Duale Hochschule Baden-Württemberg Mannheim

Dokumentation

A* Pathfinding Algorithmus

Studiengang Wirtschaftsinformatik

Studienrichtung Data Science

Verfasser:	Alexander Paul (5722106) Nicholas Link (3967704) Lucas Wätzig (8167493)
Kurs:	WWI-21-DSB
Studiengangsleiter:	Prof. Dr. Bernhard Drabant
Dozent:	Prof. Dr. Maximilian Scherer
Modul:	Integrationsseminar
Abgabedatum:	26.02.2024

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Die Visualisierung von Algorithmen spielt eine entscheidende Rolle bei der Vermittlung und dem Verständnis komplexer Konzepte in der Informatik und anderer Bereiche. Insbesondere bei der Pfadfindung, wie sie im A*Algorithmus verwendet wird, ist die Visualisierung ein wirksames Werkzeug, um die Funktionsweise des Algorithmus zu veranschaulichen und seine Anwendungen zu verdeutlichen. Diese Dokumentation gibt einen Überblick über die Grundlagen der Visualisierung, die Bedeutung der Visualisierung von Algorithmen sowie die Konzepte der Pfadfindung. Im Verlauf der Arbeit wird der A*Algorithmus genauer betrachtet, mit Augenmerk auf der Heuristik und Funktionsweise. Zudem werden verschiedene Anwendungsgebiete des A*Algorithmus beleuchtet, um aufzuzeigen, wie vielseitig dieses Konzept in der Praxis eingesetzt wird. Ein besonderer Fokus liegt auf der ausgewählten und umgesetzten Visualisierung des A*Pathfinding Algorithmus mithilfe von Manim, einem Werkzeug zur Erstellung von animierten mathematischen Inhalten. Somit ist es möglich einen Überblick über den A*Pathfinding Algorithmus, die Visualisierung und die Erstellung eines Erklärvideos zu erhalten.

2 Grundlagen

Visualisierungen spielen eine entscheidende Rolle bei der Vermittlung von komplexen Inhalten. Die Darstellung von Algorithmen ermöglicht es, deren Konzepte auf eine leicht verständliche Weise zu vermitteln. Visualisierungstechniken und -methoden geben dabei einen Einblick in bewährte Praktiken. Der folgende Abschnitt beleuchtet die theoretischen Grundlagen der Visualisierung und hebt bewährte Methoden hervor. Des Weiteren wird sich mit dem Thema Pfadfindung im Allgemeinen beschäftigt.

2.1 Visualisierung

Die Fähigkeit zu sehen und die komplexe Funktion der Augen sind faszinierende Aspekte der menschlichen Anatomie. Die Augen ermöglichen es dem Menschen, visuelle Dinge aufzunehmen und sind entscheidend für die Art und Weise, wie man Informationen interpretiert. Ein Verständnis von Visualisierungen und deren Rolle für die menschliche Wahrnehmung und Kommunikation ist daher essenziell.

Visualisierung bezieht sich auf den Prozess der Darstellung von Informationen oder Daten durch bildsprachliche Zeichen. Es geht darum, Informationen zu ergänzen oder sie ganz in die Bildsprache zu übersetzen, um komplexe Konzepte, Muster oder Zusammenhänge leicht verständlich und ansprechend zu präsentieren. Diese visuellen Darstellungen können Diagramme, Grafiken, Cartoons, Tabellen, Karten, Animationen oder andere Formen von Bildern sein (vgl. Zentrum für Schulqualität und Lehrerbildung (ZSL) o. J.; Schumann 2000). Im Kontext der Visualisierung von Informationen und dem Verständnis ihrer Bedeutung für Kommunikation und das Lernen ist es essenziell, die verschiedenen Arten genauer zu betrachten. Diese Visualisierungsarten dienen nicht nur dazu, komplexe Konzepte verständlicher zu gestalten, sondern fördern auch die aktive Beteiligung der Teilnehmer und ermöglichen ein tieferes Verständnis der behandelten Themen. Es gibt verschiedene Arten von Visualisierung, die sich in zwei Haupttypen differenzieren lassen. Statische Visualisierungen sind Abbildungen, die keine Animationen oder Bewegungen enthalten. Es ist nur eine Ansicht der Abbildungen und keine Interaktion mit den Inhalten möglich. Dynamische Visualisierungen hingegen bieten dem Nutzer die Möglichkeit, mit der Abbildung zu interagieren. Es lassen sich Bewegungen und Veränderungen in den Abbildungen darstellen. Eine

Sonderform dieser Darstellungsart ist die interaktive Visualisierung, wodurch Nutzer in der Lage sind, durch die Daten zu navigieren, sie zu filtern und zu manipulieren, um ein tieferes Verständnis zu erlangen. Die 3D-Visualisierung verwendet dreidimensionale Modelle, um komplexe Objekte darzustellen, und ermöglicht es dem Anwender, Objekte aus verschiedenen Blickwinkeln zu betrachten (vgl. Amazon Web Services 2024; e-teaching.org 2016).

Neben den Visualisierungsarten gibt es viele verschiedene Methoden, um Visualisierungen darzustellen. Unter den zahlreichen Visualisierungsmethoden sind Diagramme, Tabellen, Graphen, Karten, Dashboards, Mind Mapping, Wandzeitungen und Bilder von besonderer Bedeutung. Diagramme bieten eine grafische Darstellung von Daten, die es ermöglichen, komplexe Informationen auf einen Blick zu erfassen. Tabellen strukturieren Daten in Zeilen und Spalten, um einen klaren Vergleich und eine Analyse zu erleichtern. Graphen visualisieren Beziehungen zwischen Entitäten und stellen komplexe Netzwerke dar. Karten zeigen geografische Informationen und unterstützen die Navigation sowie räumliche Analysen. Dashboards bieten eine übersichtliche Anzeige von Kennzahlen und Echtzeitdaten. Mind Mapping ist eine kreative Technik zur Darstellung von Ideen und Konzepten in Form eines Diagramms. Diese Form gilt als eine häufig verwendete Methode der Visualisierung. Entwickelt wurde sie von Tony Buzan. Dabei wird ein zentrales Thema in der Mitte einer leeren Seite notiert, um das herum Schlüsselwörter und Assoziationen in Form von Ästen und Verzweigungen skizziert werden. Diese visuelle Darstellung erlaubt es, komplexe Zusammenhänge und Ideen übersichtlich zu strukturieren und fördert gleichzeitig kreatives Denken und Assoziationsvermögen (vgl. Buzan und Buzan 2011). Wandzeitungen sind großformatige, visuelle Darstellungen von Informationen, die von Gruppen erstellt werden, um komplexe Themen zu präsentieren und den Austausch zu fördern. Bilder dienen als visuelle Repräsentationen, um Informationen auf anschauliche Weise zu vermitteln und sind ein wesentlicher Bestandteil vieler Visualisierungsmethoden.

Diese verschiedenen Methoden der Visualisierung bieten vielfältige Möglichkeiten, komplexe Informationen verständlich darzustellen, den Austausch zwischen Teilnehmern zu fördern und das Lernen zu erleichtern. Durch ihren Einsatz können Kommunikation und Zusammenarbeit effektiv unterstützt werden (vgl. Wilbert 2024; Tableau Software 2024).

