Die visuelle Erklärbarkeit einer Reinforcement Learning Umgebung

Nadia Hintze

Hochschule für Angewandte Wissenschaften Hamburg,
Berliner Tor 7, 20099 Hamburg, Germany

Abstract. Die Visualisierung von Reinforcement Learning Prozessen hilft dabei, andernfalls schwer nachvollziehbare Vorgänge verständlich zu machen. Sie können als Funktionsnachweis eines fertig trainierten Modells oder als Einblick in den Trainingsprozess dienen. Im Rahmen dieser Forschungsarbeit wurde eine Übersicht über einige gängige Visualisierungsmethoden aus anderen Forschungsarbeiten angefertigt. Obwohl Visualisierungen unterschiedliche Zielgruppen bedienen müssen, sind uns keine Darstellungsformen aus dem XRL Bereich bekannt, die sich an Personen richten, die nicht im Detail wissen, wie die eingesetzten RL-Methoden funktionieren. Um diese Zielgruppen ansprechen zu können, haben wir Visualisierungsmethoden entwickelt, die sich optisch in eine bestehende RL-Umgebung einfügen, um so mithilfe dieses Kontextes leichter intuitiv interpretierbar zu sein. Bei der Entwicklung wurde berücksichtigt, dass ggf. mehrere Visualisierungen zusammen eingeblendet werden müssen, um in einen Kontext gebracht zu werden. Dabei besteht eine der größten Herausforderungen darin, dass sie übersichtlich bleiben müssen und gleichzeitig nicht zu viele Detailinformationen verloren gehen dürfen.

Keywords: Explainable Reinforcement Learning · XRL · Visual Analytics · ML-Agents

1 Einleitung

Bei Reinforcement Learning[1] handelt es sich um Verfahren, mit denen ein virtueller Agent lernen kann, wie er in bestimmten Situationen reagieren muss, um langfristig den meisten Gewinn für sich zu erzielen. Welche Aktionen er wann dafür benutzen muss, wird durch Ausprobieren und die Beobachtung der Folgen seines Handelns für den Gewinn herausgefunden. [8]

Mit unbekannten Situationen konfrontiert zu werden und herauszufinden zu müssen, wie am besten mit ihnen umgegangen werden kann, ist ein fester Bestandteil des menschlichen Lebens. Die Grundlagen eines solchen Lernprozesses sind deshalb intuitiv nachvollziehbare Vorgänge. Es gibt jedoch einige Eigenschaften von RL Methoden, die es zu einer Herausforderung machen, ihre

[1] Reinforcement Learning wird im Folgenden abgekürzt durch "RL".

Vorgänge zu verstehen. Dazu gehören der überaus lange Zeitraum, den ein Training in Anspruch nehmen kann und der Gebrauch von Neuronalen Netzen, die oftmals einer Black Box gleichkommen [3]. Zu einer fehlenden Übersicht trägt ebenfalls bei, dass der Spielraum der vom Agenten ausprobierten Aktionen durch teils zufällige Entscheidungen sehr groß werden kann. An dieser Stelle können Visualisierungsmethoden eingesetzt werden, um den Lernprozess sowie das gelernte Modell verständlich zu machen [9]. Das ist insbesondere wichtig, um die Funktion eines fertig gelernten Modells nachweisen zu können und um das Debugging und die anschließende Optimierung eines Trainingsdurchlaufs zu unterstützen [1].

Diese Forschungsarbeit setzt sich zweierlei Ziele. Zunächst möchte sie einen grundlegenden Eindruck davon vermitteln, welche Möglichkeiten es zur Visualisierung gibt. Zu diesem Zweck werden in Kapitel 2 einige Beispiele unterschiedlicher Forschungsprojekte vorgestellt. Zudem werden sie anhand ihrer angesprochenen Zielgruppe und dem damit verfolgten Zweck klassifiziert, um verschiedene Formen der Visualisierung miteinander vergleichbar zu machen. Das zweite Ziel dieser Forschungsarbeit ist die Erarbeitung von Visualisierungsmethoden, die sich optisch besonders intuitiv in eine bestehende RL-Umgebung einfügen. Sie sollen ein von Agenten gelerntes Verhaltensmodell verdeutlichen. In Kapitel 3 werden zunächst die für den Test entwickelte RL-Umgebung und die dafür genutzte Pipeline vorgestellt. In Kapitel 4 folgt die Präsentation der entwickelten Visualisierungen. Anschließend werden in Kapitel 5 die entwickelten Methoden evaluiert. Dabei wird unter Anderem beurteilt, inwiefern sie die definierten Ziele intuitive Lesbarkeit“ und Übersichtlichkeit mehrerer visueller Informationen“ erfüllen können. Zuletzt folgt in Kapitel 6 eine Zusammenfassung der wichtigsten Ergebnisse dieser Arbeit.

2 Visualisierungsmethoden

In diesem Kapitel folgt die Vorstellung unterschiedlicher Visualisierungsmethoden. Die dabei entstehende Liste verfolgt keinen Anspruch auf Vollständigkeit, sondern soll einem ersten, grundlegenden Eindruck dienen, sodass die in Kap. 4 entwickelten Visualisierungen in diesen Kontext eingeordnet werden können. Damit unterschiedliche Darstellungsformen miteinander verglichen werden können, wird zunächst eine Möglichkeit zur Klassifikation vorgestellt.

2.1 Klassifikation

Es gibt viele Möglichkeiten, Reinforcement Learning Modelle und ihren Lernprozess darzustellen. Mit der steigenden Komplexität der neu entwickelten Lern-Algorithmen und der damit erschlossenen Einsatzmöglichkeiten steigt auch die Anzahl der Facetten, die visualisiert werden können. Welche Darstellungsform im Einzelfall geeignet ist, kann man von unterschiedlichen Faktoren abhängig machen. Für die Evaluation und Klassifikation von Visualisierungsmethoden

wird sich im Folgenden an zwei Eigenschaften orientiert. Diese haben Heuillet et al. in ihrer Übersichtsarbeit zu Thema Explainable Reinforcement Learning[2] definiert [1].

