ScratchDrone – Systematische Programmierung von Flugdrohnen für den Informatikunterricht

Raphael Zender, Julius Höfler, Patrick Wolfien, Ulrike Lucke

Institut für Informatik
Universität Potsdam
August-Bebel-Str. 89
14482 Potsdam
vorname.nachname@uni-potsdam.de

Abstract: Informatik- und insbesondere Programmierunterricht sind heute fester Bestandteil der schulischen Ausbildung. Vereinfachte Entwicklungsumgebungen, die auf die Abstraktion typischer Programmierkonzepte in Form von grafischen Bausteinen setzen, unterstützen diesen Trend. Zusätzliche Attraktivität wird durch die Verwendung exotischer Laufzeitumgebungen (z. B. Roboter) geschaffen. Die in diesem Paper vorgestellte Plattform "ScratchDrone" führt ergänzend zu diesen Angeboten eine moderne Flugdrohne als innovative Laufzeitumgebung für Scratch-Programme ein. Die Programmierung kann dabei dank modularer Systemarchitektur auf drei verschiedenen Abstraktionsebenen erfolgen. Kombiniert mit einem mehrstufigen didaktischen Modell, der Herausforderung der Bewegung im 3D-Raum sowie der natürlichen menschlichen Faszination für das Fliegen, wird so eine hohe Lernmotivation bei jungen Programmieranfängern erreicht.

1 Motivation

Das Spektrum von Förderungsangeboten für Programmmieranfänger ist dank innovativer und erschwinglicher Soft- und Hardware in den letzten Jahren beträchtlich gewachsen. Insbesondere aus dem Bereich Robotik kommt heute eine Vielzahl von Systemen, die in den Informatikunterricht an Schulen und Hochschulen eingebunden werden können (z. B. *LEGO Mindstorms*[1], *NAO*[2]). Der eher hohe Aufwand, den Lehrende bei der Einbindung derartiger Systeme in ihren Unterricht haben, sowie die entstehenden Kosten bremsen jedoch die Verbreitung. Stärker verbreitet sind hingegen kostenlose und offene Systeme wie vereinfachte Entwicklungsumgebungen (IDEs), mit denen grundlegende Programmierkonzepte leicht vermittelt und motivierend erlernt werden können. Beispielsweise ist die grafische IDE *Scratch* [RMMH+09] an vielen Schulen im Einsatz und erfreut sich großer Beliebtheit.

Scratch setzt auf die Abstraktion typischer Programmierkonzepte in Form von grafischen Bausteinen. Programmieranfänger, die ein derartiges Programmierkonzept nutzen, müssen

[1] http://www.lego.com/de-de/mindstorms
[2] http://www.aldebaran.com/en/humanoid-robot/nao-robot

Abbildung 1: Die AR.Drone 2.0 kann im Indoor- und Outdoor-Bereich mit jeweils angepassten Schutzhüllen eingesetzt werden. [Bildquelle: Parrot SA]

sich weniger auf die Syntax konzentrieren. Der Cognitive Load-Theorie folgend verfügen sie so über mehr lernbezogene kognitive Kapazität für den primären Lerngegenstand, die Semantik einer Programmiersprache [BTZS12]. Zudem werden Computersysteme mit physikalischen Sonsoren und Aktoren (*Physical Computing* [SLMR05]) zunehmend alltagsrelevant. Die durch den Alltagsbezug entstehende Motivation wird in der Scratch-Community bereits durch eine Reihe von Erweiterungen zur Einbindung physikalischer Systeme nutzbar gemacht.

Dieser Beitrag stellt das Framework *ScratchDrone* zur Einbindung der in Abbildung 1 dargestellten *AR.Drone 2.0* – einer modernen, kostengünstigen Flugdrohne – als attraktive Laufzeitumgebung in die Scratch IDE vor. Im Vergleich zu anderen Einstiegsplattformen für Programmmieranfänger werden die "begreifbare" Zielplattform [UI00], die natürliche menschliche Faszination für das Fliegen sowie die Herausforderung der Bewegung im 3D-Raum als Motivation ausgenutzt.

Zudem wird mit ScratchDrone ein Unterrichtskonzept in mehreren didaktischen Stufen angestrebt. Lehrer werden in die Lage versetzt, ihre Schüler von der reinen Bedienersicht über die Zusammenstellung einfacher und komplexer werdenden Flugmanöver in Scratch bis hin zur Programmierung einer Flugdrohne in einer marktrelevanten Programmiersprache (z. B. Java, Javascript, PHP) zu begleiten. Eine modulare Systemarchitektur stellt die dafür notwendigen Programmierschnittstellen auf drei verschiedenen Abstraktionsebenen zur Verfügung.