Nachdem die grundlegenden Konzepte der Visualisierung und ihre Bedeutung für die Kommunikation und das Verständnis von Informationen erläutert wurden, ist es wichtig, auch

die Grundlagen der Computergrafik und Animationstechniken zu betrachten. Diese spielen eine entscheidende Rolle bei der Erstellung von visuellen Darstellungen und tragen wesentlich dazu bei, komplexe Konzepte anschaulich zu vermitteln. Die Grundlagen der Computergrafik und Animationstechniken umfassen verschiedene Aspekte, darunter Rastergrafiken, Vektorgrafiken, 3D-Modellierung, Texturierung, Beleuchtung, Rendering, Schlüsselbildanimation und Bewegungspfade. Rastergrafiken basieren auf Pixeln und eignen sich gut für Fotos und komplexe Grafiken, während Vektorgrafik auf mathematischen Formeln basiert und sich besser für Illustrationen und Logos eignet (vgl. Adobe 2023). Die 3D-Modellierung erstellt, wie die Bezeichnung schon vermuten lässt, dreidimensionale Objekte, während die Texturierung ihnen realistische Oberflächen verleiht. Beleuchtung spielt eine wichtige Rolle bei der Schaffung von Schatten, Stimmungen und Atmosphären in einer Szene, während Rendering den Prozess der Erzeugung eines endgültigen Bildes oder Films aus einem 3D-Modell beschreibt. Schlüsselbildanimation beinhaltet das Festlegen wichtiger Positionen und Bewegungen eines Objekts, während Bewegungspfade die Bahn beschreiben, entlang derer sich ein Objekt bewegt (vgl. Technologies 2024; GeeksforGeeks 2022).

Ein Ausblick auf zukünftige Trends und Entwicklungen zeigt, dass Virtual Reality (VR) und Augmented Reality (AR) immer wichtiger werden. VR ermöglicht es Benutzern, in virtuelle Welten einzutauchen, während AR digitale Inhalte in die reale Welt integriert (vgl. Splunk 2024). Diese Technologien haben das Potenzial, viele Bereiche zu revolutionieren, von der Unterhaltung über die Bildung bis hin zur Medizin. Darüber hinaus werden visuelle Effekte wie Motion Graphics, Visual Effects (VFX) und Computer Generated Imagery (CGI) immer realistischer und beeindruckender. Diese Effekte werden vor allem in Filmen, Werbung, Videospielen und anderen Medien eingesetzt, um visuelle Erlebnisse zu schaffen, die die Zuschauer faszinieren und beeindrucken (vgl. Okereke 2023; Makarov 2024).

2.2 Visualisierung von Algorithmen

Die Visualisierung von Algorithmen spielt eine entscheidende Rolle bei der Veranschaulichung ihrer Funktionsweise und der Analyse ihres Verhaltens. Um Algorithmen zu visualisieren, gibt es verschiedene Möglichkeiten.

Eine häufig verwendete Methode zur Visualisierung von Algorithmen ist das Flussdiagramm. Ein Flussdiagramm ist eine grafische Darstellung eines Algorithmus, die aus

verschiedenen symbolischen Formen wie Rechtecken, Rhomben und Pfeilen besteht. Jedes Symbol repräsentiert eine bestimmte Aktion oder Entscheidung, während die Pfeile den Fluss der Ausführung des Algorithmus darstellen (vgl. Lucidchart 2024). In Anhang A1 ist die Visualisierung des A* Pathfinding Algorithmus als Flussdiagramm dargestellt und zeigt eine Variante, den Algorithmus zu visualisieren. Eine weitere Methode zur einfachen Darstellung von Algorithmen ist die Verwendung von Pseudocode. Diese ist eine informelle Art der Programmierung, die die Struktur eines Algorithmus mit natürlicher Sprache beschreibt, ohne die genaue Syntax einer bestimmten Programmiersprache zu verwenden. Pseudocode ermöglicht es, den Algorithmus in einer allgemein verständlichen Form darzustellen, was das Verständnis erleichtert (vgl. studyflix 2024). In Anhang A2 befindet sich ebenfalls ein Beispiel für Pseudocode für den A* Algorithmus. Die Darstellung von Algorithmen mit Hilfe von Graphen ist eine weitere Methode. In der Praxis wird dazu häufig die Graphentheorie angewendet, um die Datenstrukturen und den Ablauf des Algorithmus zu veranschaulichen. Graphen bestehen aus Knoten, die die Elemente darstellen, und Kanten, die die Beziehungen zwischen den Elementen repräsentieren. Die Art der Visualisierung ermöglicht es, komplexe Algorithmen auf eine intuitive Weise zu erfassen und erleichtert die Analyse ihres Verhaltens (vgl. Bröcker 1999, S. 17-18). Aufgrund dieser Eigenschaften wurde diese Art der Visualisierung für dieses Projekt verwendet.

Die verschiedenen aufgezeigten Methoden zur Visualisierung von Algorithmen bieten eine effektive Möglichkeit, ihre Funktionsweise einfach und verständlich zu kommunizieren. Durch die Verwendung von Flussdiagrammen, Pseudocode, etc. können Entwickler komplexe Algorithmen besser analysieren, optimieren und helfen bei der Fehlersuche.

2.3 Pfadfindung

Pfadfindung ist in der Informatik eine grundlegende Problemstellung. Sie befasst sich mit der Auswahl des optimalen (kürzesten oder effizientesten) Wegs zwischen einem Startpunkt und einem Zielpunkt in einem gegebenen Raum oder einer Umgebung. Diese Problemstellung findet unter anderem in Computerspielen, Robotik oder auch in Navigationssystemen Anwendung (vgl. Ionos 2023). Pfadfindungsalgorithmen stehen verschiedenen Herausforderungen gegenüber, darunter unvorhersehbare Hindernisse, dynamische Umgebungen, Berücksichtigung von Kosten oder Prioritäten, Mehrfachziele und die

Gewährleistung von Effizienz und Genauigkeit. Dynamische Umgebungen erfordern eine kontinuierliche Aktualisierung des Pfades, um Änderungen wie sich bewegende Hindernisse oder veränderte Bedingungen zu berücksichtigen. Die Berücksichtigung von Kosten erfordert die Anpassung des Pfades basierend des zugrunde liegenden Kriteriums, wie beispielsweise die Minimierung der Zeit, der Weglänge oder andere Ressourcen. Bei der Navigation mit mehreren Zielen erfordert die Planung von Pfaden das Berücksichtigen einer bestimmten Reihenfolge oder unterschiedliche Prioritäten. Effizienz und Genauigkeit sind essenziell, um Algorithmen für Echtzeit- oder zeitkritische Anwendungen geeignet zu machen, während gleichzeitig eine annehmbare Lösung für das zugrundeliegende Problem gefunden wird (vgl. Alexander Thamm GmbH o. J.).

