerhafte Blockinteraktion.

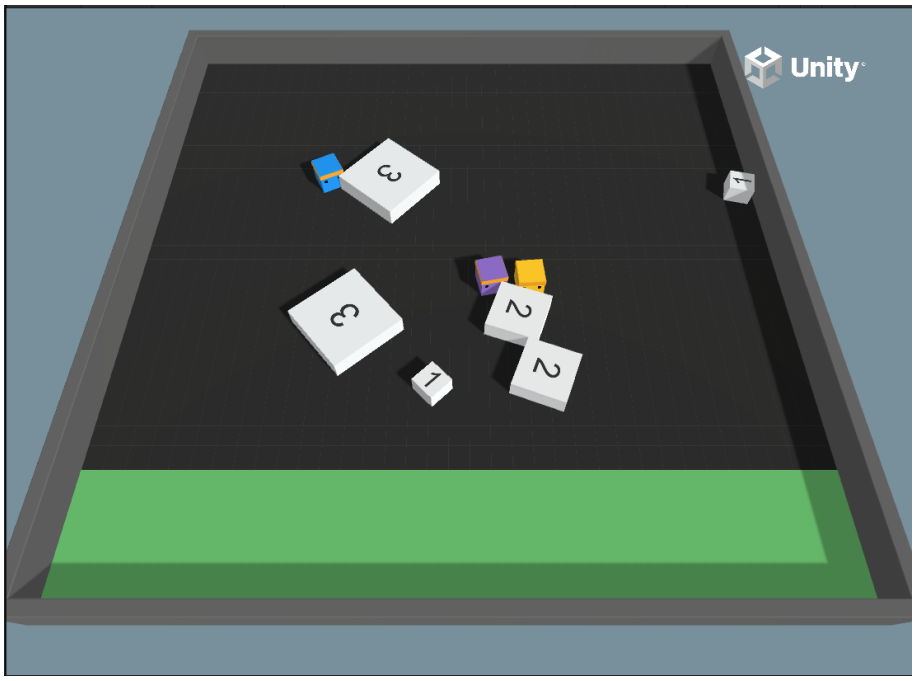

Fig. 6: Typisches Fehlverhalten im Team-Modell: gegenseitige Blockaden und unkoordinierte Blockbewegungen fernab der Zielzone. Eigene Darstellung.

4.2.2 Verhalten des individuellen Modells (Ego)

Das Modell mit egoistischer Belohnungsstruktur (Ego) wies ein fundamental anderes Verhalten auf. Die präzise individuelle Belohnung für den Agenten mit der geringsten Distanz zum versenkten Block löste das *Credit Assignment Problem* effektiv auf, führte jedoch zu einer hochgradig kompetitiven und teilweise dysfunktionalen Gruppendynamik. Einerseits agierten die Agenten stark zielgerichtet, was zu einer deutlich schnelleren Leerung der Arena führte; diese Zielstrebigkeit äußerte sich jedoch oft in destruktiver Ressourcenkonkurrenz. Bei kleinen Blöcken (Curriculum-Level 0), die von einem einzelnen Agenten bewegt werden konnten, beanspruchten häufig alle drei Agenten gleichzeitig dieselbe Ressource, was zu gegenseitigem Wegdrücken und einem ineffizienten Kampf führte. Bei mittleren Blöcken (Curriculum-Level 1), für die zwei Agenten physikalisch erforderlich waren, blieb die chaotische Mehrfachkonkurrenz bestehen: Da alle drei Agenten gleichzeitig auf denselben Block zudrängten, entstanden unkoordinierte Schubkräfte in wechselnden Richtungen, die den Transportprozess verlangsamten oder zeitweise blockierten. Diese Muster erklärten die extremen Schwankungen in der Episodenlänge aus der quantitativen Analyse. Andererseits zeigte sich das wissenschaftlich relevanteste Phänomen bei der Interaktion mit den schwersten Blöcken (Curriculum-Level 2): Da ein einzelner Agent physikalisch nicht in der Lage ist, diese Masse allein rechtzeitig zu bewegen, entwickelten die Agenten eine Form der emergenten Kooperation. Zwei oder alle drei Agenten bündelten ihre Kräfte und schoben den Block gemeinsam in Richtung der Zielzone (Fig. 7). Diese Kooperation basierte jedoch nicht auf einem programmierten Teamgedanken, sondern auf purem Eigennutz: Jeder Agent unterstützte den Schubprozess, um im Moment der Zielerreichung die optimale Position für den Erhalt der alleinigen Belohnung zu haben.

Zusammenfassend zeigte die qualitative Beobachtung, dass die egoistische Belohnungsstruktur die eigentliche Aufgabenerfüllung erzwang, sich dies jedoch durch ein hohes Maß an Konkurrenz erkaufte. Der kooperative Ansatz scheiterte hingegen in dieser spezifischen, komplexen Umgebung an der fehlenden Nachvollziehbarkeit der eigenen Handlungen für den Gesamterfolg.

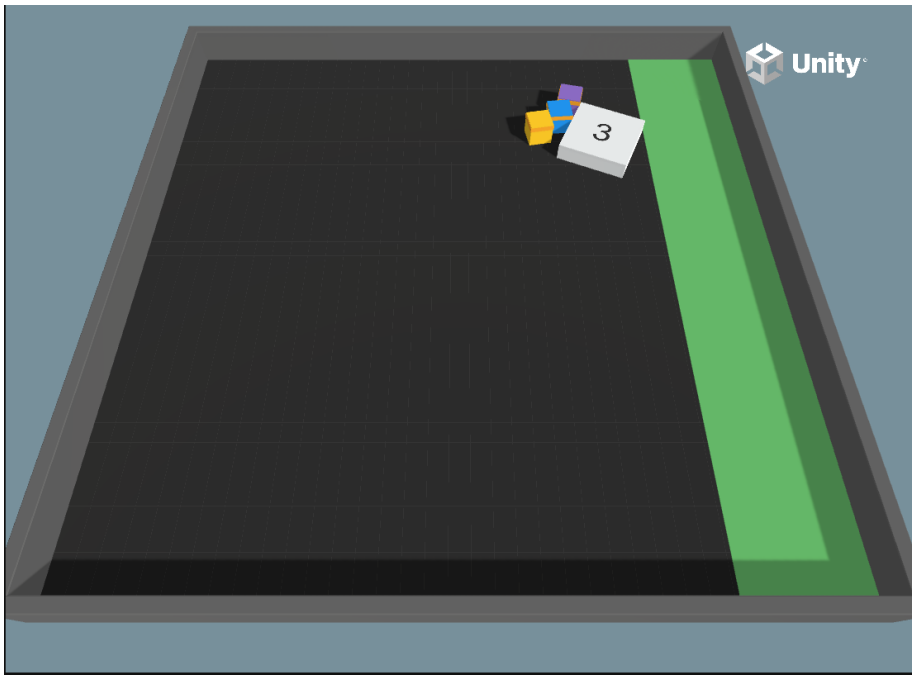

Fig. 7: Emergente Kooperation im Ego-Modell beim gemeinsamen Schieben eines schweren Blocks. Eigene Darstellung.

