



UNTERRICHTSMODUL AUTOMATISIERUNG

ROBOTERSTEUERUNG

ARBEITSBLATT UND LEHRERINFORMATION

Fachinhalte: Programmierung / Vereinfachung von Arbeitsprozessen / Einsatz von Mess-, Steuerungs- und Regelungssystemen

ROBOTERSTEUERUNG

VORAUSSETZUNGEN

Das Unterrichtsmodul lässt sich im Technikunterricht der 8. und 9. Klasse in weiterführenden Schulen einsetzen. Die Schülerinnen und Schüler besitzen grundlegende Kenntnisse von Bauteilen in Computern und haben sich bereits mit den Funktionen von Prozessoren und Platinelementen im Unterricht beschäftigt. Sie sind vertraut mit Computern und Webbrowsern. Sie können eigenständig im Internet recherchieren. Die Schülerinnen und Schüler benötigen für die Aufgabenbearbeitung einen Onlinezugang, z. B. per WLAN oder Zugang zum Schulcomputerraum. In diesem Unterrichtsmodul erfolgt eine Auseinandersetzung mit Möglichkeiten und Grenzen von technischen Systemen, die motorische Tätigkeiten ersetzen, präzisieren und erweitern. Die Schülerinnen und Schüler beschäftigen sich mit Problemstellungen aus dem Bereich des breiten Spektrums der Automatisierung am Beispiel eines virtuellen Roboters, dem „Open Roberta“-Projekt. Dabei erlernen sie die Programmiersprache NEPO und verwenden virtuelle Messsensoren zur Regelung und Steuerung des Roboters. Eine bereits erlernte Programmiersprache kann nützlich sein, ist jedoch nicht notwendig, um die Aufgabenstellung zu lösen.

HINWEISE ZUM STUNDENABLAUF

GESAMTZEIT: 90 MINUTEN

PHASE	INHALT	ZEIT
1. Motivation	Stellen Sie einen automatisierten Vorgang aus dem Alltag vor (z. B. Läuten der Schulglocke > Schülerinnen und Schüler begeben sich in ihre Klasse). Zeigen Sie auf, dass die Kinder und Jugendlichen in diesem Beispiel nach klaren Regeln programmiert wurden, und stellen Sie dies im EVA-Prinzip dar. Anschließend bearbeitet die Klasse Aufgabe 1.	10 Min.
2. Problemstellung und Arbeitsauftrag	Bereiten Sie für die Programmierungsaufgaben die Räume oder Geräte vor und prüfen Sie, ob das Programm überall läuft. Die Leistungsunterschiede der Schülerinnen und Schüler können je nach persönlichem Interesse stark variieren, motivieren Sie Ihre Lerngruppe, sich gegenseitig bei Schwierigkeiten und Unklarheiten zu unterstützen. Falls die Aufgaben wesentlich mehr Zeit in Anspruch nehmen als gedacht oder die Schülerinnen und Schüler Gefallen an der Programmierung gefunden haben, geben Sie zu einem anderen Zeitpunkt mehr Raum für die Arbeit mit der Lernplattform. Dennoch sollten Sie an dieser Stelle rechtzeitig so abbrechen, dass eine Überprüfung und Sicherung der Ergebnisse stattfinden kann.	40 Min.
3. Ergebnisvorstellung	Lösungen und unterschiedliche Programme werden im Plenum vorgestellt. Gehen Sie an dieser Stelle auf die Unterschiede bei den schülereigenen Programmen ein und loben Sie besonders effektive Codes, die Schleifen oder Logikprozesse enthalten. Achten Sie auch auf Programmierungsfehler und gehen Sie auf das Thema „Debugging“ als wichtigen Schritt in der Programmierung ein.	20 Min.
4. Sicherung	Aus den vorgestellten Programmvarianten werden nun gemeinsam Musterlösungen für alle Aufgaben erstellt. Lassen Sie die Codes im Anschluss noch einmal verbalisieren. Dazu sollten Sie besonders lernschwächere Schülerinnen und Schüler auffordern.	20 Min.

BINNENDIFFERENZIERUNG

- ▶ Die Basisaufgabe ist von allen Schülerinnen und Schülern zu lösen.
- ▶ Die Bonusaufgabe ist optional, sie dient als Reserve oder Ergänzung für leistungsstärkere Lernende.

