TECHNISCHE UNIVERSITÄT DRESDEN

Professur für Didaktik der Informatik

Diese Arbeit wurde vorgelegt an der Professur für Didaktik der Informatik

Fahre genau 1 m weit! – ein Experiment für den Informatikunterricht im Kontext von Physical Computing

Belegarbeit in der Veranstaltung „Physical Computing“ im Sommersemester 2023

von

Püschel, Joachim

Mat.-Nr. 3060281

Dozent/in: Dr. Thiemo Leonhardt, David Baberowski

Dresden, 15.09.2023

TECHNISCHE UNIVERSITÄT DRESDEN

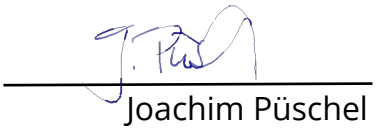
Professur für Didaktik der Informatik

Eidesstattliche Erklärung

Hiermit versichere ich, Joachim Püschel, die Seminararbeit mit dem Titel „Fahre genau 1 m weit! – ein Experiment für den Informatikunterricht im Kontext von Physical Computing“ in der Veranstaltung „Physical Computing“ im Sommersemester 2023 bei Dr. Thiemo Leonhard und David Barberowski selbständig, ohne fremde Hilfe und ohne Benutzung anderer als der von mir angegebenen Quellen und Hilfsmittel angefertigt zu haben. Alle aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche gekennzeichnet.

Die Arbeit wurde noch keiner Prüfungsbehörde in gleicher oder ähnlicher Form vorgelegt. Mir ist bekannt, dass ein Betrugsversuch mit der Note "nicht ausreichend" (5,0) geahndet wird und im Wiederholungsfall zum Ausschluss von der Erbringung weiterer Prüfungsleistungen führen kann.

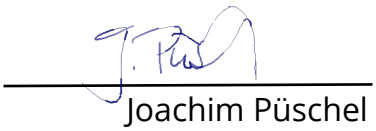
Dresden, 15.09.2023



Joachim Püschel

Für alle von mir selbst erstellten Dokumente erteile ich der Professur für Didaktik der Informatik der Fakultät Informatik der TU Dresden für Zwecke der Lehre und Forschung ein zeitlich und sachlich unbeschränktes, nichtexklusives Nutzungsrecht. Ich gebe alle von mir erstellten Dokumente zur weiteren Nutzung in Lehrveranstaltungen frei und willige insb. der Veröffentlichung in OPAL-Lernräumen ein.

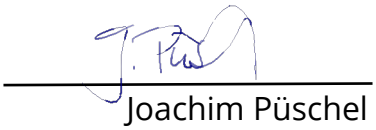
Dresden, 15.09.2023



Joachim Püschel

Darüber hinaus veröffentliche ich die von mir erstellten Konzepte und Lehr-Lern-Materialien unter der Lizenz CC-BY-SA. Damit stimme ich der freizugänglichen Veröffentlichung dieser auf Online-Plattformen (z. B. dem sächsischen Bildungsserver) explizit zu. Hierzu erkläre ich, dass alle von mir verwendeten Materialien und Inhalte entweder von mir selbst stammen oder im Sinne entsprechender Lizenzen gemeinfrei sind.

Dresden, 15.09.2023



Joachim Püschel

Inhaltsverzeichnis

Zusammenfassung

Die Schülerinnen und Schüler (SuS) beschäftigen sich mit der Robotik. Sie erhalten eine Calliope:mini (calliope.cc) samt einem Calli:bot2 (Knotech). Ihre Aufgabe ist es, diesen Roboter möglichst genau einen Meter weit fahren zu lassen. Als Hilfen erhalten Sie Arbeitsblätter für den Aufbau und den grundlegenden Umgang mit den benötigten Modulen. Die benötigte Zeit sollte 90 Minuten nicht überschreiten, kann aber je nach Erfahrung mit den Hilfsmitteln und Programmierwerkzeugen variieren.

Die SuS erarbeiten sich zunächst selbstständig mit den dargebotenen Hilfen den Umgang mit der Calliope selbst, dem Erstellen und Hochladen von Programmen mittels makecode.cc.

Die benötigten Befehle für das Starten und Stoppen der Motoren sowie die Zeitmessung werden danach erarbeitet.

Danach müssen die SuS mit den Parametern Geschwindigkeit der Motoren und der Wartezeit zwischen Starten und Stoppen experimentieren, um die geforderte Strecke von einem Meter zu absolvieren.

Die SuS müssen die Strecke abmessen und mittels Versuch und Irrtum oder Rechnung passende Parameterkombinationen festlegen, implementieren und evaluieren. Dabei müssen sie Platz haben, herumlaufen und sich untereinander austauschen, sodass kognitive, affektive und psychomotorische Lernziele erreicht werden können.

Zur Verbesserung der intrinsischen Motivation könnte wettbewerbsorientiert gearbeitet werden, indem die schnellste(n) Gruppe(n) einen Preis bekommen.

Zur Differenzierung könnte die Aufgabenstellung auf die schnellstmögliche Fahrt gehoben werden (leichte Erschwernis) oder nach dem Fahren mit einem Wendemanöver und der Rückfahrt deutlich erschwert werden.

