**Gemeinsam IT entdecken**

**IT2School**



**Modul KI-A3 – Große Gesten**

Mit KI Geschichten erzählen

Inhalt

[1 Große Gesten – Mit KI Geschichten erzählen 3](#_Toc99966708)

[2 Warum gibt es das Modul? 4](#_Toc99966709)

[3 Ziele des Moduls 4](#_Toc99966710)

[4 Die Rolle der Unternehmensvertreterin/des Unternehmensvertreters 4](#_Toc99966711)

[5 Inhalte des Moduls 5](#_Toc99966712)

[5.1 Gestenerkennung 5](#_Toc99966713)

[5.2 Blockbasiertes Programmieren in der Entwicklungsumgebung Snap! 7](#_Toc99966714)

[6 Unterrichtliche Umsetzung 8](#_Toc99966715)

[6.1 Grober Unterrichtsplan 9](#_Toc99966716)

[6.2 Stundenverlaufsskizzen 10](#_Toc99966717)

[6.2.1 Doppelstunde 1 10](#_Toc99966718)

[6.2.2 Doppelstunde 2 12](#_Toc99966719)

[6.2.3 Doppelstunde 3 13](#_Toc99966720)

[6.3 Hinweise zur unterrichtlichen Umsetzung 15](#_Toc99966721)

[7 Einbettung in verschiedene Fächer und Themen 15](#_Toc99966722)

[8 Anschlussthemen 15](#_Toc99966723)

[9 Literatur und Links 15](#_Toc99966724)

[10 Arbeitsmaterialien 16](#_Toc99966725)

[11 Glossar 17](#_Toc99966726)

[12 Fragen, Feedback, Anregungen 18](#_Toc99966727)

[13 Bildverzeichnis 18](#_Toc99966728)

# Pinch, Screen, Display, Zoom, TouchGroße Gesten – Mit KI Geschichten erzählen

Gestenerkennung und verwandte Algorithmen verwenden Schülerinnen und Schüler in ihrem Alltag regelmäßig beispielsweise zur Steuerung von Touchscreens. Gesten wie das sogenannte „Swipen“ (nach links oder rechts wischen) oder der „Pinch-Zoom“ (Daumen und Zeigefinger einer Hand zusammen-/auseinanderbewegen, um auf dem Bildschirm angezeigte Inhalte zu vergrößern oder zu verkleinern) werden häufig und unbewusst angewandt und können geräteübergreifend eingesetzt werden. Nach einem spielerischen Einstieg über das Spiel Pictionary (auch bekannt als Montagsmaler) programmieren Schülerinnen und Schüler in diesem Modul eine Gestenerkennung selbst und lernen dabei, künstliche Intelligenz kreativ einzusetzen, um eigene Geschichten zu animieren.

|  |  |  |
| --- | --- | --- |
| Lernfeld/Cluster: | Künstliche Intelligenz | |
| Zielgruppe/Klassenstufe: |  | 4. bis 5. Klasse |
|  | 6. bis 7. Klasse |
| **X** | 8. bis 10. Klasse |
| **X** | 11. bis 12. Klasse |
| Geschätzter Zeitaufwand: | 3 Doppelstunden (zusätzliche Stunden zur Vertiefung werden empfohlen) | |
| Lernziele: | * Einsatz von Gestenerkennung im Alltag erkennen und beschreiben. * Den gesamten Prozess einer Künstlichen Intelligenz (Maschinelles Lernen) anhand eines Beispiels zur Gestenerkennung nachvollziehen und aktiv gestalten:   Programmierung – Training – Anwendung   * Kreative Möglichkeiten entwickeln, die erstellte KI zum Erzählen einer Geschichte zu verwenden. * Schwächen des erstellten Programms analysieren und Lösungsvorschläge erarbeiten. | |
| Vorkenntnisse der Schülerinnen und Schüler: | Erforderlich:   * Grundkenntnisse im Programmieren mit Snap! oder Scratch | |
| Vorkenntnisse der/des Lehrenden: | Erforderlich:   * Kenntnisse im Programmieren mit Snap! oder Scratch | |
| Vorkenntnisse der Unternehmensvertreterin/des Unternehmensvertreters: | Empfohlen:   * Kenntnisse im Programmieren mit Snap! oder Scratch | |
| Sonstige Voraussetzungen: | * Pro 1-2 SuS sollte ein Computer oder Tablet zur Verfügung stehen * Ein Beamer zur Präsentation der Ergebnisse ist von Vorteil | |

# Warum gibt es das Modul?

Gestenerkennung gehört zu den ältesten Anwendungsbereichen künstlicher Intelligenz, die bereits seit den 1960er Jahren als Schnittstelle zwischen Mensch und Computer (Human Computer Interaction – HCI, siehe Glossar) dient. Sie bietet interaktive und kreative Steuerungsmöglichkeiten für elektronische Geräte, die über die Vielseitigkeit üblicher Peripheriegeräte hinausgehen. Den Schülerinnen und Schülern sind vielfältige Formen der Anwendung bereits bekannt, etwa durch das Entsperren des Smartphones, dem Swipen oder dem Spielen von Videospielen mit Bewegungssteuerung (Kinect/ Wii). In diesem Modul haben die Schülerinnen und Schüler die Möglichkeit, sich mit diesem Alltagsphänomen auseinanderzusetzen. Sie können ihre eigene Gesten-/ Zeichenerkennung programmieren, um so Phänomene im Zusammenhang mit Gestenerkennung im Alltag und deren Vorteile und Schwächen besser deuten und hinterfragen zu können.