5 Diskussion

Die quantitativen Metriken und qualitativen Verhaltensbeobachtungen aus Kapitel 4 liefern für sich genommen noch keine Antwort auf die Forschungsfrage – sie verlangen eine Einordnung in den theoretischen Rahmen, vor dem ihre Bedeutung sichtbar wird. Abschnitt 5.1 interpretiert die drei zentralen Befunde: das Scheitern der kooperativen Baseline, die emergente Kooperation unter egoistischer Belohnung und die kontraintuitive Leistungseinbuße des Curriculum-Ansatzes. Abschnitt 5.2 benennt die architektonischen und methodischen Grenzen, innerhalb derer diese Schlussfolgerungen Gültigkeit beanspruchen. *Value Decomposition*-Methoden und kontrafaktische Baseline-Ansätze adressieren das *Credit Assignment Problem* durch die mathematische Isolierung individueller Agentenbeiträge [28], [55], waren im vorliegenden IPPO-basierten Versuchsdesign jedoch bewusst nicht implementiert, um die Auswirkungen der Belohnungsstruktur in ihrer unkompensierten Form sichtbar zu machen.

5.1 Interpretation der Ergebnisse

Die Ergebnisse der Simulation zeichnen ein differenziertes Bild hinsichtlich der gestellten Forschungsfrage. Der direkte Vergleich der Belohnungsstrukturen in der modifizierten Push-Block-Arena zeigt auf, dass weder das streng kooperative noch das rein kompetitive Modell eine durchgehend effiziente Aufgabenlösung ohne unerwünschte Nebeneffekte erreicht. Vielmehr provozieren beide Ansätze spezifische, aus der Literatur bekannte Dysfunktionen im Bereich des Multi-Agent Reinforcement Learning (MARL).

5.1.1 Das Scheitern der kooperativen Baseline

Obwohl die geteilte Teambelohnung theoretisch auf eine harmonische Zusammenarbeit abzielte, belegen die quantitativen Metriken das Gegenteil. Die Stagnation des *Extrinsic Rewards* bei einem Wert von circa 3,5 bis 4,0 (vgl. Fig. 3) sowie die dauerhafte Ausreizung der maximalen Episodenlänge von 1000 Schritten (vgl. Fig. 4) zeigen, dass die Agenten die Aufgabe nicht effizient lösen konnten.

Die qualitative Beobachtung offenbarte hierbei das *Lazy-Agent*-Phänomen: Einzelne Agenten verharrten passiv in der Zielzone. In der aktuellen Forschung zu dezentralem Reinforcement Learning wird dies als direktes Symptom des ungelösten *Credit Assignment Problems* beschrieben [27], [6]. Da die Belohnung global für das gesamte Team ausgeschüttet wird, gehen individuelle, zielführende Aktionen im Rauschen der Gruppenleistung unter. Die Agenten erlernen keine Verknüpfung zwischen dem physischen Schieben eines Blocks und dem Teamerfolg. Aktuelle Analysen zu voll dezentralen Systemen bestätigen diese Koordinations-Paradoxie, bei der erzwungene globale Belohnungen in unabhängigen PPO-Architekturen die Aufgabenerfüllung massiv hemmen [7].

5.1.2 Zielerreichung durch Egoismus und emergente Kooperation

Die egoistische Belohnungsstruktur (Ego) erzeugte hingegen eine signifikant höhere Lerneffektivität mit Spitzenwerten von bis zu 5,0 beim Reward und der schnellsten Netzwerk-Konvergenz bei einer Policy Entropy von 0,10 (vgl. Fig. 5). Die individuelle Belohnung für den Agenten mit der geringsten Distanz zum Block löste die Zuordnungsproblematik auf und erzwang die physische Interaktion mit den Objekten.

Jedoch offenbarte die qualitative Analyse (vgl. Abschnitt 4.2), dass diese Effizienz durch

eine hochgradig dysfunktionale Ressourcenkonkurrenz erkauft wurde. Das Modell wies eine extreme Volatilität in der Episodenlänge auf (mit Einbrüchen auf bis zu 300 Schritte), was auf das unkoordinierte, gegenseitige Wegdrücken bei Blöcken der unteren Gewichtsklassen zurückzuführen war. Die Belohnungsstruktur begünstigte demnach keine stabile koordinierte Strategie, sondern Ressourcenkonkurrenz unter den Agenten.

Dieses Befundmuster weist eine strukturelle Ähnlichkeit mit dem in der spieltheoretischen Literatur beschriebenen Konzept der *Tragedy of the Commons* auf [56]: Analog zu Situationen, in denen rational eigennützige Akteure gemeinsame Ressourcen übernutzen und dadurch das kollektive Ergebnis verschlechtern, konkurrierten im vorliegenden Experiment alle drei Agenten simultan um dieselben Blöcke, was den Transportprozess verlangsamte anstatt ihn zu beschleunigen. Dieses Dilemma findet im MARL-Kontext seine direkte Entsprechung in Szenarien mit geteilten Ressourcen [42]. Im vorliegenden Experiment manifestierte sich diese Dynamik besonders deutlich bei kleinen Blöcken (Klasse 0): Da alle drei Agenten simultan denselben Block als Belohnungsquelle identifizierten und gleichzeitig auf ihn zudrängten, verhinderten sie gegenseitig den effizienten Abschluss der Aktion. Bei mittleren Blöcken (Klasse 1), für die zwei Agenten physikalisch erforderlich waren, blieb die chaotische Mehrfachkonkurrenz bestehen: Da alle drei Agenten gleichzeitig auf denselben Block zudrängten, entstanden unkoordinierte Schubkräfte in wechselnden Richtungen, die den Transportprozess verlangsamten oder zeitweise blockierten.

