PROJEKTSKIZZE

„Wenn Kinder selbst programmieren/Code schreiben, dann
- begreifen sie die Abstraktionen, die den Kern der Mathematik darstellen
- modellieren sie dynamisch mathematische Konzepte und Beziehungen
- gewinnen sie Selbstvertrauen in ihre Fähigkeiten und ihr Wirken als Mathematiklernende“
(Bescherer & Fest, 2019, S. 118)

ÜBERSICHT

Diese Projektskizze informiert über das Forschungsprojekt „ProMaPrim“ (Programmieren im Mathematikunterricht der Primarstufe), gegründet an der Universität Duisburg-Essen, in der Didaktik der Mathematik.

- Ansprechpartner/ Projektverantwortlicher: Dr. Ulrich Schwätzer (Universität Duisburg-Essen)
- Wissenschaftliche Supervision: Prof. Dr. Florian Schacht (Universität Duisburg-Essen)
- Kooperationsschule zur Erprobung der Materialien: Reichshof-Grundschule, Dortmund
- Laufzeit: Ab WS 20/21 ca. 2 Jahre, ggf. verlängert

KERNPUNKTE DES PROJEKTES

Das Ziel des Projektes ist es, die Erforschung von Lernumgebungen zum Programmieren im Mathematikunterricht der Grundschule zu betreiben. Dabei wird die Programmierumgebung Scratch (*scratch.mit.edu*) verwendet.

Inhaltlich geht darum, mathematische Themen mit algorithmischen Strukturen in Reichweite von Grundschüler*innen der Klassen 3/4 zu identifizieren, die sich für diese Altersgruppe sowohl eignen, Gegenstand von lösbaren Programmierproblemen zu sein, als auch Gegenstand einer vertiefenden mathematischen Exploration. Das Projekt geht dabei zwei Forschungsfragen nach:

- **Forschungsfrage 1:** Inwieweit gelingt es Schüler*innen in der Grundschule (Klasse 3/4), Elemente algorithmischer Strukturen aus mathematischer (Paper-Pencil-)Tätigkeit in kohärenten algorithmischen Strukturen auf Programmebene zu modellieren?
- **Forschungsfrage 2:** Inwieweit erlangen diese Schüler*innen mit den fertigen Programmwerkzeugen weiteren Einblick in die mathematischen Zusammenhänge des Ausgangsproblems?

Die Forschungsmethodik des Projektes orientiert sich dabei an iterativen Zirkeln nach dem Design Research Forschungsansatz (Gravemeijer & Cobb, 2006; Prediger, Gravemeijer & Confrey, 2015), ggf. aufgesplittet in Teilprojekte. Der Forschungsprozess wird dabei in mehrere Teilsegmente aufgeteilt, so dass ggf. Teilprojekte entstehen. Der prospektive Forschungsverlauf ist dabei folgender Abbildung zu entnehmen:

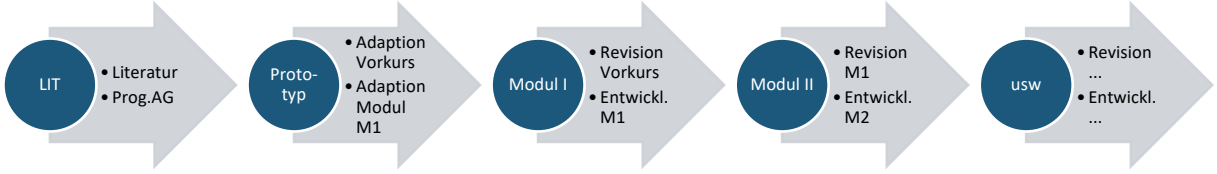

Abb. 1. Makrozyklen der Lokalen Instruktionstheorie

Im WS20/21 haben wir dem Design Research Ansatz folgend die erste Version der lokalen Instruktionstheorie formuliert. Die lokale Instruktionstheorie und der ihr zugrundeliegende Forschungsstand (Literaturstudium) und vorstudienähnlichen Erfahrungshintergrund aus Programmier-AGs (Schwätzer 2018) wird weiter unten vorgestellt.