Die erste Eigenschaft ist die Zielgruppe einer XRL-Erklärung. Wie eine Visualisierung vom Betrachter aufgenommen wird, ist abhängig von seinem Vorwissen zu dem Thema, seiner individuellen Auffassungsgabe und dem Schwerpunkt seines Interesses. So kann beispielsweise ein Domain Experte, der seit mehreren Jahren mit Machine Learning Verfahren arbeitet und der an der Weiterentwicklung dieser Verfahren interessiert ist, mehr mit speziellen Kennwerten von RL-Methoden anfangen als ein System-Benutzer, dessen Augenmerk in erster Linie auf der korrekten Funktionalität des Endprodukts liegt. Heuillet et al. identifizieren unter anderem folgende Zielgruppen: Entwickler, Benutzer, Domain Experten und Führungskräfte der Unternehmen.

Die zweite Eigenschaft zur Charakterisierung einer Visualisierungsmethode ist das damit verfolgte Ziel. Heuillet et al. definieren zwei zentrale Ziele einer XRL-Erklärung. Das erste ist ein Funktionsnachweis eines fertig trainierten Modells. Dieser Nachweis kann z.B. besonders relevant sein, um vor dem Einsatz in kritischen Umgebungen wie der Medizin oder dem Militär zu beweisen, dass das System für einen Einsatz geeignet ist. Das zweite Ziel ist die Unterstützung der Entwickler bei dem Debugging, dem Verständnis und der Optimierung der Trainings-Algorithmen.

2.2 Numerische Kenngrößen als Trainingsbeschreibung

Ein Lernprozess kann beschrieben werden, indem man die Veränderungen von numerischen Kenngrößen im Trainingsverlauf betrachtet. Um eine möglichst ausdrucksstarke Repräsentation eines Trainings zu erhalten, muss die Wahl der Kenngrößen an die RL-Umgebung und die gewählten Trainingsmethoden angepasst werden. Wang et al. haben sich beispielsweise mit der Visualisierung von einem DQN Training auseinandergesetzt [9]. Sie haben einen Agenten lernen lassen das Spiel "Breakout"[3] zu spielen. Dafür haben sie als besonders relevante Kenngrößen den erzielten Reward pro Episode, die in einer Epoche durchlaufene Anzahl von Spielrunden, den geschätzten Value Q, die Abweichung des geschätzten von dem eingetroffenen Value ("Loss") identifiziert. Diese Kenndaten haben sie als Liniendiagramm dargestellt (Abb. 1).

In einer RL-Umgebung, in der zwei konkurrierende Agenten trainieren, sind andere Größen aussagekräftiger. Als Beispiel für eine solche Umgebung kann man sich ein Tennis Spiel vorstellen. In diesem Fall ist der Reward-Verlauf nicht so repräsentativ für den Trainingsfortschritt, da die Anzahl der erzielten Punkte

[2] Explainable Reinforcement Learning wird im Folgenden abgekürzt durch XRL".
[3] Wikipedia: "Breakout (videogame)", https://en.wikipedia.org/wiki/Breakout_(video_game), Letzter Zugriff: 30.3.2021

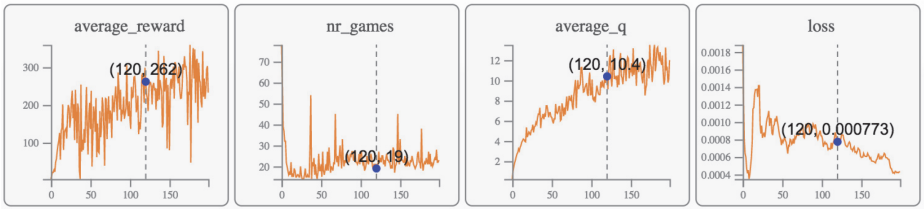

Fig. 1. RL Kenngrößen im Trainingsverlauf in DQNViz. Quelle: Wang et al., 2018, S.1 [9]

immer von der Leistung des Gegenspielers abhängt. So können auch bloß Punkte erzielt werden, weil der Gegenspieler den Ball nicht gut zurückspielt. Der Punktestand ist also keine Aussage darüber, wie gut sich die beiden Agenten insgesamt schlagen. Aufgrund der Abhängigkeit des Losses und von Q von dem Reward sind auch diese Verläufe dann nur begrenzt aussagekräftig [10]. Wenn zusätzlich die beiden Gegenspieler dieselbe Policy im "self-play" erarbeiten, sind beide stets annähernd ebenbürtig [6]. In Kap. 3 wird ein solches Beispiel skizziert. Darin wird jeder Erfolg eines Agenten mit einem positiven Reward belohnt und der Gegenspieler mit einem gleich großen, aber negativen Reward bestraft. Demzufolge schwankt der Reward für das gemeinsame Training um den Nullpunkt und könnte den Anschein erwecken, dass die Agenten sich nicht verbessert haben (Abb. 2a). Um diese Probleme zu lösen, kann stattdessen das Maß "ELO" zurate gezogen werden, um einen Trainingsfortschritt darzustellen. Dieses Maß gibt die relative Leistung eines Agenten im Laufe des Trainings an (Abb. 2b).