Teil I Grundlagen

Kapitel 1 Motivation

Physical Computing kann im Schulkontext den Unterricht vor allem in den Bereichen Algorithmierung (Klasse 8) und Projekt (Klassen 9 und 10) unterstützen. Dabei stehen die Aspekte Robotik und Produktion (Digital Fabrication) im Vordergrund. In geringerer Weise ist die nötige Hardwarenähe auch für den Aufbau von Informatiksystemen (Klasse 7) geeignet. Dabei muss man jedoch beachten, dass im Rahmen des Oberschulunterrichts die Hardware am besten möglichst gekapselt ist, damit nicht zu viele einzelne Bauteile verwendet werden müssen und nicht zu viele Routinen oder Schnittstellen gelernt werden müssen. Insbesondere der erste Aspekt ist advers zur Unterrichtung der Schüler:innen, da zwar Ein- und Ausgabeelemente veranschaulicht werden können, diese Teile jedoch nicht als Einzelteile betrachtet werden können und so zum Beispiel die Natur der Anschlüsse oder der Funktionsweise auf der Strecke bleibt.

Der Einplatinencomputer oder Mikrocontroller Calliope mini hat alle benötigten Bauteile für Ein- und Ausgabe aufgelötet, die auch alle Kanäle und Sinne der Schüler:innen bedienen: LEDs (Sehsinn), Lautsprecher (Hörsinn), Lagesensor und Knöpfe (Tastsinn). Außerdem können Erweiterungsplatinen wie der Calli:bot2 ohne Löten oder umfängliche Kennntnisse angeschlossen werden, sodass das Zusammenwirken von Informatiksystemen durch die SuS erkannt werden kann. Zur Platine kommt auch ein großes Ökosystem von Programmierumgebungen (makecode und Roberta) und vorgefertigten Beispielen unterschiedlichen Schwierigkeitsgrads. Die Programmierung kann blockbasiert oder textuell erfolgen, wobei ein schlagender Vorteil der blockbasierten Variante die Vermeidung von Syntaxfehlern und dem Lernen von Befehlen ist. Die einzelnen Blöcke haben sprechende Namen, sodass eine Art Pseudokode implementiert werden kann, der für die SuS lesbar ist.

In dem vorliegenden Experiment wird vor allem der Robotik-Aspekt verfolgt. Die Verlagerung der Arbeitsweise eines Algorithmus und der Ausgaben eines Informatiksystems aus der Abstraktheit eines Rechners auf einen beobachtbaren Gegenstand sorgt für eine

emotionalere Beschäftigung der SuS mit diesen beiden Themen. Sie können unmittelbar die Auswirkung der Änderung von Eingaben (hier Parameter für eine Methode) auf das Verhalten eines Roboters (hier Calli:bot2) beobachten. Durch Änderung und Anpassung dieser Eingaben kann ein erkenntnistheoretischer Prozess in den SuS angestoßen werden, der dem eines naturwissenschaftlichen Experiments ähnelt [Sch18].

Durch die blockbasierte Programmierung im Rahmen dieses Experiments nehmen die SuS diesen Prozess möglicherweise als nicht so abstrakt wahr, sodass diese Vorgehensweise niedrigschwellig ist und die Beobachtung und Messung stärker im Fokus liegen als die eigentliche Programmierleistung. Bei diesem Experiment wird das Programm auch durch die Vorbefassung durch eine Internetpräsenz vorgegeben und muss nur minimal abgewandelt werden. Auch Schüler:innen mit geringer Programmiererfahrung, geringer Frustrationstoleranz oder Ängsten vor dem Umgang mit Technik können hier alle wesentlichen Grundlagen erlernen.

Daneben bietet Physical Computing auch großes Potenzial für Team- und Projektarbeit: Neben der Dokumentation der Messwerte und der Änderung der Programmparameter, die eher die kognitive Lernzielebene taxieren, sind auch Beratung und Entscheidungen in einer Gruppe zentrale Notwendigkeiten und adressieren die affektive Lernzielebene. Außerdem kann Arbeitsteilung und -organisation eingeübt werden, wenn Planung und Durchführung des Experiments unter den SuS verteilt werden.

Kapitel 2 Fachdidaktische Einordnung

Das Thema Physical Computing kann im Lehrplan für Oberschulen in Sachsen (Neubarbeitung 2022) [LP22] in folgende Klassenstufen und Lernbereiche eingeordnet werden:

- Klasse 7, LB1 „Informatik im Alltag“
  - Grundlegende Entwicklungstendenzen
  - Hardware und Software
  - EVA-Prinzip,
  - Grundlegende Handlungsweisen
- Klasse 7, WB1 „Künstliche Intelligenz“
- Klasse 8, LB1 „Algorithmen und Programme“
- Klasse8, WB1 „Steuern und Regeln“
- Klasse 9, LB2 „Komplexaufgabe zur Algorithmierung“
- Klasse 9, WB1 „Informatik und Automatisierung“
- Klasse 10, LK2 „Komplexes Informatikprojekt“
  - nur bei entsprechendem Projektziel

Physical Computing hat im Informatikunterricht den großen Vorteil gegenüber klassischer Programmierung, dass ein haptischer Effekt zur Algorithmierung hinzukommt, der neben den kognitiven Zielen (u. a. Programmieren können) auch psychomotorische (etwas in der Hand halten, messen, drücken, einstellen) und affektive Lernziele bedient. Die affektiven Lernziele werden durch die direkte Rückmeldung der physischen Geräte bedient: Leuchten von LEDs, Bewegung und Töne. Dies spricht alle Sinne der Schüler:innen an und führt schneller und stärker zu emotionaler Beteiligung. Am Ende ist ein deutlich sichtbarer Effekt oder ein Endprodukt entstanden, das stofflich ist.