# Ziele des Moduls

Die Schülerinnen und Schüler…

* befassen sich mit Anwendungen künstlicher Intelligenz anhand eines alltäglichen Beispiels
* lernen die algorithmischen Grundbausteine zur Programmierung einer Gestenerkennung in Snap! kennen und wenden diese an
* implementieren eine Gestenerkennung basierend auf einem sogenannten Nearest Neighbor (nächster Nachbar) – Algorithmus
* setzten sich kreativ und handlungsorientiert mit Gestenerkennung auseinander
* reflektieren die Grenzen und Herausforderungen der Gestenerkennung in Bezug auf Bias und entwickeln entsprechende Lösungsansätze

# Die Rolle der Unternehmensvertreterin/des Unternehmensvertreters

* Unterstützung der Lehrkraft bei der Durchführung des Unterrichts
* Erfahrungsberichte über den Einsatz von Gestenerkennung (oder ähnlichem) im Unternehmensumfeld, falls vorhanden.

# Inhalte des Moduls

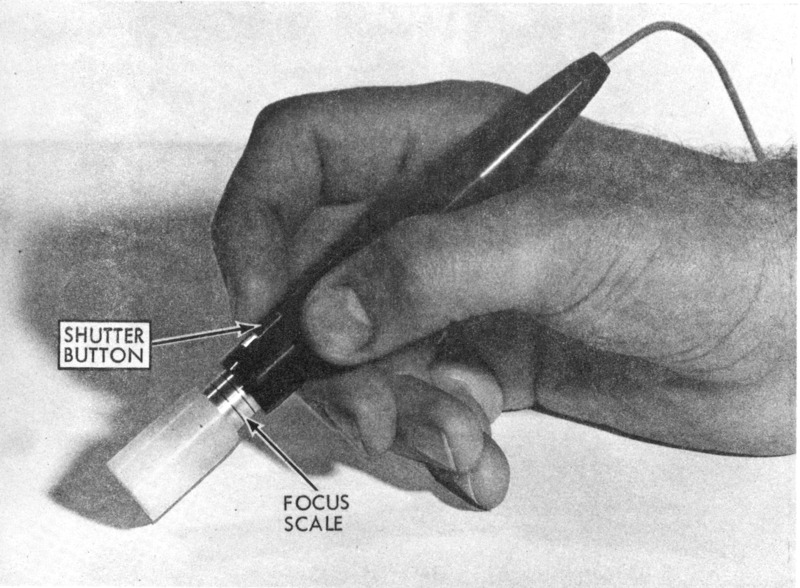
## Gestenerkennung

Die Gestenerkennung ist ein Teilbereich der Informatik, der sich mit der Interpretation menschlicher Bewegungen mithilfe von Algorithmen befasst. Als Geste werden physische Bewegungen bezeichnet, die von einem Computer wahrgenommen werden können, wie etwa Hand- oder Kopfgesten. Ziel ist es, mithilfe der Gesten den Computer zu steuern und in Anwenungsprogrammen zu agieren.

Gestenerkennungen sind vergleichsweise frühe Formen der Mensch-Computer-Interaktion (HCI). Zahlreiche Grundlagen der HCI, unter anderem zur Gestenerkennung, wurden mit Ivan Sutherlands Sketchpad bereits in den 1960er Jahren gelegt (Sutherland, 1963). Dieses erste Programm mit einer grafischen Oberfläche konnte mithilfe von Bewegungen mit einem sogenannten Lichtgriffel gesteuert werden. Es wurde 1963 im Rahmen der Doktorarbeit von Ivan Sutherland entwickelt und damit ca. zeitgleich zum Bau des ersten Maus-Prototypen. (In Kapitel 9 *Literatur und Links* befindet sich ein Video mit einer Demonstration von Sketchpad.)

A person working on a computer

Description automatically generated with low confidence



Lichtgriffel (CC BY SA 3.0, Kerry Rodden)

Ivan Sutherland demonstriert Sketchpad (CC BY SA 3.0, Kerry Rodden)

Bereits seit 1964 gibt es Programme, die in der Lage sind, Gesten automatisiert zu erkennen. So konnte beispielsweise das RAND Tablet mittels GRAIL (Graphical Input Language) gesteuert werden und 53 verschiedene handgeschriebene Ziffern, Symbole und Buchstaben erkennen und unterscheiden (Link zum Video in Kapitel 9). Ebenfalls 1964 wurde das erste trainierbare Programm zur Gestenerkennung entwickelt (Myers, 1998).

Heutzutage müssen Gesten nicht mit einem Stift auf einen Bildschirm gezeichnet werden. Sie können im zwei- oder dreidimensionalen Raum (z.B. Steuerung einer Kinect oder Wii) stattfinden und mit oder ohne Hilfsmittel ausgeführt werden. Dafür wird meist eine sogenannte TOF (Time of Flight)-Kamera verwendet. Diese sendet einen Lichtimpuls aus und misst für jeden Bildpunkt die Zeit, die ein Lichtstrahl zum Objekt und von dort wieder zurück zur Kamera benötigt. Dadurch kann für jeden Bildpunkt eine Entfernung zum nächsten Objekt berechnet werden (nähere Beschreibung von TOF-Kamers siehe Kapitel 9).

Typische Gesten, die heute zur Steuerung von Computern, Tablets oder Mobiltelefonen verwendet werden, sind beispielsweise Pinch-to-Zoom, das Zusammen- oder Auseinanderbewegen von zwei Fingern zum Zoomen, oder Swipen, um nach vorne oder zurückzublättern.