Nach einer Vorstellung der relevanten verwandten Arbeiten wird im folgenden ScratchDrone im Detail mit seinem didaktischen Konzept und der technischen Umsetzung erläutert. Zudem werden erste Evaluierungsergebnisse präsentiert. Abschließend werden dieses Paper zusammengefasst und ein Ausblick auf weitere Arbeiten mit ScratchDrone gegeben.

2 Verwandte Arbeiten

Die Einbettung von fachlichen Lehrinhalten in realistische, physische Settings ist eine beliebte Methode zur Förderung der Informatik- und Programmierausbildung. Kreative Projekte wie das *Computer-Freundebuch* [Luc11] zur Vermittlung von Computerwissen unter der Metapher eines klassischen Freundebuches, *E-Mail (nur?) für Dich* [GHW11] zur Erarbeitung von Grundlagen der Kommunikation in Computernetzen sowie die physische Herstellung programmierter, interaktiver Objekte aus dem Informatikunterricht [PR13] zeigen das breite Anwendungssprektrum dieser Lehrstrategie.

Aufgrund der Popularität von Scratch als Entwicklungsumgebung für Programmieranfänger findet sich heute weiterhin bereits eine Reihe von daraus ansteuerbaren Laufzeitumgebungen, in der Regel unter einer OpenSource-Lizenz veröffentlicht. Zudem ergibt die Fachliteratur vielfältige Ansätze zur Nutzung derartiger Medien im Bildungsprozess – beispielsweise zur Gestaltung von Lehreinheiten im Bereich Robotik [GSRLLO13]. Insbesondere durch derartige Projekte ist die Scratch-Erweiterung *Enchanting* [LMM12] zur Programmierung von LEGO Mindstorm-Systemen bekannt, die eine angepasste Firmware nutzt. Auch Scratch-Derivate zur Einbindung von Mikrocontrollern[3] und Gestenerfassungssystemen[4] in Scratch-Programmen sind verfügbar. Dabei kommt häufig die Scratch-Erweiterung *Build your Own Blocks* (BYOB) bzw. *Snap!*[5] des MIT Media Lab zum Einsatz. Diese erleichtert die Erstellung eigener Scratch-Blöcke, in denen dann die Kommunikation mit der speziellen Hardware gekapselt wird.

Auch die in der vorliegenden Arbeit verwendete Laufzeitumgebung, die AR.Drone 2.0, ist aufgrund des relativ geringen Anschaffungspreis (ca. 300 Euro), der Verfügbarkeit offener Schnittstellen sowie einer aktiven Entwickler-Community Gegenstand verschiedener Forschungsprojekte. Für den Bereich der Aus- und Weiterbildung ist beispielsweise das Projekt *Catdroid* [Sla12] von Interesse. Die Android-Software ermöglicht ebenfalls die Programmierung von Flugdrohnen über eine Scratch-ähnliche grafische Entwicklungsoberfläche.

Sämtliche Lösungen für Scratch und einzelne Laufzeitumgebungen sind als Einzellösungen zu betrachten. Das erschwert den Einsatz in einer größeren, zusammenhängenden Unterrichtssequenz bzw. in verschiedenen Schwierigkeits-/Abstraktionsstufen. Das im folgenden beschriebene Framework setzt im Unterschied dazu auf einen flexibleren und systematischen Ansatz durch einen modularen Architekturaufbau. Dieser ermöglicht beispielsweise mehrstufige didaktische Szenarien für aufeinander aufbauende Unterrichtsstufen sowie eine freie Technologiewahl durch den Lehrenden.

[3] https://github.com/MrYsLab/s2a_fm
[4] http://scratch.saorog.com
[5] http://snap.berkeley.edu

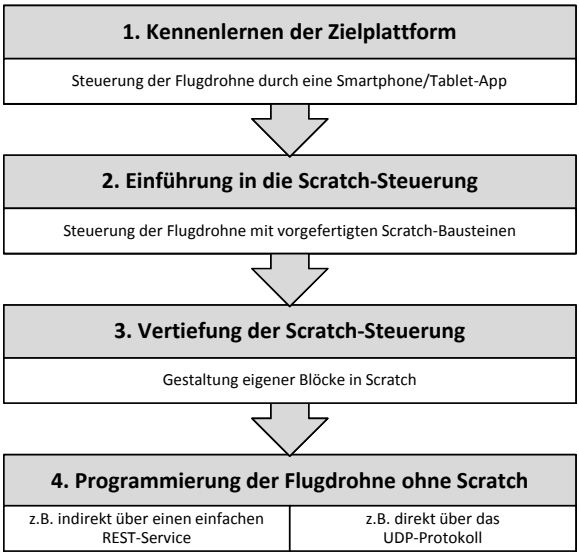

Abbildung 2: Der Einsatz von ScratchDrone ist didaktisch in ein vierstufiges Unterrichtskonzept eingebunden.