Die physikalische Massedifferenzierung der Blöcke erzeugte damit keine binäre Kooperationsanforderung, sondern eine abgestufte Kooperationslandschaft: Klasse 0 erzwang reine Ressourcenkonkurrenz, Klasse 1 erzwang eine gemischte Dynamik aus partieller Kooperation und verbleibendem Wettbewerb, und Klasse 2 erzwang vollständige Kooperation aller drei Agenten. Dieser Befund verdeutlicht, dass die physikalische Umgebungsstruktur als eigenständige Steuerungsvariable für das emergente Gruppenverhalten wirkt, die unabhängig von der gewählten Belohnungsstruktur unterschiedliche Koordinationsmodi in verschiedenen Aufgabenphasen induziert [44].

Erst die physikalischen Restriktionen der schweren Blöcke (Klasse 2) durchbrachen das Konkurrenzprinzip vollständig. Die Masse der Level-2-Blöcke fungierte dabei als implizite Koordinationsbedingung: Da ein einzelner Agent die Aufgabe physikalisch nicht lösen konnte, erzwang die Umgebungsstruktur selbst eine Konvergenz der Handlungs-

ziele, ohne dass ein expliziter Koordinationsmechanismus erforderlich gewesen wäre. Dies korrespondiert mit dem Befund, dass emergente Kooperation unter geeigneten Umgebungsbedingungen auch ohne explizite Kommunikationskanäle entstehen kann [46]. Die physikalische Umgebungsstruktur übernahm damit die Funktion einer impliziten Koordinationsschicht, die das kompetitive Belohnungssignal in kooperatives Verhalten transformierte.

Das wissenschaftlich relevanteste Resultat zeigte sich folglich in der Interaktion mit den schwersten Blöcken (Curriculum-Level 2). Da die physikalische Masse hier eine Einzelaktion unmöglich machte, bündelten die Agenten ihre Kräfte (vgl. Fig. 7). Diese Verhaltensweise ist jedoch nicht als programmierte Kooperation zu werten, sondern als *emergente Kooperation* aus reinem Eigennutz: Jeder Agent schob mit, um im Moment der Zielerreichung punktgenau die Belohnung für sich zu beanspruchen. Dieses emergente, zweckgebundene Verhalten durch Wettbewerbsdruck korreliert stark mit Phänomenen, die in großskaligen MARL-Experimenten beobachtet wurden [45]. Dieser Befund stützt die Hypothese, dass physikalische Restriktionen in einer Multi-Agenten-Umgebung kooperatives Verhalten unter bestimmten Bedingungen zuverlässiger hervorrufen können als unstrukturierte Teambelohnungen [44].

5.1.3 Leistungseinbußen durch Curriculum-Overfitting

Ein kontraintuitives Resultat der quantitativen Auswertung ist die Tatsache, dass der kooperative Ansatz mit Curriculum (ca. 3,5 Punkte) am Ende des Trainings messbar schlechter abschnitt als die unstrukturierte Team-Baseline (ca. 4,0 Punkte). Zwar ermöglichte die schrittweise Einführung der Blöcke dem Team-Modell überhaupt erst das Erlernen einer grundlegenden Interaktion und durchbrach die anfängliche Stagnation, jedoch brachte diese Methodik spezifische Nachteile mit sich.

Ein ausgedehntes Training auf simplifizierten Vorstufen kann im Reinforcement Learning zu einer Überanpassung (*Overfitting*) an frühe Lektionen führen [50], [51]. Die Agenten optimierten ihre Policy zunächst intensiv auf den Umgang mit Blöcken der unteren Gewichtsklassen (Lektion 0 mit kleinen und Lektion 1 mit mittleren Blöcken). Beim finalen Übergang in das End-Szenario (Lektion 2) mit den schweren, massereichen Blöcken ließ sich diese zu simpel erlernte Strategie nicht verlustfrei generalisieren. Erschwerend kommt

hinzu, dass dem Curriculum-Modell durch die vorab zu absolvierenden Vorstufen eine signifikant kürzere effektive Lernzeit im eigentlichen Ziel-Szenario zur Verfügung stand als der Baseline, welche die vollen zehn Millionen Schritte ausschließlich in der finalen Umgebung trainierte.

Ein weiterer Erklärungsansatz für das schlechtere Abschneiden des kooperativen Curriculum-Modells liegt in der Interaktion zwischen Aufgabenprogression und Belohnungsstruktur. Während das Curriculum für den Ego-Ansatz als wirksamer Komplexitätsreduzierer fungierte, da das scharfe individuelle Signal auch in vereinfachten Vorstufen klare Kausalitäten zwischen Aktion und Belohnung herstellt, entfaltete es für das Team-Modell eine paradoxe Wirkung: Die Vorstufen waren ausreichend simpel, um das Credit Assignment Problem temporär abzumildern und grundlegende Interaktionsmuster zu erlernen. Beim Übergang in die finale, komplexere Aufgabenstufe wurde das Zuordnungsproblem jedoch in seiner vollen Stärke wieder aktiv, da die höhere Anzahl gleichzeitig aktiver Blöcke und Agenten das kollektive Signal stärker verrauschte als in den Vorstufen. Die im Curriculum erlernten vereinfachten Strategien erwiesen sich damit nicht nur als unzureichend für die finale Aufgabe, sondern aktiv als Hemmnis: Die Agenten hatten gelernt, eine Belohnungsfunktion zu maximieren, die in der finalen Stufe eine qualitativ andere Kausalstruktur aufwies als in den Vorstufen. Dieser Befund ergänzt die bestehende Curriculum-Learning-Literatur [50], [51] um eine spezifische Warnung: In kooperativen MARL-Systemen mit globalen Belohnungen kann ein schlecht kalibriertes Curriculum nicht nur unwirksam sein, sondern aktiv zu einer Überanpassung an Vorstufen führen, die in der finalen Aufgabenstufe kontraproduktiv ist. Die Schwellenwertgestaltung des Curriculums sollte daher nicht nur die Aufgabenkomplexität, sondern auch die Belohnungsstruktur und deren Wechselwirkung mit der Agentenanzahl berücksichtigen.

5.2 Limitationen der Simulation

Die aussagekräftigen Ergebnisse dieser Arbeit müssen im Kontext der gewählten methodischen und technischen Rahmenbedingungen betrachtet werden. Die beobachteten Dysfunktionen, insbesondere das Scheitern des kooperativen Ansatzes, sind nicht zwingend auf die Unlösbarkeit der Aufgabe zurückzuführen, sondern korrelieren stark mit den architektonischen Restriktionen der verwendeten Algorithmen und der physischen Beschaffenheit der Simulationsumgebung.