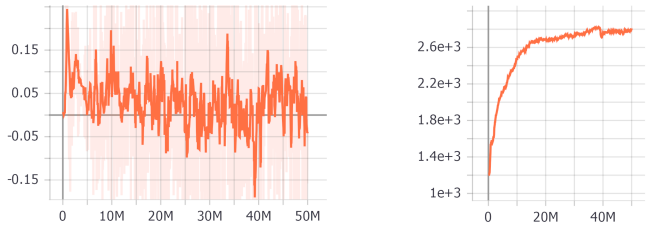

Fig. 2. a) Schwankender, kumulativer Reward (geglättet um 85%) und b) ELO pro Trainingsschritt in einem self-play Training. Quelle: Nadia Hintze (2021)

2.3 Die Verteilung unterschiedlicher Reward Kategorien und Aktionen

In einigen Fällen kann es hilfreich sein, im RL Modell verschiedene Arten von Rewards unterscheiden. Damit kann unter anderem erklärt werden, warum Agenten bestimmte Aktionen in bestimmten Situationen präferieren [1][2].

Wang et al. haben in ihrem Szenario drei Ereignis Typen definiert, die für den Agenten einen unterschiedlich hohen Reward geben [9]. Je schwieriger das Ereignis zu erreichen ist, desto höher ist der Reward. Indem sie nun die Verteilung der erzielten Rewards pro Epoche darstellen, kann man daran ablesen, welche Ereignisse der Agent im Trainingsverlauf auslösen konnte. Um diese Verteilung zu visualisieren, haben sie unterschiedliche Diagramme gewählt: ein Tortendiagramm, ein gestapeltes Balkendiagramm und ein gestapeltes Flächendiagramm (Abb. 3). Das Flächendiagramm zeigt die Verteilung der erzielten Rewards im Laufe der nacheinander folgenden Epochen eines Trainings (Abb. 3b). Somit kann festgestellt werden, wie sich die Verteilung im Laufe der Zeit verändert. Im Balkendiagramm ist die Reihenfolge der Epochen auf der x-Achse der Höhe des in der Epoche erzielten Rewards nach sortiert. Somit kann man ablesen, in welchen Epochen der Agent besonders erfolgreich war. Im Balkendiagramm kann dann eine Epoche ausgewählt werden, zu der das Tortendiagramm abermals die Verteilung der Rewards in dieser Epoche darstellt (Abb. 3a).

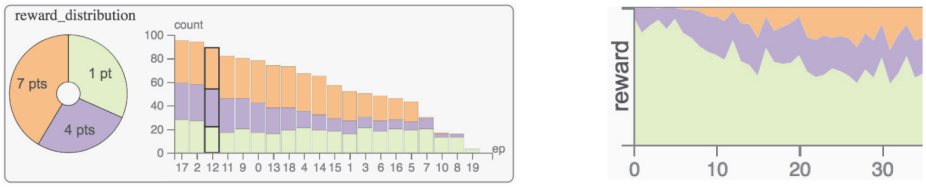

Fig. 3. Verteilung unterschiedlicher, erzielter Rewards als a) Torten- und gestapeltes Balkendiagramm und b) als gestapeltes Flächendiagramm. Quelle: Wang et al., 2018, S.1 [9], bearbeitet durch Nadia Hintze

Wang et al. nutzen die Diagramm-Varianten aus Abb. 3 ebenfalls, um die Verteilung der Aktionen, die vom Agenten getätigt werden, darzustellen [9]. Die Autoren sind in ihrer Evaluation nicht darauf eingegangen, wie hilfreich die Verteilung der Aktionen in ihrem Fall war. Wir können uns jedoch vorstellen, dass dies vom Anwendungsfall abhängt. Ein denkbarer Anwendungsfall, in dem eine solche Übersicht hilfreich wäre, ist die RL Umgebung eines entwickelten Spiels, in der der Entwickler testen möchte, ob alle von ihm eingebauten Aktionen vom Spieler benötigt werden, um ein Spiel zu gewinnen.

2.4 Die Untersuchung von Verhaltens-Patterns

Ebenfalls aufschlussreich kann die Untersuchung von Verhaltens-Patterns sein, die ein Agent im Training immer wieder verfolgt. Dafür können in einem Experience-Buffer Ketten von Aktionen, Umgebungszuständen und Rewards gefunden werden, die sich zu wiederkehrenden Strategien zusammensetzen [9]. Auch mit der Visualisierung dieser Patterns haben sich Wang et Al. auseinandergesetzt. Zunächst haben sie mit einem Liniendiagramm gezeigt, wie häufig ein fest-

gelegtes Pattern pro Epoche gefunden wurde. Dieses Diagramm ist optisch vergleichbar mit den Diagrammen der numerischen Kenngrößen in Kapitel 2.2 Zusätzlich dazu haben sie ein Diagramm erstellt, das die Abfolge der vom Agenten ausgeführten Aktionen als Liniendiagramm darstellt. In diesen Zeitverläufen werden die Zeiträume markiert, in denen ein Pattern auftritt (Abb. 4).

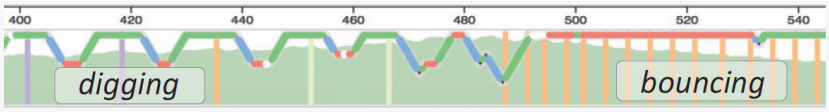

Fig. 4. Markierung der auftretenden Patterns "digging" und "bouncing" in einer Aktionskette. Quelle: Wang et al., 2018, S.1 [9], bearbeitet durch Nadia Hintze

2.5 Saliency Maps

In einem RL-Modell können Faltungsnetzwerke ("CNNs") verwendet werden, um große Mengen an Daten zu verwerten. Das ist nützlich, wenn die Observations einer Umgebung durch ein zweidimensionales Abbild der Umgebung wiedergegeben werden sollen. Dieses Verfahren kann beispielsweise bei dem Trainieren von Atari Spielen eingesetzt werden [9]. Um offenzulegen, wie die CNNs das Eingabe-Bild verarbeiten, können unterschiedliche Heatmaps angelegt werden. Diese Maps markieren die Bereiche des Bildes, die besonders viel zu dem Output der CNNs und damit zu der Entscheidung eines Agenten beitragen haben. Diese Kategorie von Visualisierungen wird auch "Saliency Map" genannt. Um das zu erreichen, gibt es viele unterschiedliche Techniken [1], [5], [7], [9]. Einige Beispiele möglicher Ausprägungen sind in Abb. 5 zu sehen.