Herausforderungen im Physical Computing sind insbesondere der Materialeinsatz. Neben einem Rechner ist in der Regel Zusatzhardware vonnöten, die je nach Einsatzzweck durchaus kostspielig sein kann. Beispiele hierfür sind 3D-Drucker oder CNC-Fräsmaschinen. Diese Hardware kann, wenn sie überhaupt angeschafft wird, in der Regel nicht für jede:n Schüle:in angeschafft werden, sondern muss gemeinschaftlich genutzt werden. Das bedeutet automatisch auch, dass die benötigte Zeit zum Fertigstellen eines Projekts steigt, aber auch dass die Geräte zwischenzeitlich anders konfiguriert werden müssen. Das erzwingt besondere Absprachen oder Prozessabläufe in der Gruppe, der Klasse und dem Kollegium.

Zusatzhardware ist aber nicht nur teuer, sondern kann kaputt gehen. Das ist vor allem bei filigranen Teilen wie Steckverbindungen der Fall, aber auch der Verlust von Kleinteilen kann Zusatzhardware unbenutzbar machen. Außerdem müssen so viele unterschiedliche physische Geräte unterhalten werden. Dazu gehört die Konfiguration, das Austauschen von Verbrauchsmitteln, aber auch die Überwachung und Nachbestellung fehlender Teile. Außerdem muss Platz für diese Hardware gefunden werden, der in unmittelbarer Nähe des Unterrichtsraumes ist, aber das Material auch sichert.

Im Unterricht können die den Schüler:innen andere Fehlerqualitäten auftreten. Wo bei fest konfigurierten Systemen oder Programmieroberflächen in der Regel syntaktische oder semantische Fehler auftreten, kann bei Zusatzhardware auch ein Bauteil Fehlfunktionen aufweisen. Es muss dann geprüft werden, ob ein syntaktischer oder semantischer oder ein Hardwarefehler auftritt. Die Propblemlösung ist dann ungleich komplizierter und stört massiv den Ablauf des Unterrichts.

Die optimale Verortung für Physical Computing liegt in Klasse 9 (LB2 oder WB1): Die Schüler:innen können im Rahmen des Projekts dieses Experiment oder ähnliche, anders geartete bearbeiten oder sich selbst solche ausdenken und bearbeiten. Die Arbeitsteilung und Dokumentation ist intrinsischer Teil des Experiments.

Teil I Eigenes Experiment

Kapitel 3 Technische Grundlagen

1. Technische Beschreibung

Die verwendeten Mikrocontroller sind einerseits der Einplatinencontroller Calliope mini und der Calli:bot2 von Knotech.

Die Calliope mini ist eine Zwischenform aus Einplatinencomputer und Mikrocontroller. Sie enthält Eingabeelemente wie Knöpfe, Pins, einen Lagesensor, GPS, Kompass und ein Mikrofon sowie Ausgabeelemente wie Pins, LEDs (5x5-Matrix und ein RGB-LED) und einen Lautsprecher. Sie ist mittels makecode oder OpenRoberta programmierbar und nutzt die Schnittstellen Bluetooth oder USB. Das Gerät hat einen 32-Bit ARM-Prozessor mit 16 MHz, 16 kB RAM und 256 kB Speicher für Programme. Weitere technische Daten finden sich unter https://www.calliope.cc/calliope-mini/technische-daten

Der Calli:bot2 ist ein Zusatzboard für Calliope mini und wird über Grove-Stecker und stromführende Kabel an die Calliope mini angeschlossen. Es enthält folgende Sensoren: 2x IR-Liniensensoren, 1x Ultraschall-Abstandssensor. Außerdem enthält es folgende Aktoren: Metallgetriebemotoren, 4x RGB-LED und 2x rote LEDs, 2x Ausgänge für Servomotoren (https://shop.knotech.de/callibot/244/calli-bot-2).

2. Aktoren und Sensoren

Für das gestellte Problem sind keine Sensoren nötig. Es muss lediglich eine Zeitdauer einstellbar sein. Diese wird über die Calliope mini gesteuert. Als Aktoren dienen unmittelbar die Pins A0 der Calliope mini und mittelbar die Motoren der Räder des Calli:bot2, die über I²C angesteuert werden.

3. Möglichkeiten der Programmierung inklusive der einsetzbaren Entwicklungsumgebungen (MakeCode, OpenRoberta, ...)

Da die Implementation des Programmkodes auf der Calliope mini lauffähig sein soll, können die Programmierumgebungen makecode.calliope.cc oder Open Roberta genutzt werden. Beide können als Blocksprachen oder textuell genutzt werden.

Hier soll makecode genutzt werden, und zwar vorrangig in der Blockansicht (siehe Schaubild 1). Das ist den Kenntnissen der SuS in der Oberschule geschuldet, die auf diese Art insbesondere keine Syntaxfehler machen können. Außerdem sind die genutzten Erarbeitungsunterlagen für die blockbasierte Programmierung erstellt.