3 ScratchDrone

Das Ziel von ScratchDrone ist nicht nur eine weitere, für Schüler interessante Laufzeitumgebung für die Scratch-Programmierung anzubieten. Die Steuerung der Flugdrohne über Scratch ist nur für zwei der vier didaktischen Ebenen relevant, für die das Framework im Programmierunterricht genutzt werden kann.

3.1 Didaktisches Unterrichtskonzept

ScratchDrone wurde für den Einsatz im Informatikunterricht an Schulen entwickelt und profitiert ggf. von Scratch-Erfahrungen, die in Informatikklassen bereits vorliegen. Das Lernziel varriert je nach Kenntnisstand der Schüler und besteht im Allgemeinen aus der Vertiefung der eigenen Programmierkenntnisse am Beispiel eines konkreten physikalischen Systems. Die Zielgruppe reicht dank des einfachen Scratch-Programmiermodells von Schülern, die noch nie programmiert haben, bis hin zu erfahrenen Schülern aus Informatik-Leistungskursen oder zu Studienbeginn. Möglich ist dies durch ein vierstufiges didaktisches Unterrichtskonzept, das in Abbildung 2 skizziert ist. Es orientiert sich an einem klassischen Curriculum, in dem das Unterrichtsthema vom Abstrakten (Steuerung über eine stark vereinfachte App) zum detalliert-konkreten (Programmierung mit typischer Programmiersprache) erarbeitet wird.

In der ersten Stufe lernen die Schüler zunächst mit Werkzeugen des Herstellers die Drohne kennen. Insbesondere die Reaktion auf Wind im Freien, auf Hindernisse im Flugbereich sowie die Geschwindigkeit und Präzision der Umsetzung von Steuerbefehlen sollten zu Beginn vermittelt werden. Flugdrohnen wie die in diesem Projekt verwendete AR.Drone 2.0 kommen beispielsweise mit Smartphone- oder Tablet-Apps, die per WLAN mit der eigentlichen Drohne kommunizieren und für die ersten Flugmanöver eingesetzt werden können. Neben den eigentlichen Flugbewegungen können über diese Apps auch Videos aufgenommen und Sensordaten abgerufen werden, so dass den Schülern das technische Potential der Drohnentechnologie vermittelt wird. Diese Stufe kann je nach Klassenstärke in ca. einer Schulstunde absolviert werden.

In der zweiten Stufe wird erstmals mit Programmierelementen gearbeitet. Die Schüler können mit vorgefertigten Scratch-Blöcken in dem Scratch-Derivat BYOB einfache Flugsequenzen programmieren. Die entsprechenden Blöcke werden mit ScratchDrone mitgeliefert und ständig erweitert. Derzeit können 17 Befehle genutzt werden. Dazu gehören:

- 3 Startkommandos (unterschiedliche Höhen)
- 1 Landekommando
- 6 3D-Bewegungen (z. B. *Drohne fliegt 30 cm nach links*)
- 2 Schwebekommandos (unterschiedliche Dauer)
- 2 Drehungen (je 45 Grad nach links und rechts)
- 2 beispielhafte Kombinationen der oberen Blöcke
- 1 RESET-Kommando (um ggf. Fehler-Zustände zu verlassen)

Die Befehle finden sich im Scratch-Bereich *Bewegung* und können mit anderen Scratch-Elementen kombiniert werden (z. B. Schleifen, Verzweigungen, Zufallszahlengenerator). Scratch-erfahrene Schüler können zudem Scratch-Ereignisse (z. B. Tastatureingaben) verwenden, um die asynchrone Flugsequenz mit synchronen Elementen anzureichern (z. B. für eine Live-Steuerung per Tastaturbefehlen). Nachdem die Schüler ihre Scratch-Programme entwickelt haben, werden diese eingesammelt und über den Lehrerrechner mit der Drohne ausgeführt. Dadurch ist sichergestellt, dass der Lehrer die Programme vor ihrer Ausführung prüfen kann, um riskante Manöver zu identifizieren und ggf. zu vermeiden. Dazu gehören beispielsweise Flüge in über 30 m Höhe oder in Bäume, die die Hardware beschädigen würden. Idealerweise werden die Programme je nach Witterung im Freien oder in einer Turnhalle bzw. großen Aula der Schule gemeinsam mit den Schülern ausgeführt. Erfahrungsgemäß wird für diese Stufe bei einer Klassenstärke von 10-15 Schülern eine Doppelschulstunde benötigt.