Fig. 5. Unterschiedliche Saliency Maps. Quellen: a) Wang et al., 2018, S. 1 [9], bearbeitet durch Nadia Hintze. b-c) Selvaraju et al., 2017, S. 2 [5]

2.6 Klassifikation der untersuchten Visualisierungsmethoden

Im Kapitel 2.1 wurde etabliert, dass Visualisierungsmethoden für unterschiedliche Zielgruppen und zu unterschiedlichen Zwecken gebraucht werden. Wang et al. haben im Rahmen ihrer Forschungsarbeit einige Personen einer festen Zielgruppe mit einbezogen. Die Forscher haben ihre Visualisierungen in Zusammenarbeit mit RL-Entwicklern erarbeitetet, die zu dem Zeitpunkt der Forschungsarbeit bereits jahrelang Erfahrungen mit RL-Methoden gesammelt hatten. Wang et al. haben positive Rückmeldungen von den Entwicklern bekommen [9]. Jedoch wäre zu untersuchen, ob diese Visualisierungsmethoden auch für andere Zielgruppen von Nutzen wären. Um alle Diagramme, ausgenommen der Saliency Maps, verstehen und interpretieren zu können, ist Fachwissen zu den dargestellten Größen und ihren Zusammenhängen zwingend notwendig. Das bedeutet, dass sich Personen, die keine tiefgehenden Erfahrungen mit Reinforcement Learning haben, zuerst das entsprechende Wissen aneignen müssen.

Ebenfalls auffällig ist, dass die Saliency Map die einzige uns bekannte Visualisierungsform ist, die sich visuell an der RL-Umgebung orientiert. Alle anderen Diagramme sind optisch davon losgelöst. Wir vermuten, dass dieser Sachverhalt es schwieriger macht, mehrere Informationen in einem Zusammenhang zu interpretieren, da erst die Zuammenänge zwischen mehreren Diagrammen und der Umgebung etabliert werden müssen.

Hinsichtlich des Zwecks wurden die meisten von uns untersuchten Visualisierungen darauf ausgelegt, in erster Linie Trainingsprozesse offenzulegen. Jedoch können mithilfe der Visualisierung auch die Endzustände eines Trainings begutachtet werden. Somit können auch Aussagen über die Funktionalität eines fertig gelernten Modells getroffen werden.

3 Die Testumgebung

In Kap. 2.6 wurde die Hypothese aufgestellt, dass ein Bedarf an Visualisierungsmethoden besteht, die auch für Personen interpretierbar sind, die kein tiefes Hintergrundwissen zum Thema Reinforcement Learning mitbringen. Eine zweite Hypothese bestand darin, dass eine Visualisierung, die sich optisch in die RL-Umgebung einfügt, es leichter macht, mehrere Informationen in einem gemeinsamen Kontext zu interpretieren. Um diese Thesen zu untersuchen, werden in den folgenden Kapiteln beispielartige Visualisierungen entwickelt, die diese beiden Ziele verfolgen sollen, ohne deshalb zu viel an Detailinformationen zu verlieren. Im Folgenden wird zunächst eine prototypische RL Umgebung erstellt, in der die Visualisierungen realisiert werden.

3.1 Pipeline

Wir haben uns für die Entwicklung einer RL Umgebung in der Unity3D Engine entschieden. Zum Trainieren der Agenten nutzen wir die Implementierung des

PPO Algorithmus des Toolkits "ML-Agents"[4]. ML-Agents verwendet dafür die Plattform "Tensorflow" und orientiert sich an der Implementierung des Algorithmus durch OpenAI. Diese Implementierung wurde erstmals von Schulman et al. entworfen und veröffentlicht [4].

3.2 RL Umgebung

Für diese Arbeit wurde als Anwendungsbeispiel eine Umgebung umgesetzt in der zwei Agenten im self-play Modus lernen, "Blobby Volley"[5] zu spielen. Dabei geht es darum, dass zwei konkurrierende Spieler in einer vereinfachten Version von Volleyball versuchen, den Ball in das Feld des jeweils anderen Spielers zu spielen. Ein Spieler kann den Ball von der eigenen Spielfeldhälfte wegbewegen, indem er geschickt positioniert dagegen springt. Wenn der Ball den Boden berührt, bekommt der Spieler auf der anderen Seite des Spielfelds einen Punkt. Die Bewegungen finden dabei in einem zweidimensionalen Raum statt (siehe Abb. 6).

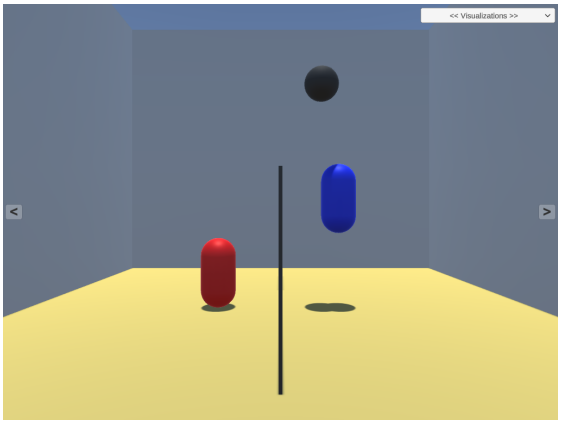

Fig. 6. Blobby Volley Umgebung in Unity3D

Ein Spieler kann zwei Aktionen durchführen. Zum einen bewegt der sich horizontal über das Spielfeld und zum anderen kann er in die Luft springen. In diesem Fall wird die horizontale Bewegung durch einen kontinuierlichen Wert zwischen -1 und 1 abgebildet. Dabei gibt die absolute Größe des Werts die Geschwindigkeit an. Der Sprung kann durch einen ebenfalls kontinuierlichen Wert zwischen 0 und 1 ausgelöst werden.
Die gewählten Rewards für das RL Modell orientieren sich an dem Spielstand. Wenn ein Spieler einen Punkt erzielt, bekommt er einen Reward von +1 und