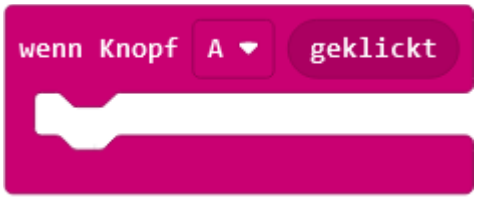


Schaubild 1: Aufruf eines Blocks nach Drücken des Knopfes A

Auch die Interaktion über die Eingabeelemente (*events*) der Calliope mini wird blockbasiert einerseits deutlicher visualisiert: Schaubild 1 im Gegensatz zu Quelltext 1. Andererseits wird die Information didaktisch reduziert und bietet weniger Ablenkung.

```
def on_button_b():
    ...
input.on_button_event(Button.B, input.button_event_click()
, on_button_b)
```

Quelltext 1: Aufruf eines Blockes ... mithilfe der textuellen Repräsentation

Für die Problemlösung im Programm ist keine Schleife vonnöten. Ein schleifenbasierter Ansatz wurde verworfen (Begründung siehe Seite 21).

Kapitel 4 Idee des Experiments

Die Schülerinnen und Schüler (SuS) beschäftigen sich mit der Robotik. Sie erhalten eine Calliope:mini (calliope.cc) samt einem Calli:bot2 (Knotech). Ihre Aufgabe ist es, diesen Roboter möglichst genau einen Meter weit fahren zu lassen. Als Hilfen erhalten Sie Arbeitsblätter für den Aufbau und den grundlegenden Umgang mit den benötigten Modulen. Die benötigte Zeit sollte 90 Minuten nicht überschreiten, kann aber je nach Erfahrung mit den Hilfsmitteln und Programmierwerkzeugen variieren.

Die SuS erarbeiten sich zunächst selbstständig mit den dargebotenen Hilfen den Umgang mit der Calliope selbst, dem Erstellen und Hochladen von Programmen mittels makecode.cc.

Die benötigten Befehle für das Starten und Stoppen der Motoren sowie die Zeitmessung werden danach erarbeitet.

Danach müssen die SuS mit den Parametern Geschwindigkeit der Motoren und der Wartezeit zwischen Starten und Stoppen experimentieren, um die geforderte Strecke von einem Meter zu absolvieren.

Zur Differenzierung kann die Schwierigkeit erhöht werden, indem nach dem Vorwärtsfahren eine 180 °-Drehung und das Rückwärtsfahren zum Startpunkt gefordert werden. In diesem Fall muss dasselbe Problem (Geschwindigkeit, Zeit) für nur einen der beiden Motoren erneut gelöst werden.

Kapitel 5 Ablauf des Experiments

1. Lernziele

Die SuS erlernen den grundlegenden Umgang mit der Platine Calliope mini und makecode, indem sie ein vorgegebenes Programm implementieren und auf die Calliope mini übertragen.

Die SuS erlernen den grundlegenden Umgang mit dem Calli:bot2 , indem sie das Gerät gemäß Anleitung [IS] zusammenbauen und ein vorgegebenes Programm darauf übertragen.

Die SuS lassen den Calli:bot2 genau 1 m weit fahren, indem sie die Parameter für die Methoden „Pause“ und „Motorleistung“ passend wählen.

Die SuS finden passende Parameter für die Methoden „Pause“ und „Motorleistung“, indem sie die Fahrtstrecke messen und die Werte dynamisch ändern.

2. Ablaufplan

a) Einordnung in das Modell von Schulze

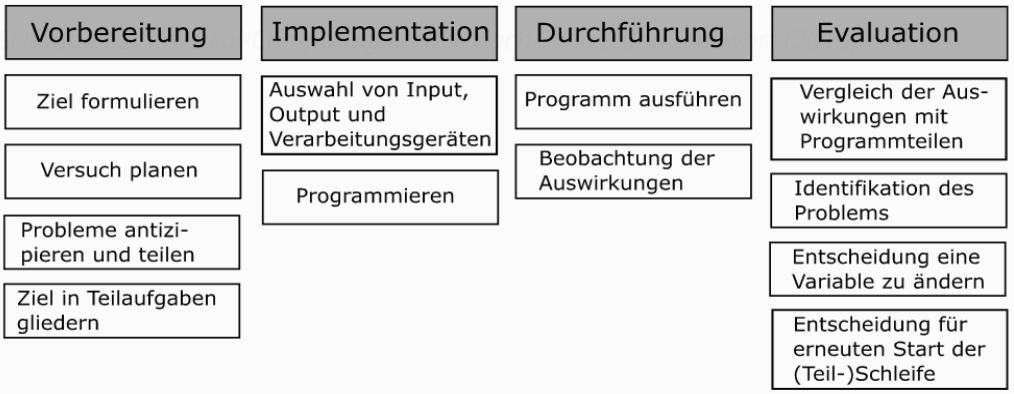


Abbildung 3.1: Physical-Computing-Modell nach Schulz und Pinkwart (2016)

Das vorliegende Experiment umfasst alle Teile des Modells von Schulze und Pinkwart [Sch18]. Allerdings führen die Schüler:innen keine Vorbereitung und keine Auswahl von Input, Output und Verarbeitungsgeräten durch. Diese Phasen wurden vom Autor ausgeführt.