Die dritte Stufe entspricht vom organisatorischen Ablauf der zweiten Stufe. Allerdings sind die Schüler nun aufgefordert eigene Blöcke zur Drohnensteuerung zu entwerfen. Dabei können sowohl existierende ScratchDrone-Blöcke wiederverwendet als auch von Grund auf neue Blöcke erstellt werden. Die Schüler nähern sich somit etwas weiter der eigentlichen Programmierung über eine marktrelevante Programmiersprache. Stufe 2 und

3 können zudem während des Unterrichts parallel laufen, um gezielt besonders leistungsstarke Schüler zu fördern (Stufe 3).

In Stufe 4 kann die Flugdrohne ohne Scratch, in einer vom Lehrer zu wählenden Programmiersprache, programmiert werden. ScratchDrone umfasst zu diesem Zweck eine Kapselung der hardwarenahen Steuerung über UDP-Befehle durch REST-Services, die von nahezu beliebigen Programmiersprachen angesteuert werden können. Sofern der Lehrer sich mit seinem Unterricht inhaltlich weiter der eigentlich Hardware nähern möchte, besteht zudem die Möglichkeit, die eigentlichen UDP-Kommandos der AR.Drone 2.0 zu nutzen. In diesem Fall ist das ScratchDrone-Framework nicht mehr erforderlich, da diese Option vom Hersteller selbst unterstützt wird.

Je nach verfügbarer Zeit und Wissensstand werden die Stufen für einen konkreten Unterricht ausgewählt. Das komplette Unterrichtskonzept eignet sich eher für 1-2 Projekttage, vor allem während der ersten Programmiererfahrungen. Sinnvoll für den kompakten ScratchDrone-Einsatz ist hingegen eine Doppelschulstunde, in der Stufe 1 mit einer weiteren der Stufen 2-4 kombiniert wird. Ferner ist dieses Konzept nur ein Vorschlag der Verwendung von ScratchDrone. Die Informatiklehrer an den verschiedenen Schulen haben jede Freiheit, dieses Konzept anzupassen, um es den Lernzielen einer konkreten Klasse sowie dem jeweiligen Fortschritt anzupassen.

3.2 Technische Realisierung

Das vierstufige Unterrichtskonzept erfordert ein modulares Framework, das die Programmier- und Benutzungsschnittstellen für die einzelnen didaktischen Stufen 2-4 anbietet. Dieses wurde an der Universität Potsdam im Rahmen zweier Bachelorarbeiten realisiert und wird derzeit im Rahmen einer weiteren Bachelorarbeit optimiert. Die Grobarchitektur ist in Abbildung 3 dargestellt.

Der AR.Drone 2.0 werden über WLAN Steuerkommandos gesendet. Zu diesem Zweck wurden vom Hersteller Parrot eine Reihe von UDP-Kommandos spezifiziert [PBED12]. Zudem kann über die UDP-Schnittstelle auf Sensoren und Kamerabilder der AR.Drone 2.0 zugegriffen werden. Diese wurden noch nicht zur Verwendung in ScratchDrone implementiert, sind aber für künftige Updates vorgesehen. Sofern der Lehrer eine Programmierung auf der Transportschicht des ISO/OSI-Modells lehrt, kann er diese UDP-Befehle direkt nutzen. Da im schulischen Informatikunterricht in der Regel auf höheren Schichten gelehrt wird, wurde im Rahmen des ScratchDrone-Projektes eine REST-Schnittstelle zur servicebasierten Steuerung der Drohne per HTTP implementiert. Diese stellt neben der nativen UDP-Schnittstelle die zweite Programmieroption auf der didaktischen Stufe 4 dar.

Als Sicherheitsmechanismus kann der Lehrer zudem über das Framework jederzeit per Tastatursteuerung in eine laufende Flugsequenz eingreifen und die Drohne so z. B. landen oder Hindernisse umfliegen. Ferner bietet die AR.Drone 2.0 selbst bereits eine Reihe von Sicherheitsmechanismen wie beispielsweise das automatische Absinken auf 6 m beim Verbindungsabbruch und die rudimentäre Kompensierung von Windböhen. Daher ist diese Plattform auch für Unerfahrene unkompliziert einsetzbar.

