DREHER, Ulrike & SCHULER, Stephanie
Landau

Debugging als Facette des Computational Thinking – Zum Einsatz des Bluebot in der Grundschule

In der Grundschule kann die Arbeit mit dem Bodenroboter Bluebot neben der Förderung des räumlichen Vorstellungsvermögens auch einen ersten Zugang zum Programmieren ermöglichen. Das Finden, Beheben und Evaluieren von Fehlern (Debugging) ist eine Teilkomponente des Computational Thinking (CT), die als Querschnittskompetenz über verschiedene Aufgabentypen hinweg im Rahmen einer Lernumgebung gefördert werden kann. Das Debugging ist deshalb besonders zentral, weil das Lernen aus Fehlern und der produktive Umgang mit Fehlern Kinder ihre Lösungsprozesse reflektieren lässt.

Theoretischer Hintergrund und Forschungsstand

CT ist in der Literatur nicht einheitlich definiert (z.B. Angeli et al., 2016; Hsu et al., 2018). Bei Angeli et al. (2016, S. 50ff.) wird ein Modell präsentiert, das aus fünf Teilkomponenten besteht, die auch in anderen Modellen enthalten sind: Das Abstrahieren, der Umgang mit Algorithmen, die Zerlegung gestellter Probleme in Teilprobleme (Dekomposition), die Generalisierung der gefundenen Lösungen und der Umgang mit Fehlern (Debugging).

Der konstruktive Umgang mit eigenen Fehlern ist wichtig, um aus ihnen lernen zu können. Oser et al. (1999) unterscheiden hierbei drei Phasen. In Phase 1 muss der Fehler zunächst erkannt werden. In Phase 2 wird der Fehler verstanden und in Phase 3 wird der Fehler korrigiert. Die Autoren erläutern, dass die Lernenden ihre Fehler durchschauen müssen, um sie in einen „Lernzusammenhang“ (S. 20) stellen zu können. „Fehler machen führt nicht zwingend zu negativem Wissen und nicht zwingend zum Lernen aus Fehlern“ (Oser et al., 1999, S. 20), erst die Reflexion führt zu einem „sinnvollen Fehlermachen“, das zum sicheren Beherrschen von Abläufen beiträgt.

Es konnte schon mehrfach gezeigt werden, dass Lernende im Lernprozess mehr davon profitieren, wenn sie neben der Erklärung, warum eine Bearbeitung richtig ist, auch erklären mussten, warum eine andere Bearbeitung nicht richtig ist (Siegler, 2002; Curry, 2004).

Das Debugging kann vom Ablauf her gut an die drei Phasen nach Oser et al. (1999) angegliedert werden, da die Korrektur von fehlerhaften Befehlsfolgen eines Algorithmus ebenfalls in den drei Schritten (Fehler erkennen, Fehler verstehen und Fehler korrigieren) verläuft. Ob es dadurch auch bereits zu einem reflektierten Umgang mit dem Fehler, im Sinne eines Aufbaus von negativem Wissen, kommen kann, bleibt an dieser Stelle eine offene Frage.

In: P. Ebers, F. Rösken, B. Barzel, A. Büchter, F. Schacht & P. Scherer (Hrsg.),
Beiträge zum Mathematikunterricht 2024.
57. Jahrestagung der Gesellschaft für Didaktik der Mathematik. WTM.
https://doi.org/10.37626/GA9783959872782.0

In der vorliegenden Studie wurde eine Lernumgebung mit Bluebots entwickelt, um die verschiedenen Teilkomponenten des CT zu fördern. In der vorliegenden Analyse sollen die Aufgabenstellungen in Bezug auf das Potential zum Lernen aus Fehlern untersucht werden.

Fragestellungen

Es soll zunächst geklärt werden, welche Lerngelegenheiten zum Debugging durch die verschiedenen Aufgabenstellungen geboten werden. Dabei geht es darum aufzudecken, welche Fehler auftreten, wann sie auftreten und ob sie erkannt und behoben wurden; und welche Rolle im Prozess dabei das digitale Werkzeug Bluebot spielt. Ziel der Analyse ist es langfristig, die Aufgaben so weiterzuentwickeln, dass die Lernenden nicht nur Fehler erkennen und korrigieren, sondern durch eine Reflexion der Fehler prozedurales negatives Wissen (Oser et al., S. 17) aufbauen können.