[4] Unity-Technologies: "ml-agents", https://github.com/Unity-Technologies/ml-agents, Letzter Zugriff: 01.04.2021
[5] Wikipedia: "Blobby Volley", https://de.wikipedia.org/wiki/Blobby_Volley, Letzter Zugriff 01.04.2021

der Gegenspieler einen von -1. Als Spielende haben wir definiert, dass ein Spieler mindestens drei Punkte erreicht haben muss. Dann wird der Gewinner abermals mit +3 Reward-Punkten belohnt und der Verlierer mit -3 Punkten betraft.
Als Observations benutzen die Agenten ihre eigene Position und ihre aktuelle Bewegungsgeschwindigkeit entlang der X- und Y-Achse. Dieselben Daten bekommen sie auch für den aktuellen Zustand des Balls. Somit hat der Observation Space eine Größe von acht kontinuierlichen Zahlen.

4 Die Visualisierungen

4.1 Observations und Aktionen

Der erste Schritt, den wir genommen haben, um ein gelerntes RL-Modell in der Unity Umgebung sichtbar zu machen, besteht darin, den aktuellen Zustand des Modells darzustellen. Den Zustand definieren wir als die aktuell vom Agenten ausgeführten Aktionen, die von ihm genutzten Observations und seinen verdienten Reward. Es ist notwendig diese zusätzlich zu visualisieren, weil ansonsten diese Aspekte in manchen Situationen für den Betrachter nicht sichtbar sind. In unserem Fall bewegt sich z.B. der Spieler nicht mehr weiter nach links oder rechts, wenn er an eine Wand oder das Netz stößt, obwohl er die Aktion dafür weiterhin ausführt. Ein weiteres Beispiel ist ein Sprung, der nur ausgeführt wird, wenn der dazugehörige, kontinuierliche Aktions-Wert auf 1 steht. Die Bewegungs-Aktionen werden als Pfeile an den jeweiligen Agenten ausgedrückt. Die Länge eines Pfeils stellt dabei die normierte Größe des kontinuierlichen Wertes dar (Abb. 7a).

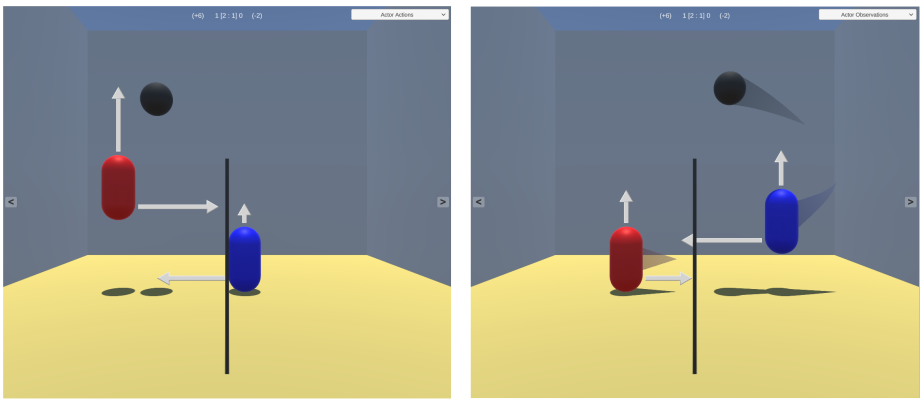

Fig. 7. a) Sichtbarkeit andernfalls versteckter Aktionen, b) Gemeinsame Visualisierung von Aktionen und Observations

Auch wenn die Observations ohne zusätzliche Visualisierungen sichtbar sind, sind manche Eigenschaften im Detail für uns trotzdem schwer visuell einzuschätzen.

In diesem Fall trifft das z.B. auf die Bewegungsgeschwindigkeiten der Spielerfiguren und des Balls zu. Um diese zu betonen und leichter einschätzbar zu machen, wurde eine farbliche Spur für die betroffenen Objekte eingebaut, die diese hinter sich herziehen. Eine Spur verdeutlicht einerseits die Bewegungsrichtung und andererseits durch ihre Länge die Bewegungsgeschwindigkeit der Objekte. Um die Spuren visuell leichter auseinander halten zu können, falls sie sich mal überschneiden, wurden sie farblich den Objekten angepasst, zu denen sie gehören (Abb. 7b).
Um für die Übersicht auch einen Eindruck des aktuell verdienten Rewards zu haben, kann der aktuelle Punktestand zusammen mit der Anzahl gewonnener Spiele und dem jeweils erreichten aufaddierten Reward eingeblendet werden. Das ist ebenfalls in Abb. 7 zu sehen.

4.2 Der geschätzte Wert eines Zustands

In Kapitel 4.1 wurde der aktuelle Zustand der Umgebung, den der Agent zur Bestimmung seiner nächsten Aktion benutzt, sichtbar gemacht. Passend dazu wird nun eine Visualisierung dafür vorgestellt, für wie gut der trainierte Agent diesen Zustand hält. Dafür wird der von ihm geschätzte Value Q repräsentiert. Um visuell zu unterstreichen, dass es sich dabei um eine Bewertung handelt, wird der Wert, angelehnt an die Farben einer Ampel, mit einem Farbverlauf zwischen rot, gelb und grün dargestellt. Als genaue Form der Visualisierung wurden zwei Alternativen in Betracht gezogen. Der erste Ansatz blendet pro Agent einen Farbverlauf als Skala ein und zeigt mithilfe eines Cursors, wie hoch der Agent Q einschätzt (Abb. 8a). Die zweite Option ist die Umsetzung der Farbe an der jeweiligen Stelle der Skala als Textur des Agenten (Abb. 8b). In Kapitel 5.2 werden die beiden Optionen gegenübergestellt und die jeweiligen Vor- und Nachteile erläutert.