5.2.1 Architektonische Grenzen (Independent PPO)

Die gravierendste methodische Einschränkung stellt die Verwendung von *Independent PPO* (IPPO) dar. In der vorliegenden Architektur agierte jeder der drei Agenten als isolierte Lerninstanz, die lediglich auf lokale Beobachtungen (Raycasts) zurückgreifen konnte. Wie aktuelle Überblicksstudien zu voll dezentralem MARL betonen [6], führt dieses Setup bei komplexen kooperativen Aufgaben zu einer massiven Nicht-Stationarität der Umgebung (Non-Stationarity). Da die Agenten ihre Policies gleichzeitig aktualisieren, ändert sich die Dynamik der Umgebung aus Sicht eines einzelnen Agenten kontinuierlich.

Diese Limitierung wurde bewusst in Kauf genommen, um die fundamentalen Auswirkungen der Belohnungsfunktion (Team vs. Ego) isoliert untersuchen zu können. Hätte das System auf einen zentralisierten Critic-Ansatz (z.B. MAPPO [39]) zurückgegriffen, der während des Trainings den globalen Zustand der Arena überblickt, wäre das *Credit Assignment Problem* voraussichtlich algorithmisch abgemildert worden. Der gewählte IPPO-Ansatz diente somit als analytischer Härtetest, der die Schwächen unstrukturierter Teambelohnungen in voll dezentralen Architekturen klar hervortreten ließ [6].

Eine weiterführende methodische Überlegung betrifft die Übertragbarkeit der gewählten Architektur auf Szenarien mit mehr als drei Agenten. Die in dieser Arbeit beobachteten Dysfunktionen sind strukturell skalierungsabhängig. Dies gilt insbesondere für das *Lazy-Agent*-Phänomen und die destruktive Ressourcenkonkurrenz: Mit zunehmender Agentenzahl verstärkt sich das Rauschen im globalen Belohnungssignal, da der Beitrag eines einzelnen Agenten zum Gesamterfolg im kollektiven Signal weiter verwässert wird [6]. Gleichzeitig nimmt die Wahrscheinlichkeit unkoordinierter Ressourcenkonflikte mit der Agentenanzahl zu. Eine Replikation des vorliegenden Experiments mit einer variablen Agentenzahl würde wertvolle Hinweise darauf liefern, ob die beobachteten Effekte in ihrer Intensität mit der Systemgröße skalieren oder ob ein Schwellenwert existiert, ab dem qualitativ andere Koordinationsprobleme auftreten.

Darüber hinaus ist die Wahl des diskreten Aktionsraums mit sieben Handlungsoptionen eine Designentscheidung, die das beobachtete Verhalten potenziell beeinflusst hat. Ein kontinuierlicher Aktionsraum, der feinkörnigere Kraftvektoren erlaubt, würde den Agenten mehr Flexibilität in der physischen Interaktion mit den Blöcken bieten und könnte insbesondere die Präzision beim Positionieren in unmittelbarer Blocknähe verbessern. Die

Verwendung des diskreten Aktionsraums war durch die Kompatibilität mit dem Unity-ML-Agents-Framework motiviert, stellt jedoch eine mögliche Quelle zusätzlicher Ineffizienz dar, da optimale Aktionen nur annäherungsweise aus dem verfügbaren Diskretisierungsgitter ausgewählt werden können.

5.2.2 Räumliche Interferenz und Volatilität

Eine weitere Limitierung ergibt sich aus der hohen Dichte an physischen Hindernissen in der modifizierten Push-Block-Arena. Durch die parallele Existenz von drei Agenten und sechs Blöcken auf engem Raum kam es zu einer extremen räumlichen Interferenz. Ein analoges Phänomen wurde in agentenbasierten Evolutionssimulationen als Quelle erhöhter Systemvolatilität beschrieben [57].

Dieser theoretische Ansatz der erhöhten Systemvolatilität spiegelte sich in der Simulation praktisch wider: Die Agenten manipulierten die Umgebung derart schnell, dass Aktionen, die einen Bruchteil einer Sekunde zuvor noch optimal erschienen, durch die unvorhersehbare Bewegung eines Mitagenten plötzlich zu einer physischen Kollision oder einem Deadlock führten. Diese indirekte Interferenz durch Mitakteure ohne explizite Kommunikationskanäle (wie das Senden von Intentions-Signalen) führte insbesondere im Ego-Modell zu den in der quantitativen Analyse dokumentierten starken Schwankungen der Episodenlänge. Zukünftige Iterationen der Simulationsumgebung könnten durch den Einsatz von direkten Kommunikationsvektoren zwischen den Agenten diese physikalische Volatilität reduzieren.

Die beobachtete räumliche Interferenz verweist zudem auf eine grundsätzliche Spannung zwischen der Dichte des Szenarios und der Reichweite der Raycast-Sensoren. Mit sieben Strahlen und einem Sichtfeld von 70 Grad verfügte jeder Agent über eine ausreichende Nahwahrnehmung für die unmittelbare Hinderniserkennung, jedoch über keine Möglichkeit, die Bewegungsintentionen seiner Mitagenten vorauszuschätzen. Da die *Decision Period* auf fünf Engine-Schritte festgelegt war, reagierten die Agenten mit einer strukturellen Verzögerung auf die sich schnell verändernde Umgebung. In hochdynamischen Phasen, vor allem wenn mehrere Agenten denselben Block gleichzeitig in unterschiedliche Richtungen schoben, war diese Reaktionslatenz ausreichend, um korrektive Handlungen systematisch zu unterbinden. Eine kürzere *Decision Period* oder der Einsatz von prädiktiven Sensor-

modulen, die die wahrscheinliche Bewegungsrichtung von Mitagenten schätzen, könnten in zukünftigen Arbeiten die Kollisionshäufigkeit reduzieren und die Nettolerneffizienz steigern.

Abschließend ist anzumerken, dass die in dieser Arbeit verwendete Metrik des extrinsischen Rewards ausschließlich den akkumulierten Punktestand erfasst, nicht jedoch die räumliche Effizienz der Agentenpfade. Agenten, die Blöcke auf längeren, weniger direkten Routen ins Ziel befördern, erzielen denselben Reward wie solche, die optimale Direktpfade nutzen. Eine Ergänzung der Evaluationsmetrik um pfadbasierte Effizienzmaße, etwa die mittlere euklidische Distanz zwischen Startposition und Zielzone bezogen auf die tatsächlich zurückgelegte Strecke, würde eine differenziertere Beurteilung der erlernten Strategien ermöglichen und könnte subtile Qualitätsunterschiede zwischen den drei Trainingsansätzen sichtbar machen, die im reinen Reward-Signal nicht abgebildet werden.