Pfadfindungsalgorithmen lassen sich nach verschiedenen Kriterien klassifizieren. Darunter zählen die Anzahl der Quellen, die Art und Weise der Suche nach dem optimalen Pfad und die Verwendung von Heuristiken. Sogenannte 'Single Source Algorithmen' ermöglichen das Finden eines Pfades von einem einzelnen Startpunkt zu einem Zielpunkt, während mehrfache Quell-Pfadfindungsalgorithmen die Suche von mehreren Start- und Zielpunkten unterstützen. Eine weitere Differenzierung ist die zwischen uninformierter und informierter Suche. Uninformierte Algorithmen sind auf die Informationen der jeweiligen Umgebung beschränkt, während informierte Algorithmen zusätzliche heuristische Informationen mit einbeziehen können. Heuristische Algorithmen, wie zum Beispiel der A* Algorithmus, verwenden Schätzungen, um die Effizienz der Suche zu verbessern. Neben diesen garantieren exakte Algorithmen, wie der Dijkstra-Algorithmus, dass die optimale Lösung gefunden wird. Diese sind jedoch in der Regel mit höherem Rechenaufwand verbunden. Diese Klassifizierung bietet einen Rahmen für die Auswahl des am besten geeigneten Algorithmus passend zu den Anforderungen und Eigenschaften der Anwendung (vgl. Neo4j o. J.; Carnegie Mellon University 2017).

3 A* Pathfinding Algorithmus

Der A* Algorithmus ist ein intelligenter Wegfindungs- oder auch Suchalgorithmus, welcher es ermöglicht, den effizientesten Weg zwischen zwei Punkten in einem Graphen zu finden. Der Algorithmus gehört zu den informierten Suchalgorithmen, das heißt, es wird eine Schätzfunktion bzw. Heuristik verwendet, um zielgerichtete Suchen durchzuführen. Der Algorithmus wurde 1968 von Peter Hart, Nils Nilsson und Bertram Raphael beschrieben und wird auch als Erweiterung und Verallgemeinerung des Djikstra-Algorithmus gesehen. Das Hauptziel des A* Algorithmus ist es, den Pfad mit den geringsten Kosten zwischen einem Startpunkt und einem Zielpunkt zu finden. Im folgenden Kapitel wird die Heuristik, die Funktionsweise und die Anwendungsgebiete des A* Pathfinding Algorithmus genauer beschrieben (vgl. Russell und Norvig 2022, S. 103-107; Hart et al. 1968).

Der A* Algorithmus ist ein intelligenter Wegfindungs- oder auch Suchalgorithmus, welcher es ermöglicht, den effizientesten Weg zwischen zwei Punkten in einem Graphen zu finden. Der Algorithmus gehört zu den informierten Suchalgorithmen, das heißt, es wird eine Schätzfunktion bzw. Heuristik verwendet, um zielgerichtete Suchen durchzuführen. Der Algorithmus wurde 1968 von Peter Hart, Nils Nilsson und Bertram Raphael beschrieben und wird auch als Erweiterung und Verallgemeinerung des Djikstra-Algorithmus gesehen. Das Hauptziel des A* Algorithmus ist es, den Pfad mit den geringsten Kosten zwischen einem Startpunkt und einem Zielpunkt zu finden. Im folgenden Kapitel wird die Heuristik, die Funktionsweise und die Anwendungsgebiete des A* Pathfinding Algorithmus genauer beschrieben (vgl. Russell und Norvig 2022, S. 103-107; Hart et al. 1968).

3.1 Heuristik

Zu Beginn gilt es zu klären, was eine Heuristik ist. Eine Heuristik ist eine Theorie oder Strategie, welche mit Hilfe von Erfahrung oder Intuition eine Entscheidung erlangt, ohne alle Möglichkeiten zu prüfen. Dies führt zu einem ressourcensparenden Vorgehen beim Ziehen einer Schlussfolgerung (vgl. Thommen und Siepermann 2018; Spektrum 2000). Im Kontext des A* Pathfinding Algorithmus bezieht sich die Heuristik auf die geringsten Kosten zwischen zwei Punkten. Kosten können in diesem Fall verschiedene Kriterien wie Entfernung,

Zeit oder Ressourcenverbrauch sein. Ausgewählt wird der Pfad, der die Kostenfunktion des Algorithmus minimiert. Mathematisch ausdrücken lässt sich diese mit der Formel:

$$f(x) = g(x) + h(x)$$

Wobei x ein spezifischer Knoten des Pfades ist. Die Kosten, die benötigt werden, um x vom Startknoten aus zu erreichen werden durch g(x) repräsentiert. Zusätzlich dazu werden die geschätzten Kosten (Heuristik) vom Knoten x zum Zielknoten durch h(x) mit einbezogen. Dadurch lenkt der Algorithmus seine Suche fast immer in die grobe Richtung des Ziels. Dies vermeidet unnötiges Erkunden im übrigen Graphen, da der A* Algorithmus priorisiert Pfaden folgt, die wahrscheinlich zu einer optimalen Lösung führen. Das heißt, wenn die Heuristik zulässig ist, garantiert der A* Algorithmus die Optimallösung, also den kostengünstigsten Weg von Start- zu Zielknoten. Zulässige Heuristik bedeutet, dass sie die tatsächlichen Kosten für die Erreichung des Ziels niemals überschätzt werden. Passiert dies doch spricht man von einer „nicht zulässigen“ Heuristik. Dabei ist die Garantie auf die Optimallösung nicht mehr gegeben, der Algorithmus kann jedoch trotzdem funktionieren und einen Pfad zum Ziel finden, mit der Gefahr, dass er möglicherweise nicht den kürzesten Pfad findet (vgl. Cui und Shi 2010; Hart et al. 1968; Russell und Norvig 2022, S. 103-107; Belwariar 2016).

Die Wahl der Heuristik h(x) im A* Algorithmus kann immer unterschiedlich sein und hängt von dem zugrundliegenden Problem ab. Die bekannteste und meist verwendetste Heuristik ist die euklidische Distanz. Definiert ist diese als der gemessene Abstand zweier Punkte mit einer Verbindungsstrecke im euklidischen Vektorraum. Mathematisch kann diese mit folgender Formel ausgedrückt werden:

$$d = \sqrt{(p-q)^2}$$

Die Nutzung der euklidischen Distanz eignet sich vor allem, wenn der Algorithmus den besten Pfad in einer Umgebung finden soll, in der diagonale Bewegungen möglich sind (vgl. Spektrum 2017). Eine alternative Methode besteht in der Nutzung der Manhattan-Metrik. Diese wird als die Summe der absoluten Differenz zwischen zwei Punkten definiert. Im Gegensatz zur euklidischen Distanz bietet die Manhattan-Distanz einen Vorteil in Umgebungen, in denen Bewegungen primär horizontal oder vertikal erfolgen. Dies ist insbesondere in kontextuellen Situationen von Bedeutung, beispielsweise bei der Navigation innerhalb

eines zweidimensionalen Raster- oder Gittern-basierten Systems wie es typischerweise bei der Kartennavigation angewandt wird (vgl. Einsiedler und Wieser 2019). Weitere Möglichkeiten zur Berechnung die Chebyshev-Distanz sowie der Dijkstra-Algorithmus oder das Nutzen von Erfahrungswerten und spezifischer Erkenntnisse (vgl. Woltmann 2021).