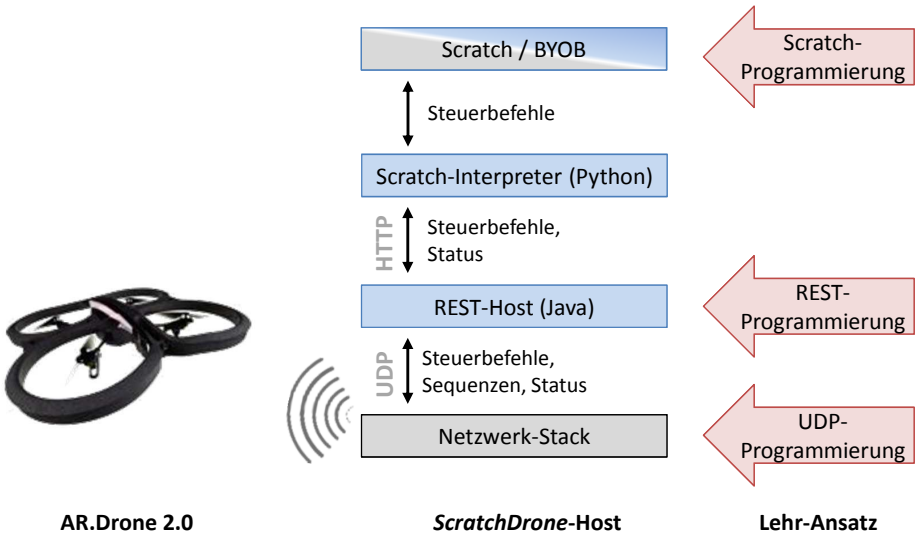

Abbildung 3: Die mehrstufige ScratchDrone-Architektur bietet verschiedene Abstraktionsstufen für den Programmierunterricht.

Die Verbindung zwischen Scratch bzw. BYOB wird durch einen Python-Connector hergestellt. Dieser ist in der Lage, die Scratch-Befehle zu interpretieren und in REST-Kommandos umzusetzen. Zudem dient dieses Python-Programm als Referenzimplementierung für das Ansprechen der REST-Schnittstellen, da Python aufgrund seiner Einfachheit in einer Reihe von Schulen bereits als Programmiersprache gelehrt wird.

Abbildung 4 zeigt die Benutzungsschnittstelle auf der obersten Schicht von ScratchDrone. Als IDE kommt BYOB zum Einsatz, das neben der typischen Scratch-Funktionalität die Definition von eigenen Blöcken erlaubt. Somit sind auf dieser technischen Ebene die didaktischen Stufen 2 und 3 durchführbar. Neben der klassischen Drag-and-Drop-GUI von Scratch wurde auch das Vorschaubild durch ein Foto der Flugdrohne auf das ScratchDrone-Szenario angepasst. Zumindest einfache Manöver können so bereits vor der physischen Ausführung als Scratch-Animation simuliert werden.

Durch die modulare Systemarchitektur können Lehrer ScratchDrone flexibel gemäß ihren eigenen Lehrplänen in den Unterricht einbinden. Sie werden technologisch nicht eingeschränkt, sobald sie die einfache Scratch-Ebene verlassen und eine marktrelevante Programmiersprache einführen möchten. Dank der REST-Kapselung der konkreten Drohnensteuerung ist zudem der Wechsel auf eine andere Flugdrohnen-Hardware mit vergleichsweise geringem Programmieraufwand möglich. Weiterhin ermöglicht diese Abstraktionsschicht das Steuern der Drohne über mehr als einen Client. Somit kann der Lehrer jederzeit das aktuell laufende Programm abbrechen oder anderweitig einschreiten, falls dies beispielsweise aus Sicherheitsgründen erforderlich ist.