Das Ziel und die Aufgabenuntergliederung findet sich in den entsprechenden Aufgabenblättern `AB_1m_Fahren*.pdf`. Die Schüler:innen werden durch die Arbeitsanleitungen aus [IS] zunächst schrittweise an den Auf- und Zusammenbau des Calli:bots2 herangeführt und schreiben ein vorgegebenes Programm.

Nach Fertigstellung dieser Phase beginnen die SuS mit der Teilphase „Programmieren“ aus der Hauptphase „Implementation“, indem sie ihr geschriebenes Programm modifizieren. Sie erhalten am Ende eine Sequenz, die aus den drei Anweisungen `SchalteMotor(beide,vorwärts,`**`leistung`**`)`, `Pausiere(`**`zeit`**`)` und `StoppeMotor(beide,bremsend)` besteht. Die Parameter `leistung` und `zeit` sind die entscheidenden Größen für dieses Experiment.

In der Hauptphase „Durchführung“ führen sie das Programm aus und beobachten die Auswirkungen, hier die Fahrtstrecke. Sie messen diese. Hierfür sollte die Lehrperson behutsam passende Vorgaben machen, also zum Beispiel die tabellarische Notation von gefahrener Strecke in Abhängigkeit der gewählten Parameter `leistung` und `zeit`. So können die Schüler:innen ihre Fortschritte uind Abhängigkeiten besser erkennen und gelangen schneller zum Ziel.

In der Evaluationsphase vergleichen die SuS ihre Beobachtung mit den Programmteilen, hier die Fahrtstrecke mit den Parametern `leistung` und `zeit`. Sie identifizieren das Problem, indem sie erkennen, dass sie entweder zu weit oder zu kurz gefahren sind. Sie erkennen, dass sie die Parameter `leistung` oder `zeit` ändern müssen, um die Fahrtstrecke von 1 m exakt zu erreichen. Je nach beobachteter Fahrtstrecke müssen sie verringert oder erhöht werden. Nach Änderung dieser Parameter springen sie zur Durchführung zurück. (Nach Lektüre von [Sch18] scheint mir die Implementation explizit unter Ausschluss der veränderlichen Parameter gedacht zu sein, denn sonst müsste formal zur Implementation zurückgesprungen werden. Die Implementation meint also nur die grundlegende Auslegung des Algorithmus.)

Die Schüler:innen messen in jeder Iteration die gefahrene Strecke und passen entsprechend die Parameter `leistung` oder `zeit` an.

Sie könnten, bei Verständnis des Prozesses, auf mögliche Optimierungen des Prozesses kommen, die vom bloßen Raten über systematische Änderungen zur zielgerichteten Änderung der Parameter führen. Diese Gedanken werden in den Tippkarten des Blocks A in der Datei `AB_Tippkarten.pdf` als Handreichungen unterstützt und angeregt.

Systematische Änderungen der Parameter werden durch Verdoppeln und Halbieren schneller zu brauchbaren Parametereinstellungen führen als beispielsweise Änderungen um nur konstante Werte.

Einige Schüler:innen könnten außerdem auf die Vereinfachung kommen, dass einer der Parameter auf sinnvollem Wert konstant gehalten werden kann und somit nur der andere verändert werden muss, was deutlich schneller zum Ziel führt. Allerdings kann es sein, dass diese Denkweise von der Lehrkraft speziell lanciert werden und an passenden Beispielen erklärt werden muss.

Eine Differenzierung für leistungsfähigere SuS ist die Ausdehnung der Aufgabe auf 1 m vorwärtsfahren, danach eine 180 °-Drehung und dann die Rückfahrt zum Startpunkt. Die Prozesse sind dieselben, bei der Drehung wird nur einer der beiden Motoren bewegt, aber auch hier müssen wieder die Parameter `leistung` oder `zeit` passend gewählt werden. Auf Seite 16 ist eine Musterlösung gezeigt.

3. Benötigte Materialien

Es werden benötigt: Calliope mini, Calli:bot2, 3xAAA-Batterien, USB-Kabel (mini-USB, USB-A), [IS], [CM], Lineal (1 m), evtl. Taschenrechner und Winkelmesser, Schreibzeug

4. Ausgearbeitete Arbeitsblätter oder Arbeitsaufträge

a) Arbeitsaufträge

Die formulierten Arbeitsaufträge sind in den Dateien `AB_1m_Fahren.pdf` bzw. `AB_1m_Fahren_und_drehen.pdf` abgelegt. Die Arbeitsaufträge sind leistungsdifferenziert. Sie enthalten einerseits

Hinweise zum Vorgehen, aber auch Hinweise auf die bereitliegenden Tippkarten, um die Frustration abzufangen.

Der längere Arbeitsauftrag ist leistungsdifferenziert, weil er dasselbe Grundproblem mehrfach lösen lässt.

b) Musterlösungen

In den folgenden Schaubildern 4 und 3 sind Musterlösungen abgebildet. Um 1 m weit zu fahren, muss das Produkt aus Motorleistung und -laufzeit 4150 betragen, das bedeutet bei 50 % Motorleistung jeder Motor 8300 ms laufen muss. Die Differenzierung nach Schnelligkeit würde also bei 100 % Leistung eine Laufzeit von etwa 4,1 s benötigen. Wollte man den Calli:bot nur 10 cm fahren lassen, müsste man ein Produkt von 415 nutzen.