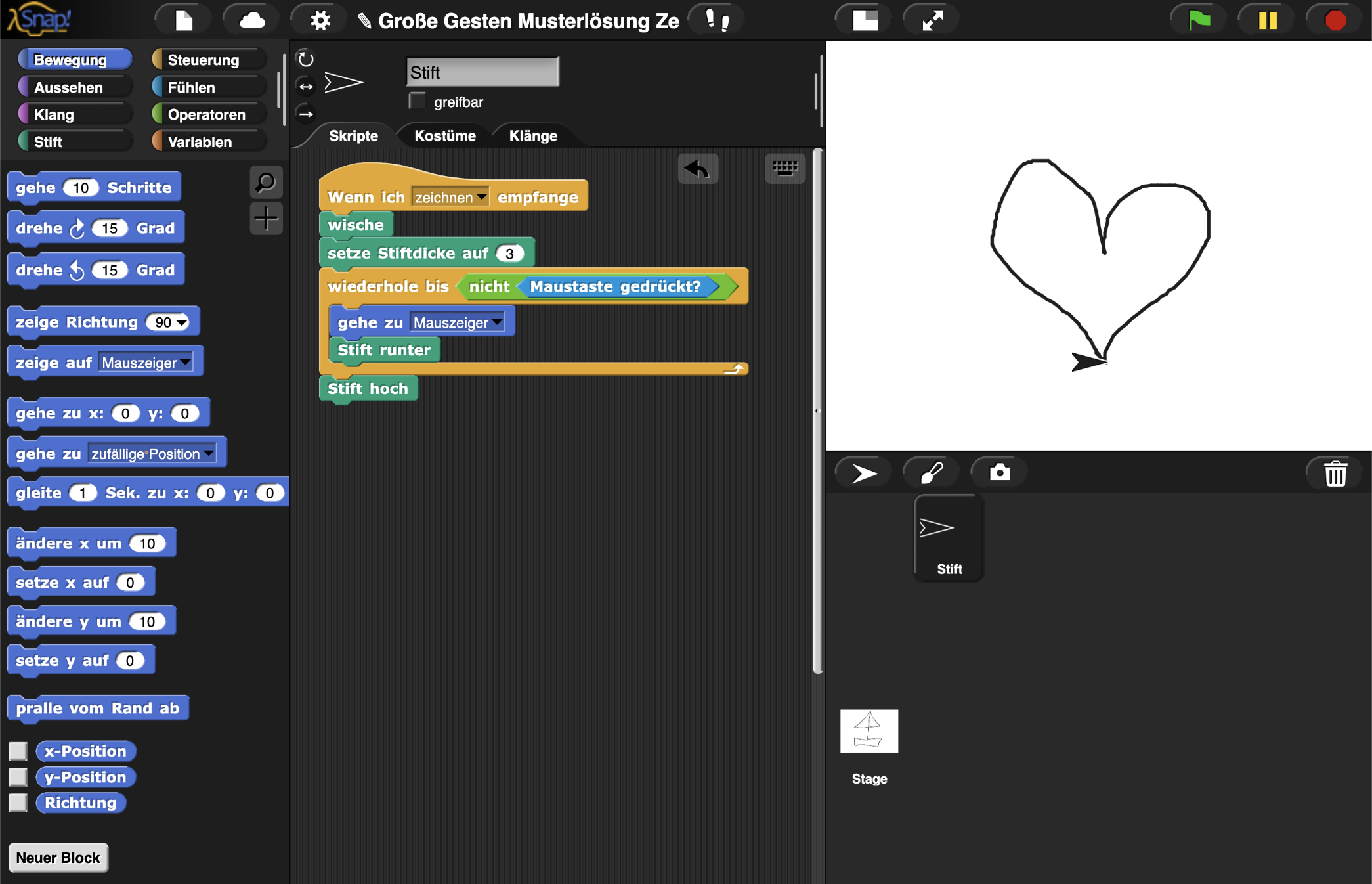
Als Gestenerkennungsalgorithmus in diesem Modul verwenden wir eine vereinfachte Form des $1 Gesture Recognizers, der 2007 von Forschenden an der University of Washington und bei Microsoft veröffentlicht wurde (Wobbrock et al., 2007). Er dient der Erkennung von 2D Einstrichgesten (Unistroke Gestures), d.h. Gesten, die ohne Absetzen am Stück gezeichnet werden. Für dieses Modul ist er besonders geeignet, da er niedrigschwellig das eigene Prototyping im HCI-Design ermöglicht. Das gesamte Programm umfasst nur 100 Zeilen Code. Der $1-Gesture Recognizer verwendet einen Nächste-Nachbar-Algorithmus (Nearest Neighbour). Diese Art von Programmen wird auch im Modul „Von Daten und Bäumen“ verwendet. In diesem Fall wird dabei der Pfad einer Geste mit einer Liste von Beispielpfaden verglichen. Der Pfad, der am ehesten mit der Zeichnung übereinstimmt, der nächste Nachbar, wird als korrekt erkannte Geste angenommen. Bei dieser simplen Form des Maschinellen Lernens handelt es sich um überwachtes Lernen (supervised learning), d.h. das Programm lernt anhand beschrifteter Eingaben, die als Beispiele übergeben werden.

Der eingesetzte Erkennungsalgorithmus (Nächster Nachbar) ist relativ leicht verständlich und kann von den Schülerinnen und Schülern selbstständig erarbeitet und programmiert werden.

Das Thema Gestenerkennung wurde für dieses Modul aus mehreren Gründen gewählt. Bereits erwähnt wurde die Tatsache, dass die Schülerinnen und Schüler Gestensteuerungen aus ihrem Alltag kennen. Außerdem handelt es sich dabei um eine Form der Künstliche Intelligenz, die verhältnismäßig wenige Beispiele zum Trainieren des Programms benötigt, um ein Ergebnis erzielen zu können. So könnte eine einfache Form des Programmes schon anhand zweier Beispiele zwischen zwei Gesten unterscheiden. Das Programm kann während der Erstellung immer wieder leicht getestet werden. Der eingesetzte Erkennungsalgorithmus (Nächster Nachbar) ist relativ leicht verständlich und kann von den Schülerinnen und Schülern selbstständig erarbeitet und programmiert werden. Aus einer Liste von Pfaden, die Punkt für Punkt verglichen werden, wird jeweils der ausgewählt, bei dem die insgesamte Differenz zum Pfad der aktuellen Geste am geringsten ist, also der „nächste Nachbar“. Mit den erkannten Gesten lassen sich hervorragend eigene Programme steuern. So bietet sich die Möglichkeit, künstliche Intelligenz kreativ einzusetzen, beispielsweise zum Storytelling. Wie bereits erwähnt eignet sich die Gestenerkennung auch dazu, problematische Bereiche der Künstlichen Intelligenz, wie Data Bias, erfahrbar zu machen.