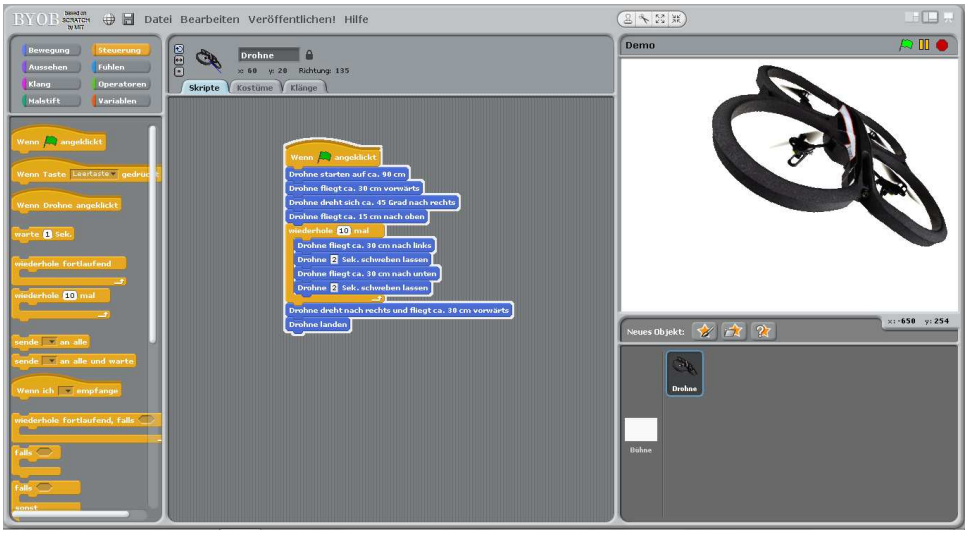

Abbildung 4: Das Scratch-Derivat BYOB wurde um Befehle zur Drohnensteuerung erweitert.

4 Evaluierung

Die Evaluierung von ScratchDrone erfolgt derzeit in Kooperation mit verschiedenen Schulen – vorwiegend Gymnasien – in Berlin und Brandenburg. Der Fokus liegt dabei zunächst auf der motivationalen Ebene. Erste Ergebnisse liegen aus zwei Testläufen vor. Aufgrund der geringen Teilnehmerzahl von insgesamt nur 32 Schülerinnen und Schülern sind die folgenden Ergebnisse als tendenziell zu betrachten. Die Tests erfolgten mit einer 9. und einer 11. Gymnasialklasse, jeweils in einer Doppelstunde des Informatikunterrichts. In beiden Klassen wurden eine Einführung in die Drohnentechnologie sowie die Scratch-Modifikationen gegeben und anschließend Stufe 2 (Einführung in die Scratch-Steuerung) des Unterrichtskonzeptes ausgeführt.

Die 20 Schüler der 9. Klasse waren größtenteils Programmieranfänger. Rudimentäre Programmiererfahrungen lagen aus dem Informatikunterricht in Scratch und Python zwar vor, dennoch stellte die Programmierung mit Scratch noch eine Herausforderung dar. Flugdrohnen kannten die Schüler bereits aus den Medien oder von einschlägigen Computerspielen. Die Schüler wurden von einem Informatiklehrer sowie einem Lehramtsstudenten für Informatik im Praktikum betreut.

Die 12 Schüler der 11. Klasse nahmen an einem Leistungskurs Informatik teil und besaßen dementsprechend schon mehr Programmiererfahrung. Neben Python und Scratch lagen beispielsweise bei allen Teilnehmern Java-Kenntnisse und vereinzelt Erfahrungen mit weiteren Programmiersprachen vor. Flugdrohnen waren ebenfalls nur aus den Medien und Computerspielen bekannt. Betreut wurden die Teilnehmer von einem Mitarbeiter aus

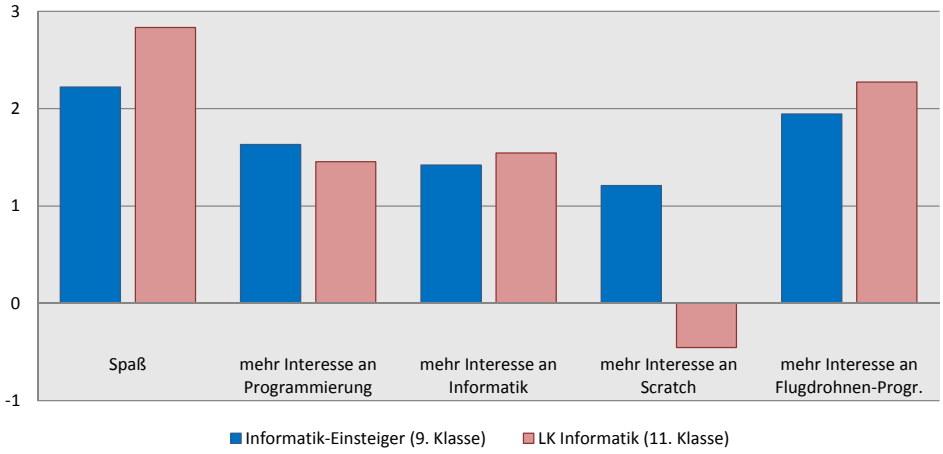

Abbildung 5: Erste Evaluierungsergebnisse belegen die motivationale Wirkung von ScratchDrone für Anfänger und fortgeschrittene junge Programmierer.

dem ScratchDrone-Projekt sowie ihrem Informatik-Lehrer und einem Lehramtsstudenten für Informatik im Praktikum.