Auf dieser Grundlage werden wir ebenfalls im WS20/21 damit beginnen, Module eines mathematikdidaktisch orientierten Programmierkurses zu entwerfen, bestehend zunächst aus einem Vorkursmodul (Adaption eines bereits empirisch erprobten Kurses, Einführung in informatische Grundideen und die Programmierumgebung Scratch) und eine ersten mathematischen Modul M1, in dem dann die Umsetzung und Vertiefung eines mathematischen Themas erfolgt.

In dem Design Research Prozess typischen Schleifen sollen dann immer wieder Kursmodule empirisch (z.B. durch Analyse von Screen- und Audiocaptures der Programmierprozesse von Kindern im pair-programming in verschiedenen Stichproben und Settings) erprobt und überarbeitet werden (siehe Folgeprozesse). Dabei fließen die Erfahrungen aus den Vorsegmenten optimierend in den Prozess mit ein. Das Ziel ist es, am Ende einen praktikablen und leicht einsetzbaren mathematikdidaktischen Programmier- und Forscherkurs für die Unterrichtspraxis zu entwickeln.

POSITIONEN: LOKALE INSTRUKTIONSTHEORIE (LIT)

An dieser Stelle erläutern wir in Auszügen unsere lokale Instruktionstheorie. Diese fasst nach dem Design Research Paradigma sämtliche für den thematischen Rahmen relevanten Theoriebezüge zusammen und projiziert sie auf die Intervention. Dabei bleibt sie nicht starr, sondern entwickelt sich im Forschungsprozess weiter. Die Entwicklung der Lernumgebung geht also parallel mit der Optimierung der lokalen Instruktionstheorie einher, der Forschungsprozess wird theorieemergent.

Im Folgenden möchten wir unsere Grundpositionen stichwortartig darlegen, die Eingang in die lokale Instruktionstheorie gefunden haben.

Grundsätzlich hat sich der Zug der Digitalisierung in den (Grund-)schulen auf die Überholspur gesetzt, nicht zuletzt aus den Erfahrungen im Distanzlehren während der COVID-19 Pandemie. Dies trifft aber vor allem auf die prospektierte technische Ausstattung zu. Didaktiker*innen aller Fächer – für unser Projekt wichtig vor allem die der Informatik und Mathematik – reklamieren eine große Forschungslücke und damit einen großen Forschungsbedarf, wie digitale Medien – hier im speziellen kindgerechte Programmierumgebungen – sinnvoll in einem (Mathematik-)Unterricht eingesetzt werden, der Problemlösen und konstruktive Zugänge als nachhaltig ansieht.

Zumindest Grundideen der Informatik, die in der Grundschule thematisiert werden sollten, hat die Didaktik der Informatik bereits formuliert (vgl. Best et al., 2019). Zudem fordern erste Bundesländer bereits curriculare Inhalte. Nordrhein-Westfalen nennt im lehrplan-ähnlichen ‚Medienkompetenzrahmen NRW' etwa „Algorithmische Muster und Strukturen in verschiedenen Kontexten erkennen, nachvollziehen und reflektieren" sowie „eine strukturierte, algorithmische Sequenz planen; diese auch durch Programmieren umsetzen" (Medienberatung NRW., o. J.). Da in der Grundschule kein eigenständiges Fach Informatik existiert, liegt es nahe, dass algorithmische Strukturen auch und besonders im Fach Mathematik verortet sein könnten. Die Fachdidaktiken sind nun aufgefordert, über die Entwicklung tragfähiger Aufgabenbeispiele diesen Bereich in die Praxis zu implementieren.

Wenngleich der Forschungsbedarf als hoch ausgewiesen wird, so gibt es erste empirische Studien, die wir im vorliegenden Projekt in den Referenzrahmen einbeziehen:

- Aus der Didaktik der Informatik der Technischen der Universität München liefert eine dem Design Research verpflichtete Studienreihe von Geldreich et al. (stellvertretend: Geldreich et al., 2019) einen

empirisch erprobten, an den Grundideen der Informatik orientierten und an konstruktivem Lernen orientierten Programmierkurs für Grundschüler*innen der Klassen 3/4, allerdings an einem außermathematischen Thema.
- Bescherer & Fest (2019) aus der Didaktik der Mathematik der PH Ludwigsburg liefern Erfahrungen aus einer ebenfalls empirischen Design Research Studie, die eine Programmierumgebung mit elementaren mathematischen Themen ebenfalls für Kinder der Klassen 3 und 4 erprobte.
- Schwätzer (2018) erprobte Programmierumgebungen in Arbeitsgemeinschaften an einer Grundschule und erarbeitete dazu vor allem eine Reihe von algorithmischen Themen u.a. aus Arithmetik und Geometrie in der Reichweite von Grundschüler*innen im gleichen Alter.