Eine wichtige Eigenschaft bei der Betrachtung von Algorithmen sowie deren Performance ist die Zeitkomplexität, also die theoretische Zeit, die der Algorithmus zur Berechnung oder Abarbeitung benötigt. Diese wird in der Literatur mit Hilfe der O-Notation beschrieben (vgl. StudySmarter 2024). Die Zeitkomplexität des A*Algorithmus hängt von verschiedenen Faktoren ab, einschließlich der Größe des Suchraums, der gewählten Heuristik und der Struktur des Graphen. Aus diesem Grund wird die Zeitkomplexität durch vereinfachende Annahmen geschätzt und ist von der verwendeten Heuristik und dessen Ergebnis sowie der Nutzung der Open- und der Closed-List stark abhängig. Ist die Heuristik genau berechnet, werden weniger Knoten durchsucht und die Laufzeit verbessert sich. Das Arbeiten mit Listen kann durch unterschiedlich verwendete Datenstrukturen optimiert werden. Daraus ergibt sich eine Worst-Case Laufzeit von $O(n^2)$ und im Average-Case von $O(n*\log(n))$, wobei n die Anzahl der Knoten im Graphen ist (vgl. Woltmann 2021). Beim Worst-Case handelt es sich um den schlechtesten ausgeführten Fall des Algorithmus, das heißt, es wird eine maximale Anzahl an Operationen durchgeführt. Im Average-Case geht man vom Durchschnitt mehrerer Fallanalysen aus, das heißt, man erhält die Laufzeit, die der Algorithmus im Normalfall benötigt (vgl. GeeksforGeeks 2012). Der Algorithmus kann jedoch dann sehr effizient sein, wenn eine für den spezifischen Fall gute Heuristik verwendet wurde.

3.2 Funktionsweise

Um den optimalen Pfad, beispielsweise in einem Netzwerk, zu finden, führt der A* Pathfinding Algorithmus eine Prioritätswarteschlange, häufig bezeichnet als ‚Open List'. In dieser werden alle betrachteten Punkte bzw. Knoten mit verschiedenen Zusatzinformationen notiert. Einerseits wird die Information gespeichert, welcher Knoten der Vorgänger war, also von welchem Knoten aus man den aktuellen erreicht hat. Des Weiteren wird der Wert der Kosten, die benötigt werden, um vom Startknoten aus bis zu dem entsprechenden Knoten zu kommen, gespeichert. Dieser Wert entspricht dem g(x) aus der Kostenfunktion. Außerdem wird auch der f(x)-Wert, also der Wert der bisher gegangenen Kosten (g(x)) addiert

mit den voraussichtlich noch zu gehenden Kosten (h(x)), gespeichert. Zusätzlich gibt es auch eine Liste bzw. einen Ablagestapel bezeichnet als ‚Done List', auf dem alle Knoten gespeichert werden, deren Nachbarn bereits betrachtet und in der ‚Open List' notiert wurden (vgl. Cox 2020; Pfeiffer 2022; Swift 2017).

Zu Beginn wird dem Algorithmus der Startknoten übergeben. Dieser wird mit den entsprechenden Werten in der Prioritätswarteschlange bzw. der ‚Open List' gespeichert. Im Fall des Startknotens wird beim Wert des Vorgängerknotens und beim Wert der bisherigen Kosten nichts eingetragen. Lediglich der Wert der Heuristik wird eingetragen, welcher schätzt wie hoch die Kosten sind, um den Zielknoten zu erreichen. Da mit dem Startknoten der erste Knoten in der ‚Open List' eingetragen ist, kann die erste Iteration beginnen. Der Startknoten steht im Fokus. In allen Iterationen wird jeder Nachbar des im Fokus stehenden Knoten betrachtet und mit den genannten Werten in der ‚Open List' gespeichert. Diese wird anschließend nach dem Wert der Heuristik aufsteigend sortiert, sodass sich immer der Knoten mit den geringsten noch zu gehenden Kosten an erster Stelle der Liste befindet.

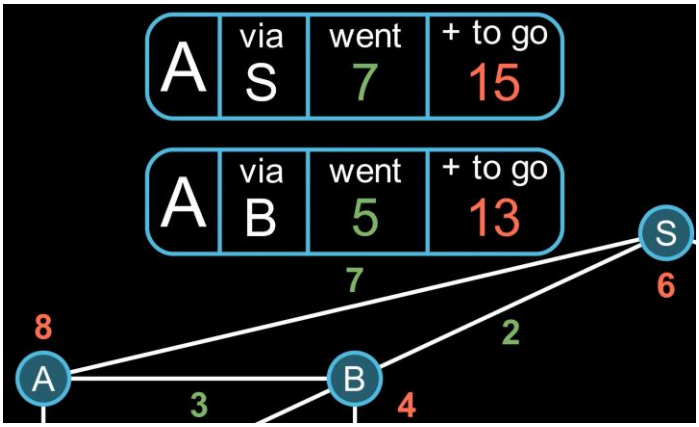


Abbildung 1: Beispiel für Vergleich der Werte eines Knoten über verschiedene Wege (Quelle: eigene Abbildung)

Wurden alle Nachbarn des im Fokus stehenden Knoten betrachtet und gespeichert, wird dieser in die ‚Done List' verschoben. Daraufhin rückt der nächste Knoten der ‚Open List' in den Fokus und eine neue Iteration mit denselben Schritten beginnt.

Befindet sich ein zu betrachtender Knoten bereits in der ‚Open List', werden am Ende die geringeren Werte gespeichert. Sollten also beispielsweise vom Startknoten (S) aus zu Knoten A Wegkosten von 7 und Gesamtkosten von 15 gespeichert sein, man aber von Knoten B aus mit Wegkosten von 5 zu A kommt und somit die Gesamtkosten bei 13 liegen, so werden die Werte des Weges über Knoten B behalten und die andern verworfen.

Die Iterationen, bei denen alle Nachbarn des jeweils im Fokus stehenden Knoten betrachtet werden, unterbrechen erst, wenn der Endknoten in den Fokus rückt. Tritt dieser Fall ein, nimmt der Algorithmus die Werte des Endknotens und liest aus diesen den Knoten, welcher als Vorgänger angegeben ist. Daraufhin werden die Werte dieses Vorgängerknotens aus der Ablageliste gezogen und wieder betrachtet, welcher sein Vorgängerknoten war. So wird nun rückwärts durch den finalen Weg iteriert, bis man wieder den Startknoten aus der Ablageliste gezogen hat. Alle gezogenen Knoten bilden dann den finalen Weg. Diese Rekonstruktion ist wichtig, da in manchen Fällen auch Knoten in der Ablageliste landen könnten, die am Ende gar nicht Teil des finalen Pfades sind (vgl. Cox 2020; Patel 2023; Patel 2014; Pfeiffer 2022; Swift 2017).

Der Grund, warum dieser Ablauf des Algorithmus so effizient ist, liegt darin, dass zum einen der Vorteil des Dijkstra-Algorithmus durch die Priorisierung von kosteneffizienten Wegen einbezogen wird, gleichzeitig allerdings auch eine Schätzung einbezogen wird, wie weit es noch zum Ziel ist. Durch genau diese Schätzung läuft der Algorithmus primär fast immer in die grobe Richtung des Ziels und konzentriert sich bei seiner Erkundung auf die vielversprechendsten Pfade (vgl. Cox 2020; Foead et al. 2021; Patel 2023; Patel 2014; Pfeiffer 2022; Swift 2017).