Im Anschluss wurden die Teilnehmer gebeten einen Feedback-Fragebogen auszufüllen und freies, verbales Feedback zu geben. Auf einer Likert-Skala wurden der Spaß mit ScratchDrone (von 3: *sehr spaßig* bis -3: *sehr langweilig*) sowie der Interessenszuwachs (*mehr Lust auf* ...) allgemeine Programmierung, Informatik, Scratch und Flugdrohnen-Programmierung (jeweils von 3: *auf jeden Fall* bis -3: *auf keinen Fall*) erfragt. Abbildung 5 fasst die Ergebnisse der Evaluierung vergleichend zusammen.

Sowohl die Einstieger als auch die fortgeschrittenen Informatikschüler hatten erwartungsgemäß großen Spaß an der Drohnenprogrammierung. Vor allem die exotische Plattform, kombiniert mit den schnellen Erfolgserlebnissen dank Scratch, haben dazu beigetragen. Der Spaß war bei dem Leistungskurs noch etwas höher. Es ist anzunehmen, dass bei der Gruppe ohnehin eine höhere IT-Affinität vorliegt, so dass sie sich stärker für die Experimente begeistern konnte.

Bei beiden Gruppen konnte zudem das Interesse an Programmierung und Informatik gesteigert werden. ScratchDrone eignet sich somit gut für die Motivation beider Themen sowie als Abwechslung zum traditionelleren Informatik-Unterricht.

Die Schüler der 9. Klasse haben zudem einen Interessensgewinn für die Scratch-Programmierung erfahren. Vor allem die durch Scratch wahrgenommene Leichtigkeit der Programmierung war für diese Gruppe neu und hat sich auch positiv auf das Scratch-Interesse ausgewirkt. Für den Leistungskurs wurde das Interesse an Scratch eher leicht geschwächt. Die Schüler fühlten sich durch Scratch eher beschränkt in ihren Möglichkeiten die Drohne zu steuern. Bestätigt wird diese Erkenntnis durch ein großes Spektrum an Vorschlägen aus dieser Gruppe für weitere ScratchDrone-Flugmanöver und -Features.

Das große Interesse beider Gruppen an der Programmierung von Flugdrohnen untermauert die Annahme, dass vor allem durch die exotische Laufzeitplattform ein deutlicher Motivationsgewinn erzielt werden kann. Weitere Evaluierungen – vor allem der Stufe 4 – müssen zeigen ob die Motivation über eine einfache Programmierumgebung wie Scratch hinaus auch für komplexere, marktrelevante Programmiersprachen ausgenutzt werden kann.

In beiden Gruppen kam zudem vereinzelt der Wunsch auf, tiefer in die Drohnensteuerung einzusteigen, als die vorgefertigten Scratch-Blöcke in Stufe 2 es erlauben. Auch von Seiten der Lehrer wurde die Nutzung der Flugdrohne für andere Programmiersprachen – wie für Stufe 4 vorgesehen – angeregt. Diese Stufen werden in den kommenden Evaluierungsdurchgängen ebenfalls Unterrichtsgegenstand sein, so dass das systematische Unterrichtskonzept in seiner Gänze fundierter bewertet werden kann.

5 Zusammenfassung und Ausblick

Der Einstieg in die Programmierung wird heute durch eine Reihe von Förderangeboten erleichtert. Insbesondere durch einfache IDEs wie die grafische Programmierumgebung Scratch können schon früh motivierende Erfolge erzielt werden. Das in diesem Paper vorgestellte Framework ScratchDrone nutzt die in Schulen breit vorhandene Scratch-Erfahrung, um Schülern die Programmierung einer Flugdrohne als exotische Laufzeitumgebung zu ermöglichen.

Im Vergleich zu ähnlichen Angeboten (z. B. aus dem Bereich Robotik) wird durch ScratchDrone die Komplexität der Bewegung im 3D-Raum fokussiert. Zudem ermöglicht die modulare Systemarchitektur den Einsatz im Informatikunterricht auf unterschiedlichen Abstraktionsstufen der Programmierung. Das resultierende didaktische Unterrichtskonzept nutzt vier aufeinander aufbauende Stufen der Programmierung, um Anfänger mit der Thematik vertraut zu machen. Einzelne Stufen können weiterhin autonom unter Berücksichtigung der Fachkenntnisse der Schüler ausgewählt und durchgeführt werden. Eine Modifikation der ScratchDrone-Architektur ist dafür nicht erforderlich.

Dank einer Kapselung der eigentlichen Drohnensteuerung hinter einer Reihe von plattformunabhängigen REST-Services bleibt dem Lehrer – über Scratch hinaus – die Entscheidung überlassen, welche Programmiersprache er lehren will. Eine Referenzimplementierung für Python liegt vor. Selbst die Drohnen-Hardware kann ausgetauscht werden, sofern die REST-Kapselung bedient wird. Die grundlegende ScratchDrone-Architektur muss auch dafür nicht geändert werden.