HINWEISE UND LÖSUNGEN ZU DEN AUFGABEN

1. GRUNDLAGEN DER AUTOMATISIERUNG

MUSTERLÖSUNG GRUNDLAGEN

Vorschlag für Liste

Name des Gerätes	Computer	Smartphone	Ampel	Supermarktkasse
Eingabe	Tastatur, Maus	Touchfeld	Druckknopf	Barcodeleser
Verarbeitung	Chipsätze auf Motherboard	Chipsätze	Computer	Kassencomputer
Ausgabe	Bildschirm	Bildschirm	Lichtsignalanlage	Anzeigedisplay
Speicher	Festplatte	SSD	–	Festplatte

Beim EVA-Prinzip handelt es sich um eine Methode zur Datenverarbeitungsanalytik. Dabei steht EVA für Eingabe-Verarbeitung-Ausgabe. Jede Datenverarbeitung kann in diese Matrix eingeordnet werden. Der Speicher gehört nicht zum eigentlichen EVA-Prinzip. Dieser wird im EVA(S)-Prinzip ergänzt.

Sollten die Schülerinnen und Schüler Probleme bei der Gerätefindung haben, können Sie die Beispielgeräte auch vorgeben. Bei der Aufgabenbesprechung sollen die Lernenden zur Schlussfolgerung kommen, dass es am Ende immer Mikrocontroller bzw. Prozessoren auf den Platinen sind, die für die Verarbeitung von Eingaben zuständig sind.

Ein Beispiel für eine menschliche Datenverarbeitung kann das Erkennen von warmem oder kaltem Duschwasser sein. Dabei erfolgt die Eingabe der Temperaturdaten über die Sensoren der Haut, die Verarbeitung erfolgt im Gehirn und die Ausgabe über Muskelreize, die zu einem Erschrecken bei zu kalten Wassertemperaturen führen.

2. MIKROCOMPUTER CALLIOPE MINI

Die Schülerinnen und Schüler finden hier einen Link mit allen Informationen zu Calliope Mini: <https://calliope.cc/>. Dort werden auch die Sensoren (Bewegungs-, Beschleunigungs-, Temperatur-, Helligkeitssensor und Kompass) beschrieben.

Der Minicomputer verfügt über eine LED-Matrix und einen Lautsprecher zur direkten Ausgabe. Weiterhin kann er an Drucker und Displays angeschlossen werden.

Der Mikrocomputer Calliope Mini kann mit der Programmiersprache Scratch programmiert werden. Scratch ist eine grafische Programmiersprache. Dabei werden einzelne Schritte der Programmierung wie Puzzlestücke miteinander verzahnt.

Die Idee der Entwickler ist, dass Calliope Mini an Schulen verteilt wird und so Schülerinnen und Schüler ihren eigenen Mikrocomputer besitzen, den sie programmieren können.

3. DEIN ERSTES PROGRAMM

Für das Erproben von Programmierung eignet sich die Lernplattform „Open Roberta“, entwickelt vom Fraunhofer-Institut, besonders gut. Auch ohne Hardware kann hier anhand einer kleinen Simulation gleich der Erfolg der Programmierung überprüft werden. Hier finden die Schülerinnen und Schüler den Link zur Internetseite: <https://lab.open-roberta.org/>. Die Seite kann sowohl von Computern als auch Smartphones und Tablets besucht werden. Auf allen Geräten kann dann browserbasiert programmiert werden.

Die verwendete Programmiersprache NEPO basiert auf der Programmiersprache Scratch. Mit dieser werden auch Mikrocomputer wie Calliope Mini programmiert (auch dafür lassen sich übrigens auf der Lernplattform Programme schreiben und testen). Somit bietet NEPO einen guten Einstieg in die Grundlagen der Programmierung.

Informieren Sie sich im Vorfeld, ob die von Ihnen verwendeten Rechner und deren Browser mit „Open Roberta“ arbeiten können. Die Schülerinnen und Schüler können sich ein Profil auf der Webseite anlegen und so Programme auch speichern. Sie sollten sich im Vorfeld bereits mit der Programmierung vertraut gemacht haben, um Fragen beantworten zu können. Geben Sie den Schülerinnen und Schülern Zeit, sich die Anleitung unter „Programmieren mit NEPO“ durchzulesen und die Funktionen auszuprobieren.

Roberta verfügt über verschiedene Umgebungen, in denen sich der Roboter bewegen kann. Um eine Umgebung in „Open Roberta“ auszuwählen, klicken Sie auf das „9 Punkte“-Symbol in der oberen Leiste, direkt neben dem Weltkugel-Symbol.