## Blockbasiertes Programmieren in der Entwicklungsumgebung Snap!

Für die Umsetzung des Moduls verwenden wir die browserbasierte Programmiersprache Snap! (<https://snap.berkeley.edu>). Snap! ist eine visuelle Programmiersprache, bei der Programmeinheiten, sogenannte Blöcke, durch Drag and Drop zu größeren Programmen zusammengesteckt werden können. Die Ergebnisse dieser Programme werden auf der sogenannten Bühne im rechten Bereich des Snap!-Fensters angezeigt (Einführung in Snap! – siehe Kapitel 9).



Palette mit Blöcken

Bühne, auf der der sichtbare Teil der Programme abläuft

Skriptbereich, in dem programmiert wird

Die Snap! Entwicklungsumgebung

In diesem Modul werden besonders die über Scratch hinausgehenden Funktionen von Snap! verwendet, die es erlauben Blöcke zu erstellen, die Werte zurückgeben (Funktionen). Ebenfalls setzen wir benutzerdefinierte Blöcke ein, an die andere Blöcke als Parameter (Eingabe) übergeben werden können. Weitere Informationen darüber, wie benutzerdefinierte Blöcke in Snap! erstellt werden können, finden sich im Material für Lehrkräfte und Unternehmensvertreter\*innen (A3-2.1-Blöcke in Snap!).

Beim ersten Öffnen ist Snap! standardmäßig auf englische Sprache eingestellt. Bei Wunsch kann die Sprache über das Einstellungsmenü (Zahnrad – Language) auf Deutsch geändert werden.

# Unterrichtliche Umsetzung

Die Materialien im Zusammenhang mit diesem Modul gliedern sich in drei Bereiche, die im Rahmen von 3 Doppelstunden durchlaufen werden können.

**Einführung**

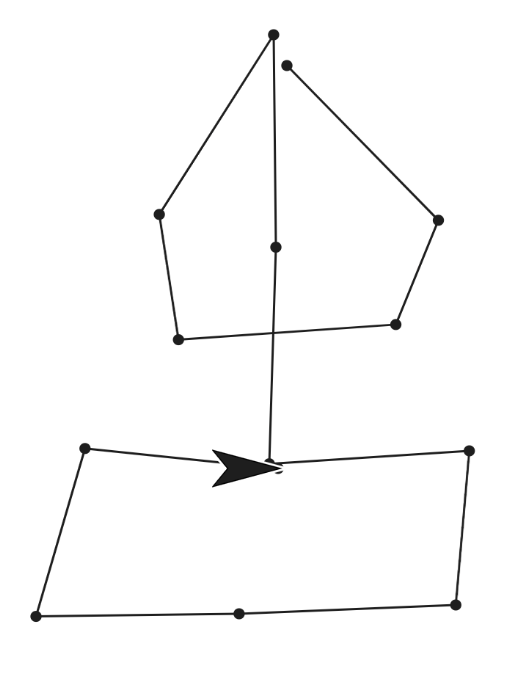
In der ersten Doppelstunde starten die Schülerinnen und Schüler mit einer Runde Pictionary (Montagsmaler), um sich auf das Thema der Stunde – Gestenerkennung und Erstellung eines Zeichenprogramms einzustimmen. Alternativ kann zum Einstieg der ersten Stunde auch einfach eine animierte Beispielgeschichte präsentiert werden (siehe: <https://tinyurl.com/GroßeGestenSchneewittchen>) Danach erarbeiten die Schülerinnen und Schüler, was Gestenerkennung ist und wie sie im Alltag eingesetzt wird. Darüber hinaus diskutieren sie, wo dabei Vor- und Nachteile liegen können. Nach dieser Einführung erstellen sie in Snap! eine erste Programmierung mit der sie Einstrichgesten aufzeichnen können. Hierfür werden Programmierblöcke verwendet, die das Zeichnen auf der Bühne ermöglichen.



Zeichenprogramm mit Ergebnis

**Vertiefung**

In der zweiten Doppelstunde wird die Gestenerkennung programmiert. Dafür müssen die Gesten aus dem Zeichenprogramm vergleichbar gemacht werden. Dies geschieht über Pfade, d.h. Listen von x- und y-Koordinaten, die während des Zeichnens gespeichert werden. Die Schülerinnen und Schüler erweitern ihr Zeichenprogramm, um damit Pfade aufzuzeichnen und wieder abzuspielen. Im Anschluss werden diese Pfade miteinander verglichen. Dafür ist es nötig, dass die Zeichnungen die gleiche Anzahl an Punkten aufweisen. Mithilfe eines „resample“-Blocks kann ausgewählt werden, auf wie viele Punkte sich ein gezeichneter Pfad verteilen soll.



Zeichnung eines Bootes normiert auf 14 Punkte mit dem "resample" Block

Nun kann die Differenz zwischen zwei Pfaden berechnet werden. Die Schülerinnen und Schüler trainineren nun ihr Programm mit mehreren Beispielgesten.

Im Anschluss daran haben die Schülerinnen und Schüler die Möglichkeit im Rahmen eines Abschlussprojekts Gesten zu verwenden und damit eine kurze Geschichte zu illustrieren. Diese kann entweder selbstverfasst oder Bestandteil einer größeren Geschichte sein, die von der Lehrkraft vorgegeben wird. So können beispielsweise Kinderbücher oder Märchen als Grundlage für eine Geschichte dienen, die von der Klasse gemeinsam erarbeitet wird.