3.3 Anwendung

Der Einsatz des A*Pfadfindung Algorithmus ist sehr vielseitig und erstreckt sich über verschiedene Gebiete. Diese reichen von Unterhaltung bis hin zu hochtechnologischen Gebieten.

In Computerspielen dient der A* Pathfinding als Grundlage zur Navigation von Charakteren und Objekten innerhalb des Spiels. In den ersten Versionen des Spiels PacMan wurde der Algorithmus verwendet, um die Bewegungen der Geister zu steuern. Darüber hinaus basiert die Steuerung von sogenannten Nicht-Spieler-Charakteren (NPCs) in beispielsweise Ego-Shootern auf dem A*Pathfinding Algorithmus und ermöglicht das taktische Manöver und intelligente Bewegungen (vgl. Candra et al. 2021; Ionos 2023). Im Bereich Transport und Logistik spielt die Wegfindung eine zentrale Rolle beispielsweise für bei der Routenplanung von Fahrzeugen oder das Verfolgen und Optimieren von Lieferketten. In Anwendungen wie Google Maps oder Apple Karten dient der Algorithmus dazu, den optimalen Weg

von einem Startpunkt zu einem Zielpunkt zu berechnen, unter Berücksichtigung von Verkehrsbedingungen. Zusätzlich dazu wird der Algorithmus in der Lieferkettenüberwachung eingesetzt, um den Transport von Waren zu überwachen und zu optimieren. Auch im Bereich der Paketzustellung findet der A* Anwendung, um effiziente Routen für die Auslieferung von Paketen zu planen und die Zustellung optimieren (vgl. Ionos 2023). Das sogenannte *Traveling Salesman Problem* ist ein prominentes Beispiel hierfür (vgl. Karimi 2021). In der Robotik und weiteren Technologien spielt der A* Pathfinding Algorithmus eine Schlüsselrolle bei der Steuerung autonomer Systeme. Hier kommt der Algorithmus zum Einsatz, um Routen durch Netzwerke und Telekommunikationssysteme zu planen, wodurch Datenpakete effizient übertragen werden können. Darüber hinaus wird der A* in der Robotik eingesetzt, um Bewegungen von Roboterarmen zu steuern und komplexe Aufgaben auszuführen, wie zum Beispiel bei chirurgischen Eingriffen (vgl. Rubio 2023; Raza et al. 2018).

4 Visualisierung des A* Pathfinding Algorithmus

Aufgrund der Komplexität Algorithmen für Anfänger und nicht Informatiker einfach und verständlich zu erklären, hat man sich in dem erstellten Video dafür entschieden, ein umfangreiches Beispiel visuell darzustellen und daran den Ablauf des A* Pathfinding Algorithmus zu erklären. Das gewählte Beispiel beinhaltet einen Graphen mit 13 verschiedenen Knoten mit einzelnen Buchstaben als Bezeichnungen, darunter der Startknoten „S“ und der Endknoten „E“. Die Knoten sind mit insgesamt 18 verschiedenen Kanten bzw. Wegen untereinander verbunden, wobei es keine direkte Verbindung zwischen dem Start- und dem Endknoten gibt. Zusätzlich erhält jede Linie einen Wert, welcher die Kosten repräsentiert, die man beim Gehen dieses Weges auf sich nimmt, was dem g(x) aus der Kostenfunktion entspricht. Diese Werte wurden Grün eingefärbt, um sie von den Heuristik-Werten zu unterscheiden. Jeder einzelne Knoten erhält zusätzlich einen Heuristik-Wert, welcher in rot eingefärbt wurde. Diese roten Werte entsprechen dem h(x) der Kostenfunktion und sind eine optimistische Schätzung, wie hoch die Kosten vom entsprechenden Knoten bis zum Endknoten sind. Vor der Visualisierung des Algorithmus wird im Video die Kostenfunktion erklärt. Die grüne und rote Farbe der Zahlen ist an die Farben aus der Erklärung angepasst, um so auch visuell ein leichteres Verständnis für die Zuschauer zu schaffen und den Überblick im Beispiel zu erleichtern.

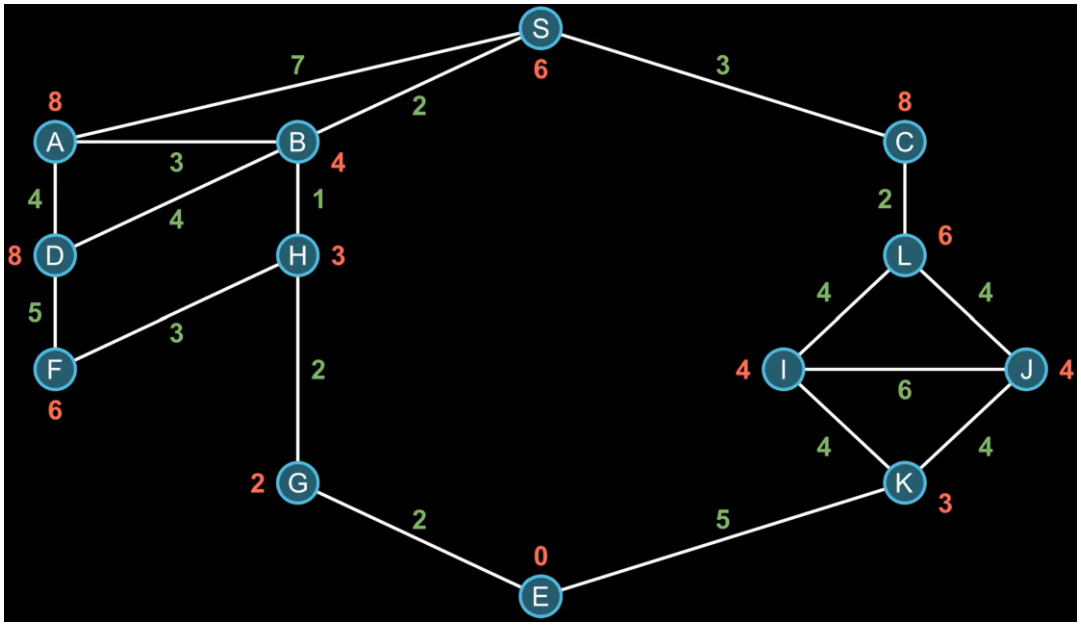


Abbildung 2: Der verwendete Graph zur Visualisierung (Quelle: eigene Abbildung)

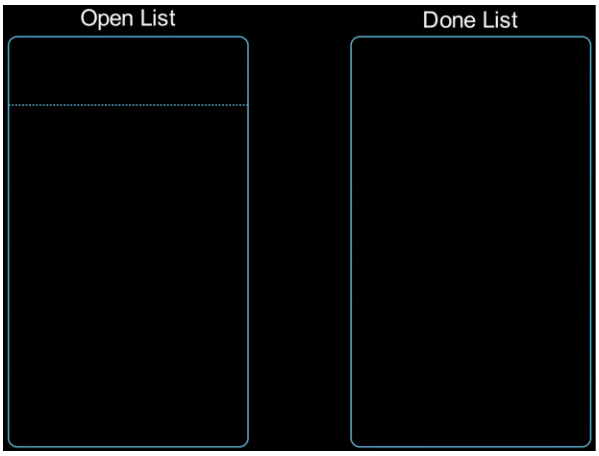


Abbildung 3: Visualisierung der ‚Open List' und ‚Done List' (Quelle: eigene Abbildung*)*

Im nächsten Schritt wurde dann der A* Algorithmus auf diesem Graphen angewendet, um den optimalen Weg von einem Startknoten zu einem Zielknoten zu finden. Dabei wurde jeder Schritt des Algorithmus genau erklärt und visualisiert, angefangen bei der Initialisierung der ‚Open List' und der ‚Done List'. Anders als bei der ‚Done List' wurde bei der ‚Open List' eine gestrichelte Linie eingefügt, um dem Zuschauenden während der Iteration des Algorithmus visuell darzustellen, welcher Knoten gerade im Fokus steht.