4. SCHLEIFENPROGRAMMIERUNG

Die Umgebung lässt sich über das „9 Punkte“-Symbol in der oberen Leiste auswählen.

Der Roboter wird über den Code programmiert, ein Quadrat abzufahren. Die korrekte Zuordnung der Codebausteine zu ihrer Beschreibung lautet: A2, B1, C4, D3. Sollte das Programm einzelnen Schülerinnen oder Schülern nicht auf Anhieb verständlich sein, fordern Sie sie auf, die Werte zu ändern und die Veränderungen zu beobachten.

Um statt einem Quadrat ein Sechseck oder ein Dreieck nachzufahren, muss der Drehwinkel im Code auf 60 bzw. 120 Grad geändert werden. Um die Strecke nur einmal abfahren zu lassen, wird der Baustein „Wiederhole x-mal, mache“ gewählt und folgende Werte eingegeben: Rechteck 4-mal/Sechseck 6-mal/ Dreieck 3-mal.

Hinweis: Bei mehreren Durchläufen gerät Roberta leider etwas aus der Bahn. Zudem muss öfters die Umgebung „Zeichnen Umgebung“ neu ausgewählt werden, damit die Strecke aufgezeichnet wird.

5. MESSEN UND REGELN MIT SENSOREN

Die Umgebung lässt sich über das „9 Punkte“-Symbol in der oberen Leiste auswählen.

Der Roboter verfügt über mehrere Sensoren, vorgegeben sind ein Ultraschall- und ein Farbsensor sowie ein Berührungs- und Kreiselsensor. Einzelheiten zum Aufbau des Roboters finden Sie mit Klick auf den Reiter „Roboterkonfiguration“.

Vor Beginn des Programms muss der Roboter per Maus vor den Farbfeldern aufgestellt werden. Im Mustercode finden sich mehrere Fahrbefehle. Diese sorgen dafür, den Roboter kurzzeitig aus dem Farbfeld zu lenken, um so den Drehbefehl einzuleiten. Schülerlösungen für die Zwei-Farben-Variation können von der Musterlösung abweichen.

MUSTERLÖSUNG MESSEN UND REGELN

The code starts with a 'Start' block and a 'zeige Sensordaten' block. It enters a 'Wiederhole unendlich oft' loop. Inside, there is a 'mache' block with a 'wenn' condition: 'gib Farbe' from 'Farbsensor Port 3' is green. If true, it performs: 'Fahre vorwärts' (Tempo 80, Strecke 40 cm), 'Drehe rechts' (Tempo 30, Grad 180), and 'Fahre vorwärts' (Tempo 20). If false, it performs 'Fahre vorwärts' (Tempo 20). This sequence repeats for a yellow field.

6. STEUERUNG – PROGRAMMIERUNG EINER FAHRSTRECKE

Die Schülerinnen und Schüler können die Fahrt des Roboters durch Abmessung der Fahrtstrecke oder über die Farbsensoren lösen. Zur Abmessung verwendet man das frei bewegliche Lineal. Die Programmierung kann dann mit genauen Zahlenwerten durchgeführt werden. Eleganter ist die Lösung über die Nutzung der Farbsensoren des Roboters (siehe Musterlösung).

MUSTERLÖSUNG PROGRAMMIERUNG FAHRSTRECKE

The code starts with 'Start' and 'zeige Sensordaten'. It enters a 'Wiederhole unendlich oft' loop. A 'mache' block contains a 'wenn' condition: 'gib Farbe' from 'Farbsensor Port 3' is green. If true, it performs: 'Fahre vorwärts' (Tempo 30, Strecke 71 cm), 'Drehe rechts' (Tempo 30, Grad 88), 'Fahre vorwärts' (Tempo 30, Strecke 30 cm), 'Stoppe', 'Warte bis' (gib Berührungssensor gedrückt Port 1 ist wahr), and 'Fahre vorwärts' (Tempo 50). If false, it performs: 'Fahre vorwärts' (Tempo 50). This sequence repeats for a yellow field, followed by a 'Stoppe', 'Warte ms' (1000), 'Fahre vorwärts' (Tempo 50, Strecke 10 cm), 'Drehe rechts' (Tempo 30, Grad 47), and 'Fahre vorwärts' (Tempo 50).