**Präsentation und Abschluss**

In der dritten Doppelstunde präsentieren die Schülerinnen und Schüler ihre Ergebnisse vor der Klasse. Dies kann entweder live mit einem Snap!-Projekt oder über ein vorher aufgezeichnetes Video erfolgen. Zum Abschluss des Moduls testen Schülerinnen und Schüler die Schwächen ihres Programms. Es gibt Einschränkungen in der Funktionalität dadurch, dass absolute Koordinaten zum Vergleich verwendet werden. So können beispielsweise anders positionierte oder stark in der Größe variierende Gesten nicht erkannt werden. Gleiches gilt, wenn die Geste in eine andere Richtung oder auf dem Kopf ausgeführt wird. Gemeinsam erarbeiten die Schülerinnen und Schüler Lösungsansätze, mit denen das Programm verbessert werden könnte, wie beispielsweise weitere Normierungsfunktionen (translatieren, rotieren, skalieren) und die Verwendung diverserer Beispiele.

## Grober Unterrichtsplan

|  |  |  |  |
| --- | --- | --- | --- |
| Inhalte | Kurze Zusammenfassung | Dauer | Material |
| Einführung in Gestenerkennung, Erstellung eines Zeichenprogramms, Animation der Zeichnungen | Die Schülerinnen und Schüler beschäftigen sich über eine Beispielgeschichte, ein Spiel und/oder eine Diskussion mit Gestenerkennung. Im Anschluss programmieren sie ein Zeichenprogramm für Einstrichgesten in Snap! Diese Gesten können in einer Erweiterung des erstellten Programms animiert werden. | 90 min | KI-A3.1.1 –  KI-A3.1.3 |
| Speicherung von Pfaden, Implementierung des Nächster-Nachbar Algorithmus zur Erkennung der Gesten, Animation der erkannten Gesten | Um Gesten erkennen zu können, müssen diese vergleichbar gemacht werden, beispielsweise über Pfade. SuS erweitern ihr Programm, um Pfade zu speichern und zu vergleichen/erkennen. Als Hausaufgabe oder in einer weiteren Unterrichtsstunde konzipieren die SuS eine Geschichte, die sie mithilfe der Gesten steuern und illustrieren. | 90 min | KI-A3.1.4 –  KI-A3.1.6 |
| Präsentation der Geschichten, Schwächen der Gestenerkennung und Verbesserungsmöglichkeiten. Diskussion über weitere Einsatzmöglichkeiten. | Die SuS präsentieren ihre Geschichten in der Klasse.  Die Gestenerkennung wird reflektiert – Schwächen und ihre Lösungsansätze, Vor- und Nachteile sowie Einsatzmöglichkeiten werden diskutiert. | 90 min | KI-A3.1.7 |

## Stundenverlaufsskizzen

**Abkürzungen/Legende**

AB = Arbeitsblatt/Arbeitsblätter; L = Lehrkraft; MuM = Mitschülerinnen und Mitschüler; SuS = Schülerinnen und Schüler;   
UV = Unternehmensvertreterin/Unternehmensvertreter

### Doppelstunde 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Zeit | Phase | Sozialform/  Lehrerimpuls | Inhalt/Unterrichtsgeschehen | Material |
|  | Vorbereitung |  | Ausdrucken bzw. Vorbereiten des Versands der Arbeitsmaterialien  Falls Einstieg über Pictionary/Montagsmaler geplant ist, müssen Beispielbegriffe auf Zettelchen vorbereitet werden.  Falls Einstieg über Geschichte geplant ist, muss diese vorbereitet werden. | Arbeitsmaterial KI-A3.1.1-1.7 |
| 5-20 min | Einstieg | Plenum | Variante 1:  Stundeneinstieg über Pictionary/ Montagsmaler: Die Klasse bildet zwei Gruppen und lost aus, welche Gruppe beginnt. Die beginnende Gruppe bestimmt ein Mitglied, das als erstes malt. Eine Zeitdauer (z.B. 2 Minuten) wird festgelegt. Die malende Person zieht dann verdeckt einen der Zettel und versucht den darauf stehenden Begriff zu malen. Dabei darf sie nicht sprechen, "Ja" und "Nein" sind aber erlaubt. Die anderen Gruppenmitglieder versuchen anhand der Zeichnung den Begriff zu erraten. Ist der Begriff korrekt, zieht die malende Person sofort den nächsten Zettel und malt weiter. Das wiederholt sich so lange, bis die vorgegebene Zeit abgelaufen ist. Jeder erratene Begriff bringt der Gruppe einen Punkt. Ist die Zeit abgelaufen, darf die andere Gruppe malen und raten. Danach wird wieder gewechselt. Das Spiel endet, wenn alle mit malen dran waren, wenn eine bestimmte Punktzahl erreicht ist oder wenn es keine Zettel mehr gibt. Wer am Ende die meisten Punkte hat gewinnt.  Variante 2:  Stundeneinstieg über eine Beispielgeschichte: Die Lehrkraft bereitet eine Geschichte vor und trägt diese mit dem fertigen Gestenerkennungsprogramm der Klasse vor. Alternativ kann auch die hier zur Verfügung gestellte Beispiel-Geschichte verwendet werden. |  |
| 10 min | Hinführung | Plenum | Das Thema der Unterrichtseinheit, Gestenerkennung, wird vorgestellt. Es wird kurz erläutert, was Gestenerkennung bedeutet. Dabei kann ein kurzer Rückblick auf Pictionary/Montagsmaler erfolgen (es gibt entscheidende und vernachlässigbare Details, gut zu malen ist keine Voraussetzung zum Erkennen)  Im Anschluss überlegen die SuS wo ihnen Gestenerkennung im Alltag begegnet und tragen dies ins Arbeitsmaterial KI-A3-1.1 ein und teilen ihre Ergebnisse in der Klasse.  Danach diskutieren die SuS kurz darüber, was die Vor- und Nachteile von Gestenerkennung (z.B. gegenüber Schrift oder Ton) sein könnten, und wo sie Anwendungsgebiete sehen. | Arbeitsmaterial KI-A3.1.1 |
| 25 min | Erarbeitung (Zeichnen) | Einzel- oder Paararbeit | Programmierung eines Zeichenprogramms anhand des Arbeitsmaterials KI-A3-1.2 | Arbeitsmaterial KI-A3.1.2 |
| 5-10 min | Sicherung (Zeichnen) | Plenum | Gesamtlösung für das Zeichenprogramm im Plenum besprechen und sicherstellen, dass alle SuS für den weiteren Verlauf ein Snap! Projekt mit diesem Stand haben. | Musterlösung KI-A3 |
| 5 min | Hinführung (Animieren) | Plenum | Eine mögliche Animation einer Geste wird von der Lehrkraft vorgestellt. Hierfür eignet sich beispielsweise das schlagende Herz. Das passende Animationsskript kann von der Lehrkraft selbst programmiert werden. Alternativ kann als Grundlage für die Vorstellung dieses Programm verwendet werden: <https://tinyurl.com/GestenAnimieren>  Falls gewünscht, kann die Funktion des „animiere“-Blocks basierend auf Material KI-A3-2.1 genauer ausgeführt werden. | Arbeitsmaterial KI-A3.2.1 |
| 25 min | Erarbeitung (Animieren) | Einzel- oder Paararbeit | Animation einiger Einstrichgesten anhand des Arbeitsmaterials KI-A3-1.3. Die SuS überlegen einige Beispiele für Einstrichgesten mit passenden Animationen und setzen diese in Snap! um. | Arbeitsmaterial KI-A3.1.3 |
| 5-10 min | Sicherung (Animieren) | Plenum | Einige mögliche Lösungen für die Animation von Gesten werden besprochen. Wenn die Zeit ausreicht, können einzelne SuS ihre Programme zeigen, ansonsten können einige Animationen kurz beschrieben werden. | Musterlösung KI-A3 |