Um die Speicherung der einzelnen Knoten und der Werte visuell darzustellen, wurde ein Muster-Label erstellt, welches die entsprechenden Informationen der Knoten anzeigt und für den Zuschauer oder die Zuschauerin klar verständlich darstellt. Zu den Informationen gehört die Bezeichnung des betrachteten Knotens, sein Vorgänger, die Kosten des bereits

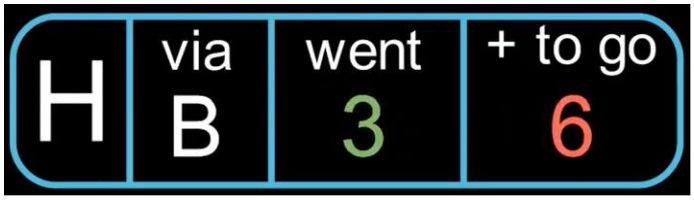


Abbildung 4: Beispiellabel des Knotens H mit entsprechenden Werten (Quelle: eigene Abbildung)

gegangenen Weges sowie der Wert der Kostenfunktion, also die Kosten bis zum Knoten, addiert mit den geschätzten Kosten bis zum Ziel. Entsprechend des Ablaufs des A* Algorithmus werden dann nach und nach die einzelnen Labels der Knoten erstellt, befüllt und verschoben. Mithilfe dieser Labels können dann die Knoteninformationen entsprechend der Regeln des Algorithmus in den Listen visuell angezeigt und verschoben werden, wodurch der Ablauf des Algorithmus visuell nachvollzogen werden kann.

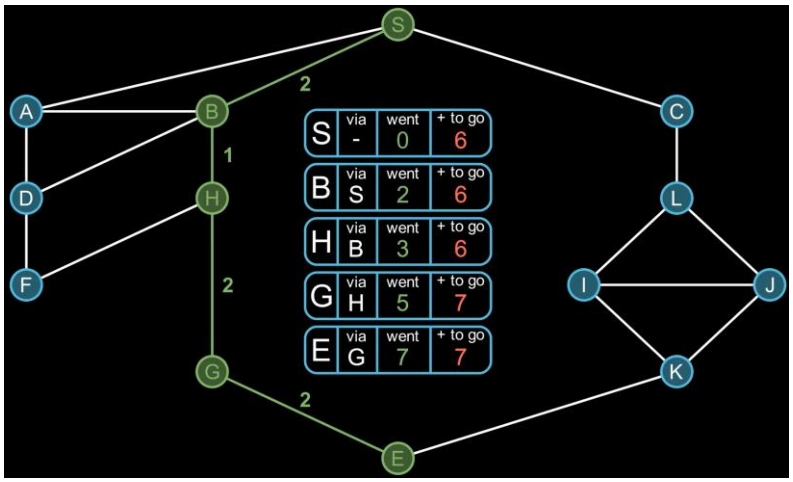


Abbildung 5: Hervorhebung des finalen Pfades (Quelle: eigene Abbildung)

Zum Abschluss wurde dann der finale Weg noch einmal visuell durch Einfärbung vom Rest des Graphen hervorgehoben, um ein abschließendes Verständnis des gegangenen Pfades beim Zuschauer zu erzeugen.

5 Schlussfolgerung

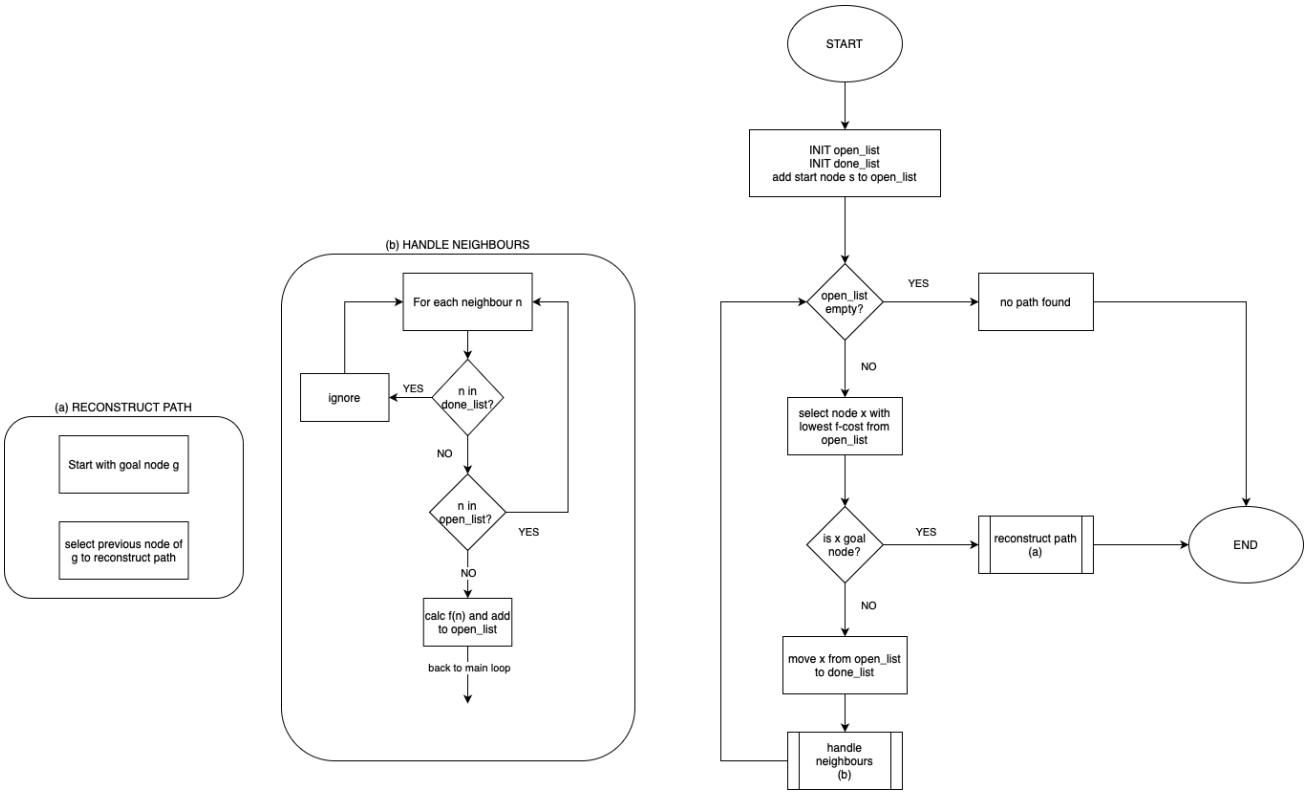
Die Arbeit beschäftigte sich mit der umfassenden Untersuchung des A* Pathfinding Algorithmus. Dabei wurden die Grundlagen der Visualisierung und des Computer Vision beleuchtet, um ein fundiertes Verständnis für die Arbeitsweise und die Anwendungsmöglichkeiten des Algorithmus zu schaffen. Im Zentrum dieser Untersuchung stand die umfassende Analyse der Funktionsweise, Eigenschaften und Besonderheiten des Algorithmus. Es wurde deutlich, dass dieser eine effiziente und effektive Methode zur Bestimmung des optimalen Wegs darstellt, indem er eine Heuristik verwendet, um in komplexen Suchräumen zielsicher den optimalen Pfad zu ermitteln. Des Weiteren ist die Flexibilität des Algorithmus hervorzuheben, die es ermöglicht, ihn in einer Vielzahl von Anwendungsgebieten einzusetzen. Von der Routenplanung bis hin zu Robotik und Computerspielen.