Schließlich ist die fehlende statistische Absicherung der Ergebnisse über mehrere unabhängige Trainingsläufe als methodische Limitation anzuerkennen. Jeder der drei Trainingsansätze wurde in dieser Arbeit einmalig über zehn Millionen Schritte trainiert. Unterschiede in der Initialisierung der neuronalen Netze, der zufälligen Startkonfiguration der Blöcke und des stochastischen Trainingsprozesses können die absoluten Metrikwerte einzelner Läufe beeinflussen. Für eine robuste statistische Aussage wären mehrere unabhängige Wiederholungsläufe pro Konfiguration erforderlich, aus denen Mittelwerte und Standardabweichungen der Lernkurven berechnet werden könnten. Dieser Ansatz ist in der RL-Forschung als Best Practice anerkannt [58] und wurde in der vorliegenden Arbeit aus Zeitgründen bei einer durchschnittlichen Trainingsdauer von sieben Stunden pro Lauf nicht realisiert. Die qualitative Konsistenz der beobachteten Verhaltensmuster, insbesondere das Lazy-Agent-Phänomen und die emergente Kooperation, deutet auf eine mögliche Robustheit der Kernbefunde hin, ersetzt jedoch keine statistische Absicherung durch unabhängige Wiederholungsläufe. Eine formale statistische Absicherung bleibt gleichwohl ein notwendiger Schritt für weiterführende Publikationen.

6 Fazit und Ausblick

Die empirischen Befunde dieser Arbeit widersprechen der intuitiven Erwartung, dass kooperative Belohnungsstrukturen in kooperativen Aufgaben überlegen sind. Abschnitt 6.1

beantwortet die Forschungsfrage auf Basis der Gesamtschau der Ergebnisse und leitet drei übergreifende Prinzipien ab, die über das konkrete Experiment hinaus Gültigkeit beanspruchen. Abschnitt 6.2 benennt konkrete Forschungsanknüpfungen und diskutiert die Übertragbarkeit der Befunde auf reale Robotik-Anwendungen.

6.1 Zusammenfassung der Arbeit

Diese Arbeit untersuchte die Auswirkungen kooperativer und kompetitiver Belohnungsstrukturen auf das emergente Verhalten in einem dezentralen Multi-Agenten-System. Anhand einer modifizierten Push-Block-Umgebung in Unity ML-Agents wurde empirisch evaluiert, welches Paradigma unter der Verwendung von *Independent Proximal Policy Optimization* (IPPO) zu einer höheren Aufgabeneffizienz führt.

Die Ergebnisse des durchgeführten Experiments deuten darauf hin, dass eine unstrukturierte geteilte Teambelohnung in diesem voll dezentralen IPPO-Setup keine effiziente Aufgabenlösung hervorbrachte. Die Agenten scheiterten am *Credit Assignment Problem*, was in passiven *Lazy-Agent*-Verhaltensweisen resultierte. Das globale Belohnungssignal reichte nicht aus, um den Agenten die Kausalität zwischen individueller physischer Anstrengung und dem Teamerfolg zu vermitteln.

Im Gegensatz dazu erwies sich die egoistische Belohnungsstruktur, bei der ausschließlich der Agent mit der geringsten Distanz zum Ziel belohnt wurde, als maßgeblich effektiver im Hinblick auf die Lösungsrate und die Konvergenzgeschwindigkeit. Obwohl dieser Ansatz stark kompetitive Dysfunktionen wie gegenseitige Blockaden bei Blöcken der unteren Gewichtsklasse hervorrief, erzwang er bei schweren Blöcken eine *emergente Kooperation*. Die Agenten kooperierten nicht aus einem programmierten Teamgedanken heraus, sondern aus purer individueller Nutzenmaximierung. Für das untersuchte IPPO-basierte Push-Block-Szenario lässt sich die Forschungsfrage dahingehend beantworten, dass die individuelle Belohnungsstruktur zu höheren Reward-Werten, schnellerer Konvergenz und phasenweise kürzeren Episoden führte als die getesteten Formen geteilter Teambelohnung, sofern die daraus resultierende Ressourcenkonkurrenz akzeptiert wird.

Betrachtet man die drei Trainingsansätze in ihrer Gesamtheit, lassen sich aus den Ergebnissen drei übergreifende Prinzipien ableiten, die über das konkrete Experiment hinaus Gültigkeit beanspruchen. Erstens bestimmt die Struktur des Belohnungssignals, nicht die Kom-

plexität des Algorithmus, die grundlegende Lernrichtung in dezentralen Mehrfachagenten-Systemen. Das Credit Assignment Problem ist kein algorithmisches Randproblem, sondern ein zentrales Designhindernis, das bereits bei drei Agenten und einer überschaubaren Aufgabe vollständig wirksam wird. Zweitens kann strukturelle Komplexitätsreduktion durch Curriculum Learning den initialen Lernstillstand zwar durchbrechen, aber nicht kompensieren, wenn das Belohnungssignal selbst strukturell unzureichend ist. Der kooperative Curriculum-Ansatz scheiterte letztlich nicht an der Aufgabenschwierigkeit, sondern an der Tatsache, dass das verwässerte Teambelohnungssignal auch nach erfolgreichem Durchlaufen der Vorstufen keine ausreichend starke individuelle Handlungsrückmeldung lieferte. Drittens können physikalische Umgebungsrestriktionen als implizite Koordinationsmechanismen wirken und kooperatives Verhalten unter kompetitiven Bedingungen erzwingen, ohne dass eine explizite Kommunikations- oder Koordinationsarchitektur erforderlich ist. Dieser Befund ist insofern wissenschaftlich bedeutsam, als er nahelegt, dass das Design der physischen Simulationsumgebung eine eigenständige Steuerungsvariable darstellt, die neben der Belohnungsstruktur und der Algorithmusarchitektur als dritte Dimension der MARL-Systemgestaltung berücksichtigt werden sollte. Die beobachtete emergente Kooperation bei schweren Blöcken entstand nicht primär durch eine kooperative Belohnungsstruktur, sondern durch eine physikalische Aufgabenbedingung, die individuelle Handlungen unzureichend machte und gemeinsames Schieben faktisch erforderlich werden ließ – ein Befund, der für das Design zukünftiger Multi-Agenten-Umgebungen unmittelbare praktische Implikationen hat.