Für eine Drehung um 90 ° muss das Produkt auf der Leistung **genau eines** Motors und der Laufzeit 474 betragen. In der Musterlösung läuft der linke Motor bei 25 % Leistung 1895 ms.

Bei der Nutzung dieser Lösungen muss Folgendes in Betracht gezogen und vorher getestet werden: Haftung der Räder am Boden, Akkustand der Batterien und leicht abweichende Leistung der beiden Motoren.

Schaubild 4: Musterlösung: 1 m fahren

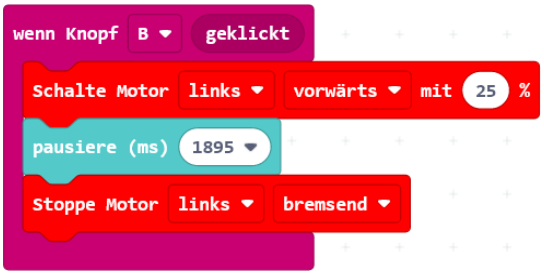


Schaubild 3: Musterlösung: 90°-Drehung

c) Weitere Materialien

Tippkarten sind in der Datei `AB_Tippkarten.pdf` zu finden. Sie können laminiert, ausgeschnitten und an die Schüler:innengruppen ausgegeben werden.

Die Tippkarten sind in zwei Blöcke unterteilt. Block A ist für alle Schüler:innengruppen, Block B nur für die anspruchsvollere Aufgabe, die zusätzlich das Umdrehen erfordert. Sie sollten in der jeweiligen Reihenfolge ausgegeben werden, also z. B. A2 nach A1, da die Tipps immer schneller zur Lösung führen, aber auch mehr Sachkunde erfordern.

Teil II Fazit und Ausblick

Kapitel 6 Fazit

Im Blockpraktikum wurden grundlegende Kenntnisse zum Umgang mit physischen Bauteilen wie Aktoren, Sensoren und Platinen sowie der Umgang mit Programmierumgebungen für diese vermittelt.

Der inhaltlich für die Lehre an einer Oberschule am besten geeignete Aspekt war die Bereitstellung von schüler:innengerechten Arbeitsblättern bzw. Internetauftritten als OER, die einen niedrigschwelligen und klug strukturierten Einstieg bieten.

Außerdem wurde vermittels einer Publikation [Sch18] die Alignierung naturwissenschaftlicher und informatischer Erkenntnisgewinnung versucht. Diese Alignierung ist grundsätzlich in Ordnung, aber zum Teil nicht ganz stichhaltig, denn auch die Naturwissenschaften zergliedern Probleme in Teilaufgaben, z. B. Vorversuche zur Festlegung geeigneter Positiv- und Negativkontrollen. Ein Beispiel hierfür wird in [Pü18] die Nutzung von γ- bzw. α-Tocopherol als geeignete Positivkontrolle angeführt. Die Publikation räumt zwar ein, dass „Abweichungen des [vom?] idealtypischen Prozesses möglich“ seien [Sch18, Seite 55], scheint aber grundlegende Experimentierprozesse anders zu bewerten als dies ein Naturwissenschaftler tun würde.

Es treten auch inhaltliche Fehler auf, die eventuell durch klarere Sprache vermieden werden könnten: „In der 2. und 3. Spalte werden den Hauptphasen [die] Teilphasen zugeordnet, in denen sie [die Hauptphasen*/die Teilphasen**] auftreten können.“ (ebda, Seite 53, Ergänzungen durch den Autor). Die Aussage des Satzes ist insoweit falsch, als dass Hauptphasen (Oberbegriff) nicht innerhalb von Teilphasen (Unterbegriff) auftreten können (Beispiel *). Wählt man die Lesart gemäß Beispiel ** erhält man eine Zuordnung von Teilphasen zu Teilphasen, die laut Schema auf Seite 54 nicht auftreten. Eine dritte Lesart ist unmöglich.

Das Blockpraktikum vermittelte einerseits Aspekte der Robotik, indem es Mikrocontroller (Calliope mini, ESP32) und Erweiterungsplatinen (Calli:bot2, Zusatzboard mit gekapselten Grundfunktionen beim ESP32) zur Verfügung stellte und niedrigschwellige Aufgaben anbot.

Außerdem wurden die vielfältigen Werkzeuge aufgezählt und die Allgemeingültigkeit der entsprechenden Schritte dargelegt. Der zweite Aspekt war die Fertigung von Produkten (Digital Fabrication) mittels des Mate-Plotters oder 3D-Druck. Hier wurde vor allem der Design- und Planungsprozess von der Ideen- über die Modellbildung über die Transformation und Interpretation zum Produkt beleuchtet. Dabei war die Dialektik zwischen den Zuständen auf Produktebene und den Schritten der Umsetzung auf Planungsebene zentraler Bestandteil des Kurses. Das vorliegende Experiment nimmt aber auf die Robotik Bezug.