### Doppelstunde 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Zeit | Phase | Sozialform/  Lehrerimpuls | Inhalt/Unterrichtsgeschehen | Material |
| 5-10 min | Hinführung | Plenum | Zusammenfassung, was bisher erarbeitet wurde. Erläuterung der Ziele dieser Doppelstunde   * Gesten miteinander vergleichbar machen * Ähnlichste Geste ermitteln und ausgeben |  |
| 5 min | Erarbeitung | Plenum | Einführung zu Pfaden:  Gesten können als Bilder nicht direkt verglichen werden, deshalb müssen Zusatzinformationen mitgespeichert werden.  Wo werden Pfade noch verwendet (z.B. Vektorgrafiken) und wie können sie in Snap! gespeichert werden. |  |
| 15 min | Erarbeitung (Pfade erzeugen) | Einzel- oder Paararbeit | Speicherung von Pfaden (Koordinaten) anhand des Arbeitsmaterials KI-A3-1.4 Erweiterung des Snap! Projekts um eine Variable, in der beim Zeichnen der jeweilige Pfad gespeichert wird. | Arbeitsmaterial KI-A3.1.4 |
| 45 min | Erarbeitung (Pfade vergleichen) | Einzel- oder Paararbeit | Vergleichen von Pfaden anhand des Arbeitsmaterials KI-A3-1.5  Anzahl an Punkten verändern mit „resample“ (ca. 15 min): Die SuS testen die Funktion des „resample“ Blocks, um Zeichen auf bestimmte Anzahl von Punkten zu normieren.  Unterschied zwischen Pfaden messen (5-10 min): Die SuS testen mit dem „Unterschied“ Block, wie verschieden zwei Pfade sind  Beispiele Speichern (10 min): Die SuS trainineren ihr Programm mit mehreren Beispielgesten mit passender Beschriftung, die in einer Variable gespeichert werden.  Ähnlichste Geste ermitteln (15-20 min): Die SuS erstellen mithilfe eines Parson’s Puzzles einen „erkenne“-Block, der die eigentliche Gestenerkennung ausführt. Dieser gibt den Namen des erkannten Pfads (dem nächsten Nachbarn) aus der Liste der Beispiele aus.  Es empfiehlt sich, nach jeder der Arbeitsphasen eine kurze Sicherungsphase einzuplanen, um sicherzustellen, dass alle SuS mit demselben Stand ihrer Snap!-Projekte weitermachen. An dieser Stelle sollte auch darauf hingewiesen werden, dass es sich hierbei um einen Algorithmus des überwachten maschinellen Lernens handelt. | Arbeitsmaterial KI-A3.1.5 |
| - | Erarbeitung (Storytelling) | Einzel- oder Paararbeit | Erstellen einer animierten Geschichte anhand des Arbeitsmaterials KI-A3-1.6  Je nach verfügbarer Zeit ist dieser Teil des Moduls als Hausaufgabe, Extraeinheit oder den Rest der Stunde denkbar.  SuS können eigene Kurzgeschichten entwickeln und animieren. Alternativ kann die L. mehrere Kapitel/Abschnitte eines Buches oder einer längeren Geschichte zur Verfügung stellen, die die SuS animieren sollen, um eine zusammenhängende längere Geschichte zu erzeugen.  Optional kann die Anzahl der verwendbaren Gesten oder die Dauer der Abschlusspräsentation vorgegeben werden. | Arbeitsmaterial KI-A3.1.6 |