Erste Evaluierungsergebnisse belegen den Erfolg von ScratchDrone bei der Motivation von Anfängern und fortgeschrittenen Programmierschülern.

Neben und in weiteren Evaluierungsläufen im April bis Juli 2014 werden auch eine Reihe organisatorischer und technischer Neuerungen sukzessive konzipiert, implementiert und getestet. Dazu gehören:

- Einbindung eines dreidimensionalen Simulators in Scratch, durch den die Drohnenbefehle durch jeden Schüler virtuell getestet werden können, bevor die physische Drohne gesteuert wird.
- Erstellung neuer ScratchDrone-Blöcke, darunter Sensorabfragen (z. B. Flughöhe) und weitere Flugmanöver (z. B. Salto in der Luft).
- Erstellung einer Live-CD für Lehrer, die den derzeit noch anspruchsvollen Setup-Prozess durch eine fertige Laufzeitumgebung vereinfacht.
- Konzeption von Flugmissionen (z. B. Parcours-Flug) zur weiteren Motivation und um die Programme der Schüler zielgerichtet vergleichen zu können.
- Untersuchung weiterführender Evaluierungsitems (z. B. Lernerfolg)

Das derzeit noch prototypische ScratchDrone-Framework wird zudem im Zuge dieser Überarbeitungen unter einer OpenSource-Lizenz für den freien Einsatz im schulischen Programmierunterricht zur Verfügung gestellt.

Danksagung

Die Autoren danken den Schülern und Lehrern aus Berlin/Brandenburg, die bei der Evaluierung mitgewirkt haben, für ihre Aufgeschlossenheit und Kreativität beim Ausprobieren von ScratchDrone.

Literaturverzeichnis

[BTZS12] Philipp Brauner, Hendrik Thüs, Martina Ziefle und Ulrik Schroeder. ScratchTab - Eine Tablet-basierte Anwendung zum Erlernen von Programmierkonzepten. In *in Proc. 2. Workshop Mobile Learning im Rahmen der DeLFI 2012*, Seiten 15–22. Fernuniversität Hagen, 2012.

[GHW11] Andreas Gramm, Malte Hornung und Helmut Witten. E-Mail (nur?) für Dich - Eine Unterrichtsreihe des Projekts Informatik im Kontext. *LOG IN*, 31(169/170), Nov 2011.

[GSRLLO13] Juan Felipe García Sierra, Francisco J. Rodríguez Lera, Camino Fernández Llamas und Vicente Matellán Olivera. Inside the Maze: Who Would Find the Cheese First, a Robot or a Mouse?: Teaching IT Using Robots. In *Proceedings of the First International Conference on Technological Ecosystem for Enhancing Multiculturality*, TEEM '13, New York, NY, USA, 2013. ACM.

[LMM12] Aidan Lane, Bernd Meyer und Jonathan Mullins. *Robotics with Enchanting and LEGO NXT - A Project Based Introduction to Programming*. Monash University, 2012.

[Luc11] Ulrike Lucke. Das Computer-Freundebuch: Ein Ansatz für Informatik in der Grundschule. In *14. GI-Fachtagung Informatik und Schule (INFOS)*, Münster, 2011.

[PBED12] Stephane Piskorski, Nicolas Brulez, Pierre Eline und Frederic D'Haeyer. *AR.Drone Developer Guide SDK 2.0*. Parrot S.A., 2012.

[PR13] Mareen Przybylla und Ralf Romeike. Physical Computing mit "My Interactive Garden". In *15. GI Fachtagung Informatik und Schule (INFOS)*, Seiten 87–91, Kiel, 2013.

[RMMH+09] Mitchel Resnick, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner, Eric Rosenbaum, Jay Silver, Brian Silverman und Yasmin Kafai. Scratch: Programming for All. *Commun. ACM*, 52(11):60–67, November 2009.

[Sla12] Wolfgang Slany. Catroid: A Mobile Visual Programming System for Children. In *Proceedings of the 11th International Conference on Interaction Design and Children*, IDC '12, Seiten 300–303, Bremen, 2012. ACM.

[SLMR05] John A. Stankovic, Insup Lee, Aloysius Mok und Raj Rajkumar. Opportunities and obligations for physical computing systems. *Computer*, 38(11):23–31, Nov 2005.

[UI00] Brygg Ullmer und Hiroshi Ishii. Emerging Frameworks for Tangible User Interfaces. *IBM Syst. J.*, 39(3-4):915–931, Juli 2000.