Für das in diesem Beleg vorgeschlagene Experiment wurden die im Blockpraktikum vorgestellten Werkzeuge Calliope mini und Calli:bot2 samt der OER zum Erstellen eines Programms und zur Ansteuerung von Motoren genutzt. Darauf aufbauend wurden einerseits eine Musterlösung sowie Hinweiskarten konzipiert. Das Experiment umfasst alle Phasen aus [Sch18], wobei die Hauptphase „Vorbereitung“ komplett und die „Auswahl von Input, Output und Verarbeitungsgeräten“ aus der Hauptphase „Implementation“ vom Fachlehrer durchgeführt werden. Das „Programmieren“ und die Hauptphasen „Durchführung“ und „Evaluation“ werden von den Schüler:innen durchgeführt. Dabei gibt es für jede Phase Hilfen, die es den Schüler:innen ermöglichen, im Prozess fortzuschreiten, wenn sie nicht weiterkommen.

Das erstellte Material verlässt sich in großen Teilen auf die von der Professur für Didaktik der Informatik gestellten Vorlagen (OER) sowie die Handlungsanweisungen auf [IS]. Die Aufgabenstellung wurde mittels der zur Verfügung gestellten Vorlage erstellt. Die Tippkarten folgen ebenfalls dem Aufbau der Vorlagen. Die Lehrkraft kann sie an ihre eigenen Erfordernisse anpassen (Ausgabe, Aushängen, digital, …).

Das Blockpraktikum ermöglichte viele Iterationen der eigenen Prozesse und nutzte auch Sensoren, welche die ablaufenden Programme allgemeiner machen sollten. Der Autor versuchte sich an einer Drehung des Calli:bot2 mittels des eingebauten Kompasses. Ein Ansatz, das GPS zur Positionsbestimmung und damit zur Längenmes-

sung einzusetzen, wäre entsprechend auch möglich und würde eine intelligentere Lösung des Problems bedeuten. Den Schüler:innen ist jedoch von solchen Lösungen abzuraten, außer sie sind sehr frustrationstolerant.

Eine Drehung mittels Kompassfunktion (Schaubild 5) scheiterte an der mangelnden Auflösung des Kompass und fehlender Verlässlichkeit der Messung.

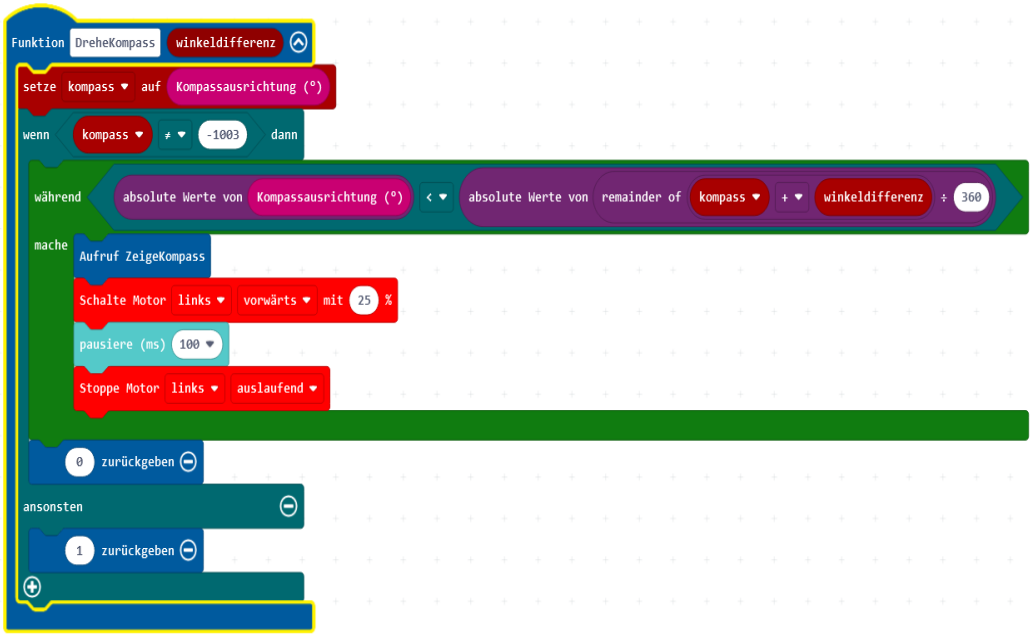


Schaubild 5: Funktion DreheKompass(winkeldifferenz)

Anhang A Literaturverzeichnis

[CM] Der Calliope mini (PDF), Arbeitsmaterial B8.2 GS, offis e. V. Und Universität Oldenburg, 2021, CC-BY-SA-NC

[IS] inf-schule, https://www.inf-schule.de/kids/calliope/callibot-einstieg/, 13.09.2023, CC BY-SA 4.0

[LP22] Lehrplan Informatik für Oberschulen (Neubearbeitung 2022), Sächsisches Staatsministerium für Kultus, https://www.schulportal.sachsen.de/lplandb/index.php?lplanid=514&lplansc=eO3sY571qOGqEqH3YXT3&token=004503b80012395c5bae7da512f4d03

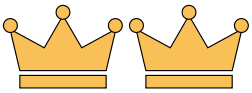
[Pü18] Püschel, J.: „Über die Eignung von Haarwurzelkulturen von *Helianthus annuus* (Ha) L. zur biotechnologischen α-Tocopherol-Biosynthese“, Dissertation, 2018

[Sch18] Schulze, S.: Physical Computing als Mittel der wissenschaftlichen Erkenntnisgewinnung in der Informatik und als fächerverbindende MINT-Arbeitsweise, Dissertation, 2018

TECHNISCHE UNIVERSITÄT DRESDEN

Tippkartensatz

für Calli:bot – fahren und drehen



Professur für Didaktik der Informatik

A1 TIPP (ZEITERSPARNIS BEI VERSUCH UND IRRTUM 1)

Statt die Methodenparameter für die Fahrtzeit immer zu raten, haltet zum Beispiel die Motorgeschwindigkeit fest und ändert nur die Fahrtzeit.