### Doppelstunde 3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Zeit | Phase | Sozialform/  Lehrerimpuls | Inhalt/Unterrichtsgeschehen | Material |
| 5 min | Hinführung | Plenum | Ziele der Stunde (Präsentation und Reflexion) und Ablauf der Präsentationsphase erläutern |  |
| 10 min | Hinführung | Einzel- oder Paararbeit | Optionale Übung der Präsentationen |  |
| 30 min | Präsentation | Plenum | Alle SuS präsentieren ihr Projekt in der Klasse.  Dies kann live mithilfe des Snap!-Projekts erfolgen. Das Projekt kann wie in Lehrkräftematerial KI-A3-2.2 erläutert über URL oder als Datei geteilt werden. Alternativ können die SuS vorher ein Video (Screencast mit Ton) erstellen, das während der Präsentationsphase abgespielt wird.  Das Projekt kann wie in Lehrkräftematerial KI-A3-2.2 erläutert über URL oder als Datei geteilt werden. |  |
| 15 min | Erarbeitung (Reflexion) | Paararbeit | Reflexion der Gestenerkennung anhand des Arbeitsmaterials KI-A3-1.7  Welche Fälle kann dein Programm nicht erkennen? In Zweiergruppen die im Arbeitsmaterial beschriebenen Fälle testen.  Rückblick auf die Einstiegsdiskussion zu Anwendungsmöglichkeiten, Vor- und Nachteile von Gestenerkennung. | Arbeitsmaterial KI-A3.1.7 |
| 10 min | Sicherung  (Reflexion) | Plenum | Die Ergebnisse werden gesammelt und beispielsweise in Form eines Tafelaufschriebs festgehalten. Besonders gut eignet sich dafür eine Tabelle, in deren zweiter Spalte die Lösungsansätze der nächsten Aufgabe notiert werden können. |  |
| 15 min | Erarbeitung  (Reflexion 2) | Plenum oder Gruppenarbeit | Reflexion der Gestenerkennung anhand des Arbeitsmaterials KI-A3-1.7 Vorurteile (Bias): Wie benachteiligt dein Programm Menschen und was kann man dagegen tun?  In der Gruppe oder im Plenum wird gesammelt, wie sich die erarbeiteten Probleme auf bestimmte Personengruppen auswirken können.  Es werden Lösungsansätze erarbeitet, um dies zu mitigieren:   * Mehr und diversere Beispiele * Verwendung von Gesten, die sich deutlicher unterscheiden * Einführung weitere Funktionen wie translatieren, rotieren und skalieren | Musterlösung KI-A3 |
| 5 min | Sicherung | Plenum | Die erarbeiteten Lösungsansätze werden in der zweiten Spalte der Tabelle auf der Tafel festgehalten. |  |

## Hinweise zur unterrichtlichen Umsetzung

Ein Beispielprojekt das zum Einstieg in den Unterricht verwendet werden kann, findet sich hier:   
<https://tinyurl.com/GrosseGestenBeispiel>.

Ein leeres Starterprojekt, das bereits alle benötigten Blöcke zur Durchführung des Moduls enthält, befindet sich hier: <https://tinyurl.com/GrosseGesten>.

Ein Lösungsprojekt mit zwei Beispielen ist unter folgendem Link abgelegt: <https://tinyurl.com/GrosseGestenLoesung>.

# Einbettung in verschiedene Fächer und Themen

Dieses Modul eignet sich neben dem Einsatz im Informatikunterricht auch für weitere Fächer. Insbesondere die weiterführenden Aktivitäten, die dafür sorgen sollen, dass die Gestenerkennung „fairer“ wird, sind im Mathematikunterricht einsetzbar. Dabei werden Blöcke erstellt, mit denen die Pfade skaliert, rotiert und translatiert werden können.

Aufgrund der Abschlussaufgabe eignet sich das Projekt natürlich auch für den Einsatz im Sprachunterricht, wo SuS ihre eigenen Kurzgeschichten illustrieren können. Alternativ kann auch eine Lektüre in mehrere Teile geteilt werden. Diese müssen dann von den SuS zusammengefasst und animiert werden. Im Anschluss kann daraus eine größere zusammenhängende Geschichte animiert werden.

# Anschlussthemen

Zur Vertiefung des Konzepts des überwachten Lernens beziehungsweise den Vergleich zu weiteren Formen des maschinellen Lernens eignen sich die KI-Module B3 – Schlag den Roboter und A2 – Die Bananenjagd.

Eine mögliche Reihung, die man auch für eine Projektwoche nutzen kann, ist:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Ein Bild, das Text enthält.  Automatisch generierte Beschreibung | + | Ein Bild, das Text enthält.  Automatisch generierte Beschreibung | + | Ein Bild, das Text, Automat enthält.  Automatisch generierte Beschreibung | + |  |

# Literatur und Links

* Sutherland, Ivan Edward (1963): Sketchpad: A man-machine graphical communication system. Massachusetts Institute of Technology, PhD
* Video zu Sketchpad von Ivan Sutherland mit Kommentaren von Alan Kay (Englisch):  
  <https://www.youtube.com/watch?v=495nCzxM9PI>
* Myers, Brad A. (1998): A Brief History of Human Computer Interaction Technology. ACM interactions. Vol. 5, no. 2. pp. 44-54.
* Video zu GRAIL (Englisch):  
  <https://www.youtube.com/watch?v=2Cq8S3jzJiQ>
* So funktioniert die Kinect Tiefenkamera: <https://docs.microsoft.com/de-de/azure/kinect-dk/depth-camera>
* Wobbrock, Jacob, Wilson, Andrew, Li, Yang (2007): Gestures without libraries, toolkits or training: A $1 recognizer for user interface prototypes. UIST: Proceedings of the Annual ACM Symposium on User Interface Software and Technology. 159-168.  
  Link zum Original-Paper: <http://faculty.washington.edu/wobbrock/pubs/uist-07.01.pdf>
* Einführung in Snap!: <https://open.sap.com/courses/snap1-1-de>
* Video zum Bau des Gestenerkenners in Snap! (Englisch): <https://www.youtube.com/watch?v=vhnWey4lFIo>