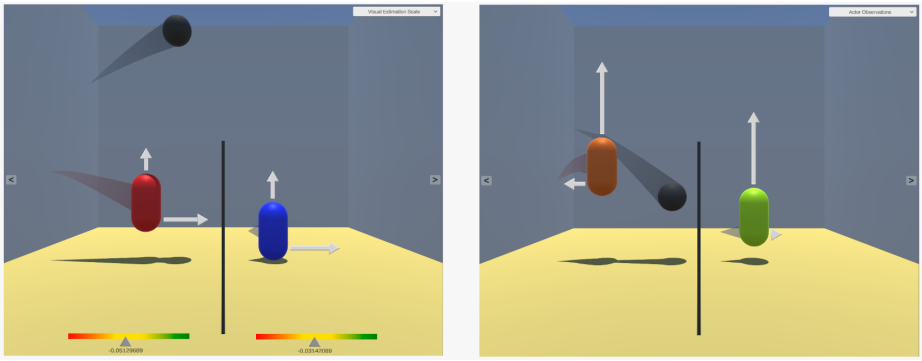

Fig. 8. Die Visualisierung von Q als a) Skala und b) Texturfarbe

4.3 Der Überblick über mehrere Visualisierungen

Die erläuterten Visualisierungen wurden mit dem Hintergedanken entworfen, dass mehrere davon gleichzeitig in der Umgebung eingeblendet werden können, wenn sie in einen Kontext gebracht werden müssen, um gemeinsam interpretierbar zu sein.
Bei vielen Visualisierungen haben wir bewusst darauf verzichtet in der Umgebung die genauen numerischen Werte einzublenden. Das hat zur Folge, dass bei der Betrachtung dieser Darstellungen die Einschätzung der Werte nur grob erfolgen kann. Dafür hat diese Herangehensweise den großen Vorteil, dass mehrere Visualisierungen eingeblendet werden können, ohne dass das Bild sofort visuell überladen ist und den Betrachter überfordern könnte. Um die genauen Werte des aktuellen Umgebungs-Zustands zu bekommen, wurden Skripte angefertigt, die diese Werte zu jedem Zeitpunkt zusätzlich im Unity Inspektor anzeigen können (siehe Abb. 9).

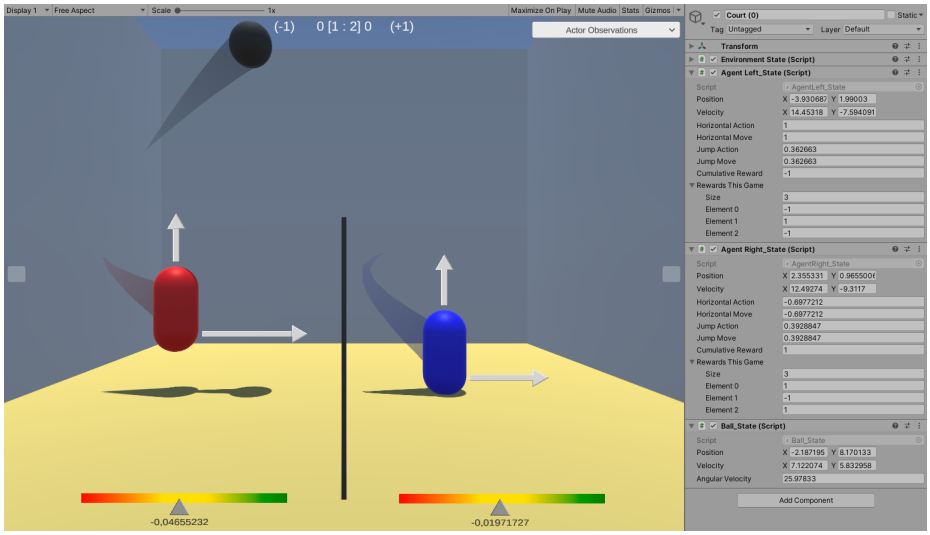

Fig. 9. Darstellung des Umgebungszustands

Die zweite Funktion ist ein Dropdown-Menü, über das einzelne Darstellungen ein- und ausgeschalten werden können. Das soll die nötigte Flexibilität bieten, um immer nur genau die Visualisierungselemente einblenden zu können, die für den Betrachter hilfreich sind und in einem gemeinsamen Kontext betrachtet werden müssen.

5 Auswertung

Das folgende Kapitel beschäftigt sich mit der Evaluation der vorgestellten Visualisierungsmethoden. Zunächst werden die Darstellungsformen hinsichtlich der zu Beginn in Kapitel 3 definierten Ziele kritisch betrachtet. Diese Ziele lauteten zum einen die intuitive Lesbarkeit, um auch Zielgruppen bedienen zu können, die keine tiefgehenden RL Kenntnisse haben, und zum anderen eine übersichtliche Darstellung verschiedener Visualisierungen in einem gemeinsamen Kontext.