A2 TIPP (ZEITERSPARNIS BEI VERSUCH UND IRRTUM 2)

Um schneller zum richtigen Wert zu kommen, nutzt für eure Beobachtungen immer Verdoppelungen oder Halbierungen. So kommt ihr schnell zu den passenden Werten.

A3 TIPP (ZEITERSPARNIS BEI VERSUCH UND IRRTUM 3)

Statt die Methodenparameter für die Fahrtzeit immer zu raten, nutzt eure Messergebnisse direkt aus, um eure Zielwerte zu errechnen.
Wenn ihr mit eurer Fahrtzeit nur 30 cm statt 1 m schafft, müsst ihr eure Fahrtzeit folgendermaßen berechnen:

$$\text{neue Fahrtzeit} \quad t = \frac{1\,m}{0{,}3\,m} \cdot \text{bisherige Fahrtzeit}$$

B1 TIPP (ZEITERSPARNIS BEI FAHRTWEG UND DREHUNG)

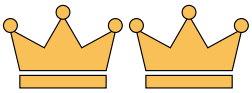
Da ihr nach jedem Ändern eurer Methodenparameter das Programm neu aufspielen müsst, kostet das wertvolle Zeit. Nutzt die beiden Knöpfe, um mit einem Knopf den Fahrtweg und mit dem anderen Knopf den Drehwinkel zu beobachten. So könnt ihr beim Ändern des Programms immer zwei Werte gleichzeitig ändern.

B2 TIPP (MESSVERBESSERUNG BEI DREHUNG)

Wenn ihr die Drehung auf einen Knopf programmiert, dann lasst sich den Callibot mehrmals direkt nacheinander drehen, z. B. zweimal. Ihr könnt dann besser messen, wie groß eure Abweichung ist.

TECHNISCHE UNIVERSITÄT DRESDEN

Fahre genau 1 m weit



Professur für Didaktik der Informatik

AUFGABE

Macht euch mit eurem Calli:bot2 vertraut und baut ihn korrekt zusammen. Programmiert ihn so, dass er vorwärts und rückwärts fährt.

HINWEIS

Nutzt dazu die folgende Website (oder den QR-Kode):
https://www.inf-schule.de/kids/calliope/callibot-einstieg

ACHTUNG

Ihr arbeitet mit Strom. Achtet auf die richtige Polung der Batterien und die richtige Verkabelung.

TEILZIEL ERREICHT

Euer Calli:bot2 ist jetzt in der Lage:

- vorwärts zu fahren,
- kurz zu stoppen und danach
- rückwärts zu fahren.

AUFGABE

Lasst euren Calli:bot jetzt möglichst genau 1 m geradeaus fahren. Danach soll er sich um 180 ° drehen und zum Ausgangspunkt zurückfahren.

HINWEIS

Bearbeitet dazu euer bisheriges Programm.
Überlegt euch, welche Parameter (Zahlen) ihr ändern müsst, messt die Distanz und ändert diese Zahlen erneut bis ihr euer Ziel erreicht habt.

Um den Calli:bot sich drehen zu lassen, lasst nur einen der beiden Motoren drehen.

MATERIALIEN

Ihr benötigt ein langes Lineal (1 m) oder Maßband oder einen Zollstock.
Außerdem benötigt ihr Platz!

WEITERE HILFE

Falls ihr nicht weiter kommt, fragt nach den Tippkarten.

TECHNISCHE UNIVERSITÄT DRESDEN

Fahre genau 1 m weit

Professur für Didaktik der Informatik

AUFGABE

Macht euch mit eurem Calli:bot2 vertraut und baut ihn korrekt zusammen. Programmiert ihn so, dass er vorwärts und rückwärts fährt.

HINWEIS

Nutzt dazu die folgende Website (oder den QR-Kode):
https://www.inf-schule.de/kids/calliope/callibot-einstieg

ACHTUNG

Ihr arbeitet mit Strom. Achtet auf die richtige Polung der Batterien und die richtige Verkabelung.

TEILZIEL ERREICHT

Euer Calli:bot2 ist jetzt in der Lage:

- vorwärts zu fahren,
- kurz zu stoppen und danach
- rückwärts zu fahren.

AUFGABE

Lasst euren Calli:bot jetzt möglichst genau 1 m geradeaus fahren.

HINWEIS

Bearbeitet dazu euer bisheriges Programm.
Überlegt euch, welche Parameter (Zahlen) ihr ändern müsst, messt die Distanz und ändert diese Zahlen erneut bis ihr euer Ziel erreicht habt.

MATERIALIEN

Ihr benötigt ein langes Lineal (1 m) oder Maßband oder einen Zollstock.
Außerdem benötigt ihr Platz!

WEITERE HILFE

Falls ihr nicht weiter kommt, fragt nach den Tippkarten.