6.2 Zukünftige Forschungsansätze

Basierend auf den methodischen Limitationen dieser Arbeit ergeben sich klare Anknüpfungspunkte für zukünftige Forschungsprojekte. Um das identifizierte *Credit Assignment Problem* bei kooperativen Aufgaben zu überwinden, sollte der isolierte IPPO-Ansatz durch Algorithmen ersetzt werden, die dem Paradigma des *Centralized Training with Decentralized Execution* (CTDE) [26] folgen. Ansätze wie *Counterfactual Multi-Agent Policy Gradients* (COMA) [29] nutzen einen zentralen Critic, der während der Trainingsphase auf globale Zustandsinformationen zugreift und den individuellen Beitrag jedes Agenten zum Teamerfolg mathematisch isoliert. Eine Replikation des vorliegenden Push-Block-Szenarios mit einem CTDE-Algorithmus könnte belegen, ob sich dadurch die Vorteile

beider Welten (effiziente Aufgabenlösung ohne destruktive Ressourcenkonkurrenz) vereinen lassen.

Die Relevanz solcher dezentralen Lösungsansätze beschränkt sich dabei nicht auf virtuelle Simulationen, sondern ist für die reale Robotik von essenzieller Bedeutung. In physischen Umgebungen ist eine zentrale Steuerung aufgrund von Kommunikationslatenzen, begrenzter Bandbreite oder potenziellen Ausfallrisiken oft nicht realisierbar, weshalb physische Agenten zwingend auf Basis lokaler Beobachtungen kooperieren müssen [6], [26]. Die Erkenntnisse zu dezentraler Strategiebildung und emergenter Kooperation lassen sich direkt auf aktuelle Herausforderungen der Automatisierungstechnik übertragen. Praxisnahe Beispiele hierfür sind die Koordination autonomer Ernteroboter, bei denen mehrere Greifarme kollisionsfrei auf engem Raum interagieren müssen [59], oder die Steuerung von unbemannten Wasserfahrzeugen (USVs), die in stark restriktiven, maritimen Umgebungen kooperative Einkreisungsmanöver durchführen [60]. Zukünftige Forschungen sollten daher den Transfer der in Simulationsumgebungen erprobten Belohnungsstrukturen auf physische Hardware-Prototypen fokussieren.

Ein weiterer vielversprechender Forschungsansatz ergibt sich aus der Beobachtung, dass sowohl die rein kooperative als auch die rein kompetitive Belohnungsstruktur spezifische, strukturell bedingte Dysfunktionen hervorriefen. Dies legt nahe, dass hybride Belohnungsarchitekturen, die individuelle und kollektive Anreize in einem konfigurierbaren Verhältnis kombinieren, möglicherweise die Vorteile beider Paradigmen vereinen können, ohne deren jeweilige Nachteile zu übernehmen [43]. Eine solche Hybridstruktur könnte beispielsweise eine primäre individuelle Belohnung für den physisch beitragenden Agenten mit einem sekundären, abgeschwächten kollektiven Bonussignal kombinieren, das alle Agenten für das Teamergebnis mitverantwortlich macht, ohne das individuelle Signal im kollektiven Rauschen zu vergraben. Die Kalibrierung des relativen Gewichts zwischen beiden Anteilen würde dabei ein eigenes Optimierungsproblem darstellen, dessen Lösung von der spezifischen Aufgabenstruktur, der Agentenzahl und der gewählten Algorithmusarchitektur abhinge. Darüber hinaus wäre es wissenschaftlich aufschlussreich, das vorliegende Experiment auf physikalisch heterogenere Umgebungen auszuweiten, also auf Szenarien, in denen nicht nur die Masse der Objekte, sondern auch ihre Form, Reibungseigenschaften oder Interaktionsmechanismen variieren. Solche Szenarien würden den Koordinationsbedarf zwischen den Agenten weiter differenzieren und ermöglichen, die Grenzen des

Tragedy-of-the-Commons-Effekts bei kompetitiven Belohnungsstrukturen systematisch zu kartieren. Die vorliegenden Befunde legen nahe, dass die Belohnungsstruktur als Designparameter im MARL-System systematisch unterschätzt wird: Ihre gezielte Gestaltung kann sich als wirksames Instrument zur Steuerung emergenten kollektiven Verhaltens erweisen [41].

Literatur

[1] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. Cambridge, MA: The MIT Press, 2018.

[2] D. Silver *et al.*, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[3] ——, "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play," *Science*, vol. 362, no. 6419, pp. 1140–1144, 2018.

[4] O. Vinyals *et al.*, "Grandmaster level in StarCraft II using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[5] C. Berner *et al.*, "Dota 2 with large scale deep reinforcement learning," *arXiv preprint arXiv:1912.06680*, 2019.

[6] S. Gronauer and K. Diepold, "Multi-agent deep reinforcement learning: A survey," *Artificial Intelligence Review*, vol. 55, no. 2, pp. 895–943, 2022.

[7] C. S. De Witt *et al.*, "Is independent learning all you need in the StarCraft multi-agent challenge?" *arXiv preprint arXiv:2011.09533*, 2020.

[8] V. François-Lavet, P. Henderson, R. Islam, M. G. Bellemare, and J. Pineau, "An introduction to deep reinforcement learning," *Foundations and Trends in Machine Learning*, vol. 11, no. 3–4, pp. 219–354, 2018.

[9] R. E. Bellman, *Dynamic Programming*. Princeton, NJ: Princeton University Press, 1957.

[10] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3–4, pp. 279–292, 1992.

[11] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 26–38, 2017.

[12] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[13] L.-J. Lin, "Self-improving reactive agents based on reinforcement learning, planning and teaching," *Machine Learning*, vol. 8, no. 3–4, pp. 293–321, 1992.

[14] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine Learning*, vol. 8, no. 3–4, pp. 229–256, 1992.

[15] V. Konda and J. Tsitsiklis, "Actor-critic algorithms," in *Advances in Neural Information Processing Systems*, vol. 12. MIT Press, 1999.

[16] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Advances in Neural Information Processing Systems*, vol. 12. MIT Press, 1999.

[17] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," in *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017, pp. 2778–2787.

[18] L. Busoniu, R. Babuska, and B. de Schutter, "A comprehensive survey of multiagent reinforcement learning," *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, vol. 38, no. 2, pp. 156–172, 2008.

[19] P. Hernandez-Leal, M. Kaisers, T. Baarslag, and E. Munoz de Cote, "A survey of learning in multiagent environments: Dealing with non-stationarity," in *Workshops at the 31st AAAI Conference on Artificial Intelligence*, 2017.

[20] D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein, "The complexity of decentralized control of Markov decision processes," *Mathematics of Operations Research*, vol. 27, no. 4, pp. 819–840, 2002.

[21] F. A. Oliehoek and C. Amato, *A Concise Introduction to Decentralized POMDPs*. Cham: Springer, 2016.