Alle drei genannten Studien setzen nicht nur auf entdeckendes Lernen innerhalb der Programmierumgebung Scratch, sondern auch auf ‚unplugged' Programmieren als wichtigen Baustein, wenngleich in unterschiedlichen Bereichen – während Schwätzer zunächst ‚unplugged' algorithmische Mathematik mit den Kindern betreibt, und diese dann in Programmstrukturen übersetzen lässt, steht ‚unplugged' Programmieren bei ersteren vor allem für den Einstieg in informatische Grundideen.

Zudem scheint ‚pair-programming', also Programmieren in Partnerarbeit, ein wichtiger Schlüssel zum Erfolg zu sein, wie nicht nur die beiden ersteren Studien herausstellen, sondern auch in einer quantitativ empirischen Studie (vgl. Iskrenovic-Momcilovi, 2019) nachgewiesen wurde.

Darüber hinaus enthalten alle Bezugsstudien wichtige Elemente geeigneten Scaffoldings, um den Kindern die problemlösende Umsetzung algorithmischer Strukturen in der Programmierumgebung zu ermöglichen.

Keine uns bekannte Studie hat sich dagegen mit der Nutzung fertiger, durch Kinder selbst erstellter Programme als Werkzeug für mathematische Erkenntnisse tiefergehend beschäftigt. Wenngleich diese Idee von Bescherer & Fest (2019) angedacht ist, so lieferte diese Studie hier keine Ergebnisse, sondern reklamiert weiteren Forschungsbedarf.

Aus diesem Referenzrahmen heraus wollen wir einen empirisch erprobten Programmierkurs entwickeln, der neben der Umsetzung mathematisch-algorithmischer Strukturen in Programme auch das Ziel hat, mit den fertigen Programmen dann wieder Forscherfragen in der Mathematik beantworten zu lassen. Zudem soll durch die Erforschung dieser Lernumgebung der genannte Referenzrahmen und die von uns aufgestellte lokale Instruktionstheorie weiterentwickelt und zur Diskussion gestellt werden.

PROGRAMMIERKURS

Hier wollen wir nun erste Einblicke in die Entwicklung und Erprobung des Programmierkurses bieten, der bzw. dessen Entwicklung gleichsam auch Antworten auf unsere Forschungsfragen geben soll. Wir gehen dabei von folgenden Grundkonstanten aus:

- Als Programmierumgebung ist Scratch allgemein anerkannt und damit gesetzt.
- Der Kurs soll aus (zunächst) zwei Teilen bestehen
  1. einer Einführung in die Programmierumgebung Scratch, orientiert an erprobten Zugängen der Referenzprojekte, adaptiert auf Inhalte der Mathematik,
  2. und aus zunächst einem (später mehreren) Modulen zu einem mathematischen Thema.
- Die Auswahl des/ der mathematischen Themen der Kursmodule soll nicht nur an der Umsetzbarkeit in der Programmierumgebung, sondern vor allem auch an der Reichhaltigkeit für sich anschließende mathematische Explorationen orientiert sein.
- Die Entwicklung der Module erfolgt dabei in mehreren iterativen Zirkeln gemäß dem Design Research Ansatz (siehe oben).

Im aktuellen, gerade begonnenen Forschungsprozesssegment steht die Entwicklung eines Prototyps des Programmierkurses im Fokus, der aus zwei wesentlichen Elementen besteht:

1. Adaption des Vorkurses von Geldreich et al. (2019) mit Wechsel des dort verwendeten Kontextes „Zirkus“ auf einen mathematischen Kontext unter Integration der Befunde von Bescherer und Fest (2019).
   - Übernommen werden dabei
     - die Kompaktheit des Kurses,
     - der Einstieg mit einem ‚unplugged‘ Problem,
     - die Übertragung auf Scratch,
     - kollaboratives aktives Lernen, im ‚pair-programming‘,
     - das Arrangement als Lernzirkel für eigene Lerntempi, und
     - das Abdecken der Grundideen der Informatik Anweisung, Sequenz, Wiederholung und Bedingung.
   - Adaptiert werden
     - der Inhalt durch Transfer auf ein basales mathematisches Thema,
     - die höhere Kompaktheit des Einstiegs durch inhaltliche Fokussierung,
     - die erste Erprobung von Vertiefungen und Reflexionen des gewählten mathematischen Themas.