# Arbeitsmaterialien

|  |  |  |
| --- | --- | --- |
| Nr. | Titel | Beschreibung |
| ☻ KI-A3.1.1 | Gestenerkennung im Alltag | Arbeitsblatt mit Fragestellungen zur Verwendung von Gestenerkennung sowie deren Vor- und Nachteile. |
| ☻ KI-A3.1.2 | Ein Zeichenprogramm erstellen | Arbeitsblatt mit Programmierhinweisen zur Erstellung eines Zeichenprogramms für Einstrichgesten. |
| ☻ KI-A3.1.3 | Zeichnungen animieren | Arbeitsblatt mit Programmierhinweisen zur Animation von Malspuren. |
| ☻ KI-A3.1.4 | Pfade aufzeichen | Arbeitsblatt mit Programmierhinweisen zum Speichern von Pfaden, d.h. Listen aus x- und y- Koordinaten. |
| ☻ KI-A3.1.5 | Pfade miteinander vergleichen | Arbeitsblatt mit Erklärungen zum „resample“ und „Unterschied zwischen“ Block. Programmierhinweise zur Erstellung des „erkenne“ Blocks, d.h. der Implementierung des Nächsten Nachbar Algorithmus. |
| ☻ KI-A3.1.6 | Eine Geschichte erzählen | Arbeitsblatt mit Erklärungen zum Nachrichten versenden in Snap!. Arbeitsauftrag für das Storytelling-Projekt. |
| ☻ KI-A3.1.7 | Gestenerkennung Reflexion | Arbeitsblatt mit Anweisungen zum Testen des Programms auf Biases. |
| ☻ KI-A3 Zusatzmaterial | Blöcke in Snap! | Blockglossar für Blöcke, die nicht in Scratch vorhanden sind. Benutzerdefinierte Blöcke in Snap! erstellen. Erklärung benutzerdefinierter Blöcke, die im Modul verwendet werden. |
| ☻ KI-A3 Blockbibliothek | Blockbibliothek für dieses Modul | Blockbibliothek für Snap!, die alle benutzerdefinierten Blöcke enthält, die für das Modul benötigt werden. Kann in das Snap!-Fenster gezogen oder über *Dateimenü -> Importieren*  in Snap! geladen werden.“ |
| ☻ KI-A3 Zusatzmaterial | Speichern und Teilen von Projekten | Speichern von Projekten in der Cloud und lokal als Datei. Teilen von Projekten, die in der Cloud gespeichert sind. |
| ☻ KI-A3 Musterlösung | Musterlösung | Lösungen zu allen Aufgabenstellungen inklusive Links zu den Projekten. |

**Legende**

☻ Material für Schülerinnen und Schüler

☻ Material für Lehrkräfte sowie Unternehmensvertreterinnen und Unternehmensvertreter

☻ Zusatzmaterial

# Glossar

|  |  |
| --- | --- |
| Begriff | Erläuterung |
| Animation | Erzeugen eines bewegten Bildes durch rasche Abfolge von Standortveränderungen, Grafikeffekten oder Teilbildern. |
| Bias | Verzerrung statistischer Ergebnisse aufgrund falscher Untersuchungsmethoden. Im Bereich Künstliche Intelligenz häufgt aufgrund nicht repräsentativer Trainingsdaten. |
| Einstrichgeste (Unistroke Gesture) | Bewegung, die ein Symbol in einem Zug ohne Absetzen aufzeichnet. |
| HCI (Human Computer Interaction) | Forschungsfeld, das sich mit dem Design und der Verwendung von Computer-Technologie an der Schnittstelle zwischen Menschen und Computern beschäftigt. Entwickelt und erforscht Methoden, wie Menschen mit Computern interagieren können. |
| Nächster Nachbar (Nearest Neighbour) | Algorithmus, der aus einer Liste von Kandidaten den auswählt, der bestimmten Kriterien am nächsten kommt. In diesem Fall die Geste aus einer Liste von Beispielgesten, deren Pfad dem Pfad der aktuellen Zeichnung am ähnlichsten ist. |
| Storytelling | Mitteilen von Erkenntnissen, Fakten oder Meinungen, indem sie mit eigenen Erfahrungen und Erlebnissen verknüpft und in für das Zielpublikum ansprechende Geschichten gekleidet werden. |

# Fragen, Feedback, Anregungen

Sie haben das Modul ausprobiert und nun Fragen, Anregungen oder Feedback für uns? Darüber freuen wir uns, denn mit Ihren Erfahrungen können wir Schritt für Schritt einen FAQ (Frequently Asked Questions) für die neuen KI-Module aufbauen oder die Module weiter entwickeln.

Bitte füllen Sie folgende Umfrage über Surveymonkey aus: <https://bit.ly/3DAemzw> über den folgenden QR-Code kommen Sie ebenfalls zur Surveymonkey-Umfrage:



Sie können sich auch gerne unter [bildung@wissensfabrik.de](mailto:bildung@wissensfabrik.de) melden.

# Bildverzeichnis

* S. 6 Lichtgriffel: CC BY SA 3.0 Kerry Rodden, von Wikimedia Commons: <https://commons.wikimedia.org/wiki/File:SketchpadDissertation-Fig4-1.tif>
* S. 6 Sketchpad: CC BY SA 3.0 Kerry Rodden, von Wikimedia Commons: <https://commons.wikimedia.org/wiki/File:SketchpadDissertation-Fig1-2.tif>