[22] M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning," in *Proceedings of the 11th International Conference on Machine Learning (ICML)*, 1994, pp. 157–163.

[23] M. Tan, "Multi-agent reinforcement learning: Independent vs. cooperative agents," in *Proceedings of the 10th International Conference on Machine Learning (ICML)*, 1993, pp. 330–337.

[24] L. Matignon, G. J. Laurent, and N. Le Fort-Piat, "Independent reinforcement learners in cooperative Markov games: A survey regarding coordination problems," *The Knowledge Engineering Review*, vol. 27, no. 1, pp. 1–31, 2012.

[25] M. Lauer and M. A. Riedmiller, "An algorithm for distributed reinforcement learning in cooperative multi-agent systems," in *Proceedings of the 17th International Conference on Machine Learning (ICML)*, 2000, pp. 535–542.

[26] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Advances in Neural Information Processing Systems*, vol. 30, 2017, pp. 6379–6390.

[27] Y.-H. Chang, T. Ho, and L. P. Kaelbling, "All learning is local: Multi-agent learning in global reward games," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 16, 2004, pp. 1–8.

[28] P. Sunehag *et al.*, "Value-decomposition networks for cooperative multi-agent learning based on team reward," in *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2018, pp. 2085–2087.

[29] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, vol. 32, 2018.

[30] M. Samvelyan *et al.*, "The StarCraft multi-agent challenge," in *Proceedings of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2019, pp. 2186–2188.

[31] J. Terry *et al.*, "PettingZoo: Gym for multi-agent reinforcement learning," in *Advances in Neural Information Processing Systems*, vol. 34, 2021, pp. 15 032–15 043.

[32] K. Zhang, Z. Yang, and T. Başar, "Multi-agent reinforcement learning: A selective overview of theories and algorithms," in *Handbook of Reinforcement Learning and Control*. Cham: Springer, 2021, pp. 321–384.

[33] A. Tampuu *et al.*, "Multiagent cooperation and competition with deep reinforcement learning," *PLOS ONE*, vol. 12, no. 4, p. e0172395, 2017.

[34] A. Juliani *et al.*, "Unity: A general platform for intelligent agents," *arXiv preprint arXiv:1809.02627*, 2020.

[35] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[36] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, 2015, pp. 1889–1897.

[37] V. Mnih *et al.*, "Asynchronous methods for deep reinforcement learning," in *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, 2016, pp. 1928–1937.

[38] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," in *4th International Conference on Learning Representations (ICLR)*, 2016.

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

[40] R. Axelrod and W. D. Hamilton, "The evolution of cooperation," *Science*, vol. 211, no. 4489, pp. 1390–1396, 1981.

[41] L. Panait and S. Luke, "Cooperative multi-agent learning: The state of the art," *Autonomous Agents and Multi-Agent Systems*, vol. 11, no. 3, pp. 387–434, 2005.

[42] J. Z. Leibo, V. Zambaldi, M. Lanctot, J. Marecki, and T. Graepel, "Multi-agent reinforcement learning in sequential social dilemmas," in *Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2017, pp. 464–473.

[43] E. Hughes *et al.*, "Inequity aversion improves cooperation in intertemporal social dilemmas," in *Advances in Neural Information Processing Systems*, vol. 31, 2018, pp. 3326–3336.

[44] M. Zhao, N. Tang, A. L. Dahmani, Y. Zhu, F. Rossano, and T. Gao, "Sharing rewards undermines coordinated hunting," *Journal of Computational Biology*, vol. 29, no. 9, pp. 1022–1030, 2022.

[45] B. Baker *et al.*, "Emergent tool use from multi-agent autocurricula," in *8th International Conference on Learning Representations (ICLR)*, 2020.

[46] I. Mordatch and P. Abbeel, "Emergence of grounded compositional language in multi-agent populations," in *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, vol. 32, 2018.

[47] A. Dafoe *et al.*, "Open problems in cooperative AI," *arXiv preprint arXiv:2012.08630*, 2020.

[48] Unity Technologies, "Unity machine learning agents toolkit (ML-Agents)," https://github.com/Unity-Technologies/ ml-agents, 2024.

[49] A. Y. Ng, D. Harada, and S. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," in *Proceedings of the 16th International Conference on Machine Learning (ICML)*, 1999, pp. 278–287.

[50] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th Annual International Conference on Machine Learning (ICML)*. ACM, 2009, pp. 41–48.

[51] S. Narvekar, B. Peng, M. Leonetti, J. Sinapov, M. E. Taylor, and P. Stone, "Curriculum learning for reinforcement learning domains: A framework and survey," *Journal of Machine Learning Research*, vol. 21, no. 181, pp. 1–50, 2020.

[52] M. Abadi *et al.*, "TensorFlow: A system for large-scale machine learning," in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, 2016, pp. 265–283.

[53] W. McKinney, "Data structures for statistical computing in Python," in *Proceedings of the 9th Python in Science Conference (SciPy)*, 2010, pp. 51–56.

[54] J. D. Hunter, "Matplotlib: A 2D graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.

[55] T. Rashid, M. Samvelyan, C. S. de Witt, G. Farquhar, J. Foerster, and S. Whiteson, "Monotonic value function factorisation for deep multi-agent reinforcement learning," *Journal of Machine Learning Research*, vol. 21, no. 178, pp. 1–51, 2020.

[56] G. Hardin, "The tragedy of the commons," *Science*, vol. 162, no. 3859, pp. 1243–1248, 1968.

[57] C. Barnes, "Interference and volatility in evolutionary agent-based systems," Ph.D. dissertation, Aston University, 2021.

[58] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, "Deep reinforcement learning that matters," in *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 2018, pp. 3207–3214.

[59] W. Huang, Z. Miao, T. Wu, Z. Guo, W. Han, and T. Li, "Design of and experiment with a dual-arm apple harvesting robot system," *Horticulturae*, vol. 10, no. 12, p. 1268, 2024.

[60] Z. Du, S. Yang, and W. Wang, "An adaptive difference policy gradient method for cooperative multi-USV pursuit in multi-agent reinforcement learning," *Journal of Marine Science and Engineering*, vol. 14, no. 3, p. 252, 2026.

Transparenzerklärung

Ich, erkläre hiermit, dass die vorliegende Arbeit selbstständig angefertigt wurde. Die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche, gemäß der am Studiengang bestehenden Richtlinien, kenntlich gemacht.

Dokumentation der Verwendung von KI-gestützten Tools

Im Rahmen der Arbeit wurden folgende KI-gestützte Tools für folgende Zwecke verwendet.