Methode

Im Fokus der Untersuchung stehen zwei Aufgaben aus einer Lernumgebung, die bei acht Kinderpaaren aus Klasse 3 und 4 eingesetzt wurden (Dreher & Schuler, 2023). Die Lernumgebung wurde gemäß dem Design-Based-Research-Ansatz (Prediger et al., 2012) entwickelt und in mehreren Zyklen überarbeitet. Die Lernumgebung erstreckte sich insgesamt über fünf Sitzungen à 45 Minuten. Die Sitzungen umfassten dabei neben einer Einführung (Kennenlernen des digitalen Werkzeugs) und einem Modul zum unterschiedlichen Darstellen von Fahrtwegen, Module zu den Teilkomponenten Sequenzieren, Debugging und Generalisieren. Es gab also Aufgabenstellungen, die sich explizit mit fehlerhaften Fahrtwegen beschäftigten und Aufgabenstellungen, die andere Teilkomponenten des CT förderten, hierbei also Fehler im Zuge des Bearbeitungsprozesses zufällig auftraten. Die Sitzungen wurden videographiert und mittels einer kodierenden Beobachtung ausgewertet (Pauli, 2012). Der Kodierleitfaden fokussierte dabei auf die verschiedenen Teilaspekte des Debugging, um zu erheben, ob und welche Fehler entstanden sind, ob diese erkannt und behoben wurden. Dabei gibt es Subkategorien wie *Fehler finden* und *Fehler beheben*, aber auch Subkategorien zu den verschiedenen Fehlerarten, die im Bearbeitungsprozess der Aufgaben auftreten können. Hierbei sind beispielsweise die Fehlerarten *Fehler bei Drehung* oder *Auslassen/Vergessen von Befehlen* berücksichtigt. Im Folgenden werden zwei Aufgaben betrachtet (Abb. 1 und 2), bei denen *Fehler bei Drehung* zufällig im Zuge des Bearbeitungsprozesses entstanden sind. Die Kinder waren jeweils aufgefordert, zum abgebildeten Fahrtweg eine Befehlsfolge zu erstellen, die sie anschließend mit dem Bluebot überprüften.

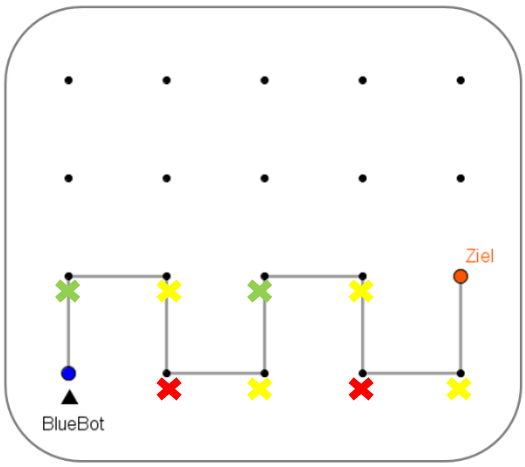

Abb 1. Aufgabenbeispiel Girlande

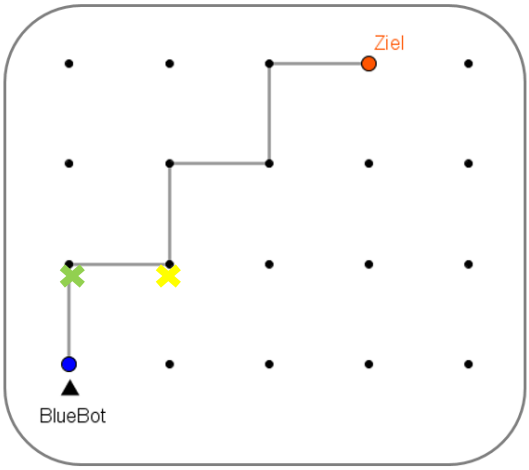

Abb 2. Aufgabenbeispiel Treppe (erstellt mit geogebra.org)

Ergebnisse

Bezüglich der Fehlerarten zeigte sich, dass den Tandems das geradeaus fahren keine Probleme bereitete, Drehungen hingegen mehrfach zum Auftreten von Fehlern beitrugen. Hierbei lassen sich die Drehungen in verschiedene Schwierigkeitsgrade bezüglich der rechts/links-Unterscheidung unterteilen. Wenn der Bluebot nach Norden blickt (grün markiert), bereitet es nur einem geringen Teil der Tandems Probleme die korrekte Drehung in der Befehlsfolge auszuwählen. Bei der Blickrichtung Osten/Westen (gelb markiert) und der Blickrichtung Süden (rot markiert) bereitet es deutlich mehr Tandems Schwierigkeiten die richtige Drehung zuzuordnen. Beim Finden der Fehler übernimmt der Bluebot eine wichtige Funktion, da die erstellte Befehlsfolge jeweils mit dem Bluebot überprüft wurde. Allen Tandems gelang es nach der Eingabe einer fehlerhaften Befehlsfolge, diese mit dem Fahrtweg des Bluebot abschnittweise abzugleichen, sodass die Position des Fehlers ausfindig gemacht werden konnte und es zu einer Korrektur des Fehlers kam.