In dem erstellten Erklärvideo wurden die wichtigsten Punkte des A* Pathfinding Algorithmus behandelt und mit Hilfe von Graphen und einem Beispiel geeignet visualisiert dargestellt.

Trotz dieser Leistungsfähigkeit des A* Algorithmus bleibt Raum für weiterführende Forschungen und Entwicklungen. Optimierungspotenzial lässt sich im Bereich der Heuristiken finden, die ausschlaggebend für die Leistung des Algorithmus sind. Darüber hinaus bieten hybride Ansätze, die A* mit anderen Technologien kombinieren, weiteren Potenzial zur Leistungssteigerung. Die Integration von beispielsweise maschinellem Lernen und Künstlicher Intelligenz eröffnen Perspektiven für noch besser Anpassungen an die Umgebung, in der der Algorithmus operieren soll. Somit bleibt der A* Pathfinding Algorithms ein spannendes Forschungsgebiet mit großem Potenzial.

Anhang

A1. Beispiel für ein Flowchart anhand des A* Pathfinding Algorithmus



Flowchart am Beispiel des A* Pathfinding Algorithm (eigene Abbildung)

A2. Pseudocode für den A* Pathfinding Algorithmus

```
// initialize open and closed list
let openList empty list
let doneList empty list

// add start node to openList
add startNode to openList

// iterate through nodes
while openList is not empty

    // get focusNode
    let the focusNode = node with least f(x) value
    remove the focusNode from the openList
    add the focusNode to the doneList

    // check if endNode is reached
    if focusNode is endNode
        break the loop -> reconstruct path

    // get neighbour nodes of focusNode
    let neighbours of focusNode equal to adjacent nodes

   //
    for each neighbour

        // check neighbour already processed
        if neighbour is already in doneList
            pass this neighbour and continue with next one

        // calculate cost for each neighbour
        neighbour.g = focusNode.g + distance between neighbour
        and focusNode
        neighbour.h = estimated distance from neighbour to end
        neighbour.f = neighbour.g + neighbour.h

        // neighbour already in openList
        if neighbour aleardy in openList
            if neighbour.g is greather than openList-node's g
                pass this neighbour and continue next one

        // add the neighbour to openList
        add the neighbour to openList
```

Pseudocode am Beispiel des A* Pathfinding Algorithm (eigene Abbildung nach Swift 2017)

6 Literaturverzeichnis

Adobe (2023): Rastergrafiken: Was sie ausmacht und wofür man sie nutzt. Online verfügbar unter https://www.adobe.com/de/creativecloud/design/discover/raster-graphic.html, zuletzt aktualisiert am 18.02.2024, zuletzt geprüft am 18.02.2024.

Alexander Thamm GmbH (o. J.): Pathfinding. Online verfügbar unter https://www.alexanderthamm.com/de/data-science-glossar/pathfinding/, zuletzt aktualisiert am 24.01.2023, zuletzt geprüft am 21.02.2024.

Amazon Web Services (2024): Was ist Datenvisualisierung – Erklärung von Datenvisualisierung – AWS. Online verfügbar unter https://aws.amazon.com/de/what-is/data-visualization/, zuletzt aktualisiert am 16.02.2024, zuletzt geprüft am 18.02.2024.

Belwariar, Rachit (2016): A Search Algorithm. In: *GeeksforGeeks*, 16.06.2016. Online verfügbar unter https://www.geeksforgeeks.org/a-search-algorithm/, zuletzt geprüft am 12.01.2024.

Bröcker, Christoph A. (1999): Verteilte Visualisierung geometrischer Algorithmen und Anwendungen auf Navigationsverfahren in unbekannter Umgebung, S. 17–18. Online verfügbar unter https://freidok.uni-freiburg.de/fedora/objects/freidok:9/datastreams/FILE1/content, zuletzt geprüft am 18.02.2024.

Buzan, Tony; Buzan, Barry (2011): Das Mind-Map-Buch. Die beste Methode zur Steigerung Ihres geistigen Potenzials. 7., aktual. Aufl. Landsberg, München: mvg.

Candra, Ade; Budiman, Mohammad Andri; Pohan, Rahmat Irfan (2021): Application of A-Star Algorithm on Pathfinding Game. In: *J. Phys.: Conf. Ser.* 1898 (1), S. 12047. DOI: 10.1088/1742-6596/1898/1/012047.

Carnegie Mellon University (2017): Pathfinding. Online verfügbar unter https://www.cs.cmu.edu/~112-n22/notes/student-tp-guides/Pathfinding.pdf, zuletzt geprüft am 21.02.2024.

Cox, Graham (2020): A* Pathfinding Algorithm. In: *Baeldung on Computer Science*, 13.08.2020. Online verfügbar unter https://www.baeldung.com/cs/a-star-algorithm, zuletzt geprüft am 18.12.2023.

Cui, Xiao; Shi, Hao (2010): A*-based Pathfinding in Modern Computer Games. Online verfügbar unter https://www.researchgate.net/publication/267809499_A-based_Pathfinding_in_Modern_Computer_Games, zuletzt aktualisiert am 12.01.2024, zuletzt geprüft am 12.01.2024.

Einsiedler, Manfred; Wieser, Andreas (2019): Metrische Räume. Online verfügbar unter https://wp-prd.let.ethz.ch/WP0-CIPRF9693/chapter/metrische-raume/, zuletzt aktualisiert am 24.02.2024, zuletzt geprüft am 24.02.2024.

e-teaching.org (2016): Formen der Visualisierung. Online verfügbar unter https://www.e-teaching.org/didaktik/gestaltung/visualisierung/formen, zuletzt aktualisiert am 18.02.2024, zuletzt geprüft am 18.02.2024.

Foead, Daniel; Ghifari, Alifio; Kusuma, Marchel Budi; Hanafiah, Novita; Gunawan, Eric (2021): A Systematic Literature Review of A* Pathfinding. In: *Procedia Computer Science* 179, S. 507–514. DOI: 10.1016/j.procs.2021.01.034.

GeeksforGeeks (2012): Worst, Average and Best Case Analysis of Algorithms. In: *GeeksforGeeks*, 19.02.2012. Online verfügbar unter https://www.geeksforgeeks.org/worst-average-and-best-case-analysis-of-algorithms/, zuletzt geprüft am 19.02.2024.