5.1 Lesbarkeit

Die angemessene Evaluation der Interpretierbarkeit von XRL-Methoden ist mit einigen Schwierigkeiten verbunden. Heuillet et al. identifizieren dafür zwei Gründe [1]. Einer davon ist der Umstand, dass die Erklärbarkeit von Reinforcement Learning noch nicht so stark als Forschungsschwerpunkt etabliert ist, wie in anderen Bereichen des Maschinellen Lernens. Daher gibt es noch keinen weitreichenden Konsens darüber, wie Metriken für Bewertungen aussehen können. Der zweite Grund ist die Eigenschaft von Erklärungen und Visualisierungen, von unterschiedlichen Zielgruppen betrachtet und ausgewertet zu werden (vgl. Kap. 2.1). Damit geht einher, dass der Effekt und der Nutzen einer Visualisierung immer von der individuellen Auffassungsgabe, dem aktuellen Kenntnisstand und dem Interesse des jeweiligen Betrachters abhängig ist. Somit ist für eine Bewertung stets die Berücksichtigung von subjektiven Aussagen notwendig. Deshalb ist ein häufig gewähltes Mittel der Evaluation eine Befragung in Form von Nutzerstudien. Heuillet et al. schlagen vor, dabei gezielt die Differenz zwischen dem tatsächlich gelernten RL-Modell und dem bei der Betrachtung entstandenen, mentalen RL-Modell zu bestimmen. Dafür können Fragen an den Betrachter wie z.B. "Welches Ziel will der Agent erreichen?" oder "Welche Aktion wird der Agent als nächstes ausführen?" gestellt werden. Bei einer Evaluation der Visualisierungsmethoden wäre besonders interessant zu beobachten, wie viel Hintergrundwissen ein Teilnehmer zum Thema RL im Schnitt benötigt, um die Visualisierungen richtig zu deuten. Dabei könnte auch festgestellt werden, wie unterschiedlich unsere Visualisierungsformen und die Darstellungen in Diagrammform, angelehnt an die aus Kapitel 2, von verschiedenen Zielgruppen interpretiert werden. Dafür wäre es erforderlich, die Inhalte der Visualisierungsformen so zu gestalten, dass sie inhaltlich vergleichbar sind.

5.2 Übersichtliche Darstellung mehrerer Visualisierungen

Um mehrere Informations-Visualisierungen in einem gemeinsamen Kontext betrachten zu können, ist es wichtig, dass diese übersichtlich wirken. Wir haben bei der Entwicklung mehrere, wesentliche Eigenschaften identifiziert, die diese Wirkung negativ beeinflussen können.

Die erste dieser Eigenschaften ist, dass sich zu viele Visualisierungen gleichzeitig zu hektisch und uneinheitlich bewegen. Wenn dies passiert, kann das

dazu führen, dass der Betrachter visuell überlastet wird und der Überblick darunter leidet. Da sich alle von uns entwickelten Visualisierungen immer auf den aktuellen Zustand der Umgebung beziehen, verändern sie sich im Spielverlauf. Somit sind einige Bewegungen in dieser Visualisierungsform unvermeidbar. Allerdings gibt es Darstellungen, bei denen man diese optischen Veränderungen einschränken kann. In Kapitel 4.2 wurden die Repräsentation von Q als farbliche Skala und als Texturfarbe vorgeschlagen. Die Umsetzung als Farbskala hat z.B. den Vorteil, dass sich mit dem Cursor nur ein kleiner Bereich des Bildes im laufenden Spiel bewegt. Wenn Q als Textur dargestellt wird, hat das den Nachteil, dass mit den Spielfiguren die zentralsten Objekte der Umgebung dauernd ihren Zustand verändern, was zusammen mit anderen Veränderungen, wie z.B. den Aktions-Pfeilen, irritierend wirken kann.

Die zweite Eigenschaft ist, dass sich unterschiedliche, wichtige Objekte in der Kameraansicht überlappen können. Zu solchen Verdeckungen kann es vor allem dann kommen, wenn für die Visualisierungen zu viele Objekte der Umgebung neu hinzugefügt werden. Das spricht für eine Q Visualisierung als farbliche Textur und nicht als am Kamerabild orientierte Skala, die einer Umgebung immer gesondert hinzugefügt werden muss. Ein zweiter Umstand, der Überlappungen begünstigt, ist eine Kamera, die dynamisch unterschiedliche Blickwinkel annehmen kann. Dieses Problem kann umgangen werden, indem statisch die Blickwinkel festgelegt werden, aus denen man die Umgebung am besten betrachten sollte und die Visualisierungen gezielt an diese anpasst werden. Dafür spricht ebenfalls, dass manche Visualisierungen einfach nicht gut für manche Blickwinkel geeignet sind. So wird z.B. die Interpretation des Sprung-Pfeils schwierig, wenn die Kamera von oben auf das Spielfeld blickt (Abb. 10). Um zu verstehen, dass der Pfeil einen Sprung darstellt, ist es aber wichtig, dass er eindeutig nach oben zeigt.

Die dritte Eigenschaft, die die Übersichtlichkeit negativ beeinflussen kann, besteht dann, wenn die Zugehörigkeit der visuellen Elemente zu den jeweiligen Agenten nicht klar erkennbar ist. Die Visualisierung von Q als Texturfarbe hat die Stärke, dass in diesem Fall kein Objekt der Umgebung künstlich hinzugefügt werden muss. Somit muss keine zusätzliche Zuordnung erfolgen. Bei dem Q Value als farbliche Skala muss immer darauf geachtet werden, dass die Skala nah genug am jeweiligen Agenten ist oder anderweitig kenntlich gemacht wird, zu welchem Agenten sie gehört.

Die letzte von uns identifizierte Eigenschaft ist eine Farbgebung der Visualisierungen, bei der die entsprechenden Objekte sich nicht mehr genügend vom Hintergrund abheben. Insbesondere bei der Visualisierung durch Veränderungen der Textur, muss dies berücksichtigt werden. Wenn Q als farbliche Textur dargestellt wird und in dem aktuellen Zustand beispielsweise gelblich dargestellt wird, kann das aus dem Blickwinkel in Abb. 10 der Farbe des Bodens ähnlichsehen.
Aufgrund dieser Abwägung würden wir den Q-Value auf Standbildern eines festen Umgebungs-Zustands als Textur darstellen und im laufenden Spiel als Skala.