2. Adaption eines erprobten mathematischen Inhaltes von Schwätzer (2018) auf die Kursstruktur, aber auch dessen Ausweitung auf die zweite Forschungsfrage (Einblick in mathematische Zusammenhänge mit dem fertigen Programmwerkzeug).
   - Beibehalten werden dabei
     - der Paper-Pencil-‚unplugged‘-Einstieg in das mathematische Thema zur (Re-)Aktivierung der zugrundeliegenden Grundvorstellungen,
     - die Übertragung der algorithmischen Struktur in die Programmierumgebung Scratch als konstruktiver Prozess,
     - die Bereitstellung einer teilvorgefertigten Programmierumgebung im Sinne sinnvollen Scaffoldings,
   - Modifiziert wird dabei
     - das Grundsätzliches Arbeiten im ‚pair-programming‘,
     - die Übertragung der Scaffolding-Elemente des Vorkurses,
     - die Hinterfragung geeigneter Zwischenschritte der Übertragung der algorithmischen Struktur in die Programmierumgebung – ggf. sind hier mehrere Entwurfsvarianten nötig:
       - Über ‚unplugged‘-Scratch nach Geldreich et al. (2019),
       - über Programmablaufpläne nach Schwätzer (2018),
       - über die Möglichkeit des reaktiven Testens direkt in der Programmierumgebung (angeregt durch Bescherer & Fest, 2019),
     - das Hinzufügen von ‚Forscherfragen‘ zum Teilen mit Rest mit geeigneten Scaffolding-Elementen, und
     - das Hinzufügen eines abschließenden Reflexionsprozesses der gesamten Lerngruppe.

Erste Einblicke in den Prototyp des Basismodul des Programmierkurses sind auf der Webseite *promaprim.de* unter Downloads einsehbar.

LITERATUR

Bescherer, C. & Fest, A. (2019). Mathematik und informatische Bildung. Programmieren mit Scratch. (Medienpädagogik interdisziplinär. 12). In T. Junge & H. Niesyto (Hrsg.), Digitale Medien in der Grundschullehrerbildung. Erfahrungen aus dem Projekt dileg-SL (S. 117–130). München: kopaed.

Best, A., Borowski, C., Büttner, K., Freudenberg, R., Fricke, M., Haselmeier, K. et al. (2019, Februar 7). Kompetenzen für informatische Bildung im Primarbereich. Zugriff am 30.10.2020. Verfügbar unter: http://dl.gi.de/handle/20.500.12116/20121

Geldreich, K., Simon, A. & Hubwieser, P. (2019). Design-Based Research als Ansatz zur Einführung von Algorithmik und Programmierung an bayerischen Grundschulen; A Design-Based Research Approach for introducing Algorithmics and Programming to Bavarian Primary Schools. MedienPädagogik: Zeitschrift für Theorie und Praxis der Medienbildung, 33, 53–75.

Gravemeijer, K. & Cobb, P. (2006). Design research from the learning design perpective. In J. Van den Akker, J. Van den Akker, K. Gravemeijer, S. McKenney & N. Nieveen (Hrsg.), Educational design research (S. 45–85). London, New York: Routledge.

Iskrenovic-Momcilovic, O. (2019). Pair Programming with Scratch. *Education and Information Technologies*, *24*(5), 2943–2952.

Medienberatung NRW. (o. J.). Medienkompetenzrahmen NRW. Zugriff am 6.11.2020. Verfügbar unter: https://medienkompetenzrahmen.nrw/

Prediger, S., Gravemeijer, K. & Confrey, J. (2015). Design research with a focus on learning processes: an overview on achievements and challenges. ZDM, 47(6), 877–891.

Schwätzer, U. (2018). Programmieren in der Grundschule: Erfahrungen | Scratch-Codes | Tipps & Tricks.