Bei der Analyse der ersten Aufgabe (Abb. 1: Girlande) zeigten sich bei allen Tandems *Fehler bei der Drehung*, wenn der Bluebot zuvor in Blickrichtung Westen/Osten fährt (vier gelbe Kreuze sind mögliche Fehlerpunkte). Bei Drehungen in der Blickrichtung Norden zeigten sich lediglich bei 2 von 8 Tandems Probleme (zwei grüne Kreuze sind mögliche Fehlerpunkte). Wenn die Blickrichtung gen Süden ausgerichtet war, traten Fehler bei 7 von 8 Tandems auf (zwei rote Kreuze sind mögliche Fehlerpunkte). Im weiteren Verlauf ändert sich die Anforderung durch die veränderte Blickrichtung des Bluebots.

Bei der Analyse der zweiten Aufgabe (Abb. 2: Treppe) zeigte sich, dass nur

noch 4 der 8 Tandems Fehler bei einer Drehung mit Blickrichtung Westen/Osten machten. Interessant ist hierbei, dass die Aufgabe dazu anregte einen sich wiederholenden Baustein für den Fahrtweg zu kreieren. Damit traten Fehler, nur bei der 2. Drehung auf und nicht im späteren Verlauf des Fahrtwegs.

Durch die Lernumgebung wird bei den Kindern auch das räumliche Vorstellungsvermögen geschult, da sie sich für die Drehungen in den Bluebot hineinversetzen müssen. Für das Aufgabendesign kann abgeleitet werden, dass die Fahrtwege, die die Blickrichtung Westen/Osten und Süden enthalten das größte Fehlerpotential aufweisen. Welche Blickrichtung hierbei die größte Herausforderung bietet, soll in weiteren Studien untersucht werden. Außerdem unterstützt der Bluebot das Erkennen und Beheben von Fehlern. Ob auch ein Lernen aus Fehlern und die Ausbildung von negativem Wissen bei den Lernenden erfolgt ist, kann aus dem Vergleich der beiden Aufgabenbearbeitungen (Abb. 1 und 2) implizit abgeleitet werden. Für explizite Folgerungen müsste sich eine Interviewstudie anschließen.

Literatur

Angeli, C., Voogt, J., Fluck, A., Webb, M., Cox, M., Malyn-Smith, J., & Zagami, J. (2016). A k-6 computational thinking curriculum framework: Implications for teacher knowledge. *Educational Technology & Society*, *19*(3), 47–57.

Curry, L. A. (2004). The effects of self-explanations of correct and incorrect solutions on algebra problem-solving performance. In *Proceedings of the Annual Meeting of the Cognitive Science Society* (Vol. 26, No. 26).

Dreher, U., & Schuler, S. (2023). Programmieren mit dem Käferroboter Bluebot: Eine Lernumgebung zur Förderung des Computational Thinking in der Primarstufe. *Mathematik differenziert Nr.2/2023*, S. 40-45.

Hsu, T.-C., Chang, S.-C. & Hung, Y.-T. (2018). How to learn and how to teach computational thinking: Suggestions based on a review of the literature. *Computers & Education*, *126*, 296–310.

Oser, F., Hascher, T., & Spychiger, M. (1999). Lernen aus Fehlern. Zur Psychologie des „negativen" Wissens. (S.11-42) In W. Althof (Hrsg.). *Fehlerwelten*. Wiesbaden: Springer.

Pauli, C. (2012). Kodierende Beobachtung. In H. de Boer & S. Reh (Hrsg.), *Beobachtung in der Schule – Beobachten lernen* (S. 45–64). Wiesbaden: VS Verlag für Sozialwissenschaften.

Prediger, S., Link, M., Hinz, R., Hußmann, S., Thiele, J., & Ralle, B. (2012). Lehr-Lernprozesse initiieren und erforschen – Fachdidaktische Entwicklungsforschung im Dortmunder Modell. *MNU*, *65*(8), 452–457.

Siegler, R. S. (2002). Microgenetic studies of self-explanation. *Microdevelopment: Transition processes in development and learning*, 31-58.