GeeksforGeeks (2022): Computer Animation. In: *GeeksforGeeks*, 11.07.2022. Online verfügbar unter https://www.geeksforgeeks.org/computer-animation/, zuletzt geprüft am 18.02.2024.

Hart, Peter; Nilsson, Nils; Raphael, Bertram (1968): A Formal Basis for the Heuristic Determination of Minimum Cost Paths. In: *IEEE Trans. Syst. Sci. Cyber.* 4 (2), S. 100–107. DOI: 10.1109/TSSC.1968.300136.

Ionos (2023): Pathfinding: Wegfindung in der Informatik. In: *IONOS*, 05.05.2023. Online verfügbar unter https://www.ionos.de/digitalguide/online-marketing/web-analyse/pathfinding/, zuletzt geprüft am 21.02.2024.

Karimi, Akbar (2021): What Is a Heuristic Function? In: *Baeldung on Computer Science*, 22.05.2021. Online verfügbar unter https://www.baeldung.com/cs/heuristics, zuletzt geprüft am 21.02.2024.

Lucidchart (2024): Was ist ein Flussdiagramm? Online verfügbar unter https://www.lucidchart.com/pages/de/was-ist-ein-flussdiagramm, zuletzt aktualisiert am 15.02.2024, zuletzt geprüft am 18.02.2024.

Makarov, Andrew (2024): 12 Augmented Reality Trends of 2024: New Milestones in Immersive Technology. In: *MobiDev*, 02.01.2024. Online verfügbar unter https://mobidev.biz/blog/augmented-reality-trends-future-ar-technologies, zuletzt geprüft am 18.02.2024.

Neo4j (o. J.): Path Finding Algorithms. Online verfügbar unter https://neo4j.com/developer/graph-data-science/path-finding-graph-algorithms/, zuletzt aktualisiert am 11.05.2023, zuletzt geprüft am 21.02.2024.

Okereke, Israel (2023): Emerging trends in Augmented Reality (AR) and Virtual Reality (VR) technology, 07.08.2023. Online verfügbar unter https://www.linkedin.com/pulse/emerging-trends-augmented-reality-ar-virtual-vr-israel-okereke, zuletzt geprüft am 18.02.2024.

Patel, Amit (2023): Introduction to A*. Online verfügbar unter https://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html, zuletzt aktualisiert am 16.10.2023, zuletzt geprüft am 18.12.2023.

Patel, Amit J. (2014): Introduction to the A* Algorithm (Red Blob Games). Online verfügbar unter https://www.redblobgames.com/pathfinding/a-star/introduction.html.

Pfeiffer, Simon (2022): Pathfinding with Javascript: The A* Algorithm. In: *DEV Community*, 03.06.2022. Online verfügbar unter https://dev.to/codesphere/pathfinding-with-javascript-the-a-algorithm-3jlb, zuletzt geprüft am 18.12.2023.

Raza, Sayyed Jaffar Ali; Gupta, Nitish A.; Chitaliya, Nisarg; Sukthankar, Gita R. (2018): Real-World Modeling of a Pathfinding Robot Using Robot Operating System (ROS). Online verfügbar unter http://arxiv.org/pdf/1802.10138.pdf.

Rubio, Fatima (2023): The 5 Most Powerful Pathfinding Algorithms. Online verfügbar unter https://www.graphable.ai/blog/pathfinding-algorithms/, zuletzt aktualisiert am 22.11.2023, zuletzt geprüft am 21.02.2024.

Russell, Stuart J.; Norvig, Peter (2022): Artificial intelligence. A modern approach. Unter Mitarbeit von Ming-wei Chang, Jacob Devlin, Anca Dragan, David Forsyth, Ian Goodfellow, Jitendra Malik et al. Fourth edition, global edition. Boston: Pearson (Always learning). Online verfügbar unter https://elibrary.pearson.de/book/99.150005/9781292401171.

Schumann, Heidrun (2000): Visualisierung. Grundlagen und Allgemeine Methoden. Unter Mitarbeit von Wolfgang Müller. 1st ed. Berlin, Heidelberg: Springer Berlin / Heidelberg.

Spektrum (2000): Heuristiken. Heidelberg. Online verfügbar unter https://www.spektrum.de/lexikon/psychologie/heuristiken/6524, zuletzt aktualisiert am 04.12.2014, zuletzt geprüft am 18.02.2024.

Spektrum (2017): euklidischer Abstand. Springer Verlag GmbH. Deutschland, zuletzt aktualisiert am 14.11.2019, zuletzt geprüft am 18.02.2024.

Splunk (2024): Was ist Augmented Reality und Virtual Reality? Online verfügbar unter https://www.splunk.com/de_de/data-insider/what-are-augmented-reality-and-virtual-reality.html, zuletzt aktualisiert am 18.02.2024, zuletzt geprüft am 18.02.2024.

studyflix (2024): Pseudo-Code und Struktogramme. Online verfügbar unter https://studyflix.de/informatik/pseudo-code-und-struktogramme-830, zuletzt aktualisiert am 18.02.2024, zuletzt geprüft am 18.02.2024.

Swift, Nicholas (2017): Easy A* (star) Pathfinding. Online verfügbar unter https://medium.com/@nicholas.w.swift/easy-a-star-pathfinding-7e6689c7f7b2, zuletzt geprüft am 18.12.2023.

Tableau Software (2024): Handbuch zur Datenvisualisierung: Definition, Beispiele und Lernressourcen. Online verfügbar unter https://www.tableau.com/de-de/learn/articles/data-visualization, zuletzt aktualisiert am 18.02.2024, zuletzt geprüft am 18.02.2024.

Technologies, Unity (2024): What is Computer Animation. Online verfügbar unter https://unity.com/solutions/computer-animation-explained, zuletzt aktualisiert am 18.02.2024, zuletzt geprüft am 18.02.2024.

Thommen, Jean-Paul; Siepermann, Markus (2018): Definition: Heuristik. In: *Springer Fachmedien Wiesbaden GmbH*, 14.02.2018. Online verfügbar unter https://wirtschaftslexikon.gabler.de/definition/heuristik-34474, zuletzt geprüft am 18.02.2024.

Wilbert, Annette (2024): Methodeneinsatz - Visualisieren. Konrad-Adenauer-Stiftung e.V. Online verfügbar unter https://www.kas.de/de/web/politische-bildung/visualisieren, zuletzt aktualisiert am 08.02.2024, zuletzt geprüft am 08.02.2024.

Woltmann, Sven (2021): A*-Algorithmus (mit Java-Beispiel). In: *HappyCoders.eu*, 27.01.2021. Online verfügbar unter https://www.happycoders.eu/de/algorithmen/a-stern-algorithmus-java/, zuletzt geprüft am 18.02.2024.

Zentrum für Schulqualität und Lehrerbildung (ZSL) (o. D.): Visualisierung. Hg. v. Land Baden-Württemberg. Zentrum für Schulqualität und Lehrerbildung (ZSL). Online verfügbar unter https://lehrerfortbildung-bw.de/st_digital/medienkompetenz/digipraes/projekt/bausteine/bau/visualisierung.html, zuletzt aktualisiert am 22.12.2023, zuletzt geprüft am 22.12.2023.