KI-gestütztes Tool	**Art der Unterstützung**	**Beitrag zur Arbeit**
Google Gemini & Anthropic Claude	Sprachanalyse und -korrektur	Sprachliche Glättung und Gegenlesen meiner selbst verfassten Texte zur Verbesserung des wissenschaftlichen Ausdrucks.
Google Gemini & Anthropic Claude	Ideenfindung	Nutzung als „Sparringspartner“ zur Diskussion abstrakter Konzepte und zur kritischen Reflexion meiner Ergebnisse.
Google Gemini & Anthropic Claude	Literaturrecherche	Zusammenfassung komplexer Fachartikel und Unterstützung bei der Suche nach relevanten Quellen.
Google Gemini	Generierung von Code	Hilfe bei Python-Skripten (Diagramme), C#-Logikfragen für Unity sowie Formatierung der

		Arbeit in LaTeX und BibTeX.
Google Gemini	Übersetzungen	Übersetzung der deutschen Zusammenfassung für den englischen Abstract.

Sämtliche Inhalte, die mithilfe von KI-gestützten Tools generiert und als direkte oder indirekte Zitate in die Arbeit übernommen wurden, sind gemäß der vom Studiengang vorgegebenen Zitierweise klar als solche gekennzeichnet.

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides statt, dass ich die vorliegende Bachelorarbeit selbständig angefertigt habe. Die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

Die Arbeit wurde bisher weder in gleicher noch in ähnlicher Form einer anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Innsbruck, 16. Mai 2026

_____

Manuel Junker

Anhang

Inhaltsverzeichnis

A Implementierung der Belohnungslogik (C#)

Die folgenden Code-Ausschnitte dokumentieren die technische Umsetzung der Belohnungssysteme innerhalb der Simulationsumgebung.

A.1 Individuelle Belohnungsvergabe (Ego-Modell)

Individuelle Distanzberechnung und Belohnung

```
// 1. Suche den Agenten mit der geringsten Distanz zum versenkten Blo
PushAgentCollab closestAgent = null;
float minDistance = float.MaxValue;

foreach (var item in AgentsList)
{
    float dist = Vector3.Distance(item.Agent.transform.position, col.
    if (dist < minDistance)
    {
        minDistance = dist;
        closestAgent = item.Agent;
    }
}

// 2. Belohne exakt diesen Agenten mit dem Wert des Blocks
if (closestAgent != null)
{
    closestAgent.AddReward(score);
}
```

A.2 Gemeinsame Teambelohnung (Shared Reward)

Globale Teambelohnung

```
// Erteilung der Block-Belohnung an die gesamte Agentengruppe
m_AgentGroup.AddGroupReward(score);
```

```
if (done) {
    m_AgentGroup.EndGroupEpisode(); // Beendet Episode fuer alle Mitg
}
```

A.3 Implementierung der Strafen (Hurry Up Penalty)

Hurry Up Penalty im Ego-Modell

```
void FixedUpdate()
{
    m_ResetTimer += 1;
    // Kontinuierliche Bestrafung pro Step zur Foerderung von Geschwi
    foreach (var item in AgentsList)
    {
        item.Agent.AddReward(-1.0f / MaxEnvironmentSteps);
    }
}
```

B Zentrale PPO-Trainingskonfiguration

Der folgende Auszug zeigt die Basis-Konfiguration des PPO-Algorithmus am Beispiel des kooperativen Modells (*TeamMitCurriculum*).

trainer_config.yaml

```
default_settings: null
behaviors:
  PushBlockCollab:
    trainer_type: ppo
    hyperparameters:
      batch_size: 1024
      buffer_size: 10240
      learning_rate: 0.0003
      beta: 0.01
      epsilon: 0.2
      lambd: 0.95
      num_epoch: 3
      shared_critic: false
      learning_rate_schedule: linear
      beta_schedule: linear
      epsilon_schedule: linear
    checkpoint_interval: 500000
    network_settings:
      normalize: false
      hidden_units: 256
      num_layers: 2
      vis_encode_type: simple
      memory: null
      goal_conditioning_type: hyper
      deterministic: false
    reward_signals:
      extrinsic:
```

```
      gamma: 0.99
      strength: 1.0
      network_settings:
        normalize: false
        hidden_units: 128
        num_layers: 2
        vis_encode_type: simple
        memory: null
        goal_conditioning_type: hyper
        deterministic: false
    curiosity:
      gamma: 0.99
      strength: 0.05
      network_settings:
        normalize: false
        hidden_units: 256
        num_layers: 2
        vis_encode_type: simple
        memory: null
        goal_conditioning_type: hyper
        deterministic: false
      learning_rate: 0.0003
      encoding_size: 256
  init_path: null
  keep_checkpoints: 5
  even_checkpoints: false
  max_steps: 10000000
  time_horizon: 64
  summary_freq: 10000
  threaded: false
  self_play: null
  behavioral_cloning: null
```

```
env_settings:
  env_path: null
  env_args: null
  base_port: 5005
  num_envs: 1
  num_areas: 1
  timeout_wait: 60
  seed: -1
  max_lifetime_restarts: 10
  restarts_rate_limit_n: 1
  restarts_rate_limit_period_s: 60
engine_settings:
  width: 84
  height: 84
  quality_level: 5
  time_scale: 20
  target_frame_rate: -1
  capture_frame_rate: 60
  no_graphics: false
  no_graphics_monitor: false
environment_parameters:
  lesson_level:
    curriculum:
    - value:
        sampler_type: constant
        sampler_parameters:
          seed: 1938
          value: 0.0
      name: Lesson0_SmallBlocks
      completion_criteria:
        behavior: PushBlockCollab
        measure: reward
```

```
        min_lesson_length: 100
        signal_smoothing: true
        threshold: -0.6
        require_reset: false
    - value:
        sampler_type: constant
        sampler_parameters:
          seed: 1939
          value: 1.0
      name: Lesson1_MediumBlocks
      completion_criteria:
        behavior: PushBlockCollab
        measure: reward
        min_lesson_length: 100
        signal_smoothing: true
        threshold: -0.2
        require_reset: false
    - value:
        sampler_type: constant
        sampler_parameters:
          seed: 1940
          value: 2.0
      name: Lesson2_LargeBlocks
      completion_criteria: null
checkpoint_settings:
  run_id: Team_Curriculum_02
  initialize_from: null
  load_model: false
  resume: false
  force: false
  train_model: false
  inference: false
```

```
  results_dir: results
torch_settings:
  device: null
debug: false
```