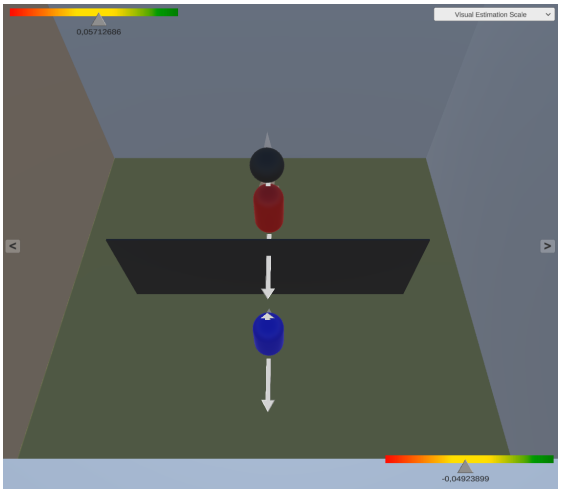

Fig. 10. Spielfeld Ansicht aus einem ungünstigen Blickwinkel

5.3 Übertragbarkeit auf andere Anwendungsfälle

In Kapitel 3 wurde die Anforderung definiert, dass sich unsere Visualisierungen in die RL-Umgebung visuell möglichst nahtlos einfügen sollen, damit sie intuitiv lesbar sind. Das hat den eindeutigen Nachteil, dass die Visualisierungen an jede Umgebung neu angepasst werden müssen. Wie ein Aspekt, wie z.B. eine Aktion oder eine Observation, im dreidimensionalen Raum verständlich dargestellt werden kann ist abhängig davon, wie diese Aspekte genau definiert sind. Ob es das wert ist, diesen Aufwand in Kauf zu nehmen, hängt unter anderem davon ab, wie viel lesbarer solche Visualisierungen tatsächlich verglichen mit den Diagrammen aus Kapitel 2 sind. Wie bereits in Kapitel 5.1 beschrieben wurde, müsste dies in einer Nutzerstudie untersucht werden.

6 Fazit

Diese Forschungsarbeit ist im Wesentlichen in zwei Teile unterteilt.

Im ersten Part wurden verschiedene Visualisierungsmethoden anhand von XRL-Forschungsprojekten vorgestellt. Auf diese Weise wurde ein Eindruck der aktuell eingesetzten Darstellungsformen vermittelt. Dabei wurde festgestellt, dass die überwiegende Mehrheit dieser Methoden Detailwissen zu den jeweils eingesetzten RL-Verfahren voraussetzt. Somit schließen sie Zielgruppen aus, die nicht über dieses Wissen verfügen. Zudem wurde bemerkt, dass die Saliency Map die einzige uns bekannte Visualisierungsform ist, bei der eine direkte, visuelle Verbindung zwischen der RL-Umgebung und den zusätzlichen Informationen hergestellt wird. Das kann die Übersichtlichkeit komplexer Informationen beeinträchtigen, da die Anwendungsumgebung als Referenzpunkt fehlt.

Im zweiten Part wurden eigene Formen der Visualisierung vorgestellt. Die dabei verfolgten Ziele wurden aus den erläuterten Lücken in den bestehenden Darstellungsformen abgeleitet. Das erste Ziel ist die intuitivere Lesbarkeit, die dadurch erreicht werden soll, dass die Visualisierungen in den Kontext der Lernumgebung optisch eingeordnet werden. Das zweite Ziel ist ein verbesserter Überblick über mehrere, in einem gemeinsamen Kontext einblendbare Visualisierungen. Es wurden Darstellungen entwickelt, die als Beispiel für diese neue Art von Visualisierungen dienen können. Als zu visualisierende Information wurden die Input-Zustände eines gelernten RL-Modells und der von Agenten geschätzten Q Value ausgewählt. Da eine Nutzerstudie zur Evaluation der Lesbarkeit notwendig ist, wurde ein Ansatz dazu erläutert, wie diese Studie aufgebaut werden könnte. Zudem wurden einige Eigenschaften erläutert, die die Visualisierungen erfüllen sollten, um den Zielen gerecht werden zu können.

References

1. Heuillet, A., Couthouis, F., Díaz-Rodríguez, N.: Explainability in deep reinforcement learning. Knowledge-Based Systems **214**, 106685 (2021). https://doi.org/https://doi.org/10.1016/j.knosys.2020.106685
2. Juozapaitis, Z., Koul, A., Fern, A., Erwig, M., Doshi-Velez, F.: Explainable reinforcement learning via reward decomposition. In: proceedings at the International Joint Conference on Artificial Intelligence. A Workshop on Explainable Artificial Intelligence. (2019), https://finale.seas.harvard.edu/publications/explainable-reinforcement-learning-reward-decomposition
3. Nikulin, D., Ianina, A., Aliev, V., Nikolenko, S.: Free-lunch saliency via attention in atari agents. In: 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW). pp. 4240–4249 (2019). https://doi.org/10.1109/ICCVW.2019.00522
4. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms. CoRR **abs/1707.06347** (2017), http://arxiv.org/abs/1707.06347
5. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Batra, D.P..D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. International Journal of Computer Vision **128**, 336–359 (2020). https://doi.org/https://doi.org/10.1007/s11263-019-01228-7
6. Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T., Simonyan, K., Hassabis, D.: A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. Science **362**(6419), 1140–1144 (2018). https://doi.org/10.1126/science.aar6404, https://science.sciencemag.org/content/362/6419/1140
7. Simonyan, K., Vedaldi, A., Zisserman, A.: Deep inside convolutional networks: Visualising image classification models and saliency maps. CoRR **abs/1312.6034** (2014)
8. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. The MIT Press, second edn. (2018)
9. Wang, J., Gou, L., Shen, H.W., Yang, H.: Dqnviz: A visual analytics approach to understand deep q-networks. IEEE Transactions on Visualization and Computer Graphics **25**(1), 288–298 (2019). https://doi.org/10.1109/TVCG.2018.2864504

10. Zai, A., Brown, B.: Einstieg in Deep Reinforcement Learning: KI-Agenten mit Python und PyTorch programmieren. Carl Hanser Verlag (2020)