FOLL-E: Teaching First Order Logic to Children*

Simon Vandevelde and Joost Vennekens

KU Leuven, De Nayer Campus, Dept. of Computer Science, Belgium
Leuven.AI — KU Leuven Institute for AI, B-3000 Leuven, Belgium
Flanders Make — DTAI-FET
{s.vandevelde, joost.vennekens}@kuleuven.be

This demonstration presents FOLL-E, our First-Order Logic Learning Environment geared towards children. First-Order Logic (FOL) is an important foundation of many fields, including mathematics, philosophy and computer science. In the domain of AI, the importance can hardly be exaggerated, with many of the first AI systems being based on FOL.

In the past years, many key insights from CS and computational thinking have been successfully taught to children of various ages [1], such as through Scratch [7] and Blockly [3]. These provide useful skills to children by stimulating computational thinking at an early age. In the field of AI, there are also efforts to help children understand machine learning and neural networks [5,6]. However, teaching FOL to children seems curiously understudied, even though it nicely complements Scratch-like tools: Scratch teaches the "how", while logic encourages the "what".

To teach logic to children, we need to overcome a few challenges:

- FOL has a steep learning curve
- it does not "do anything" by itself
- it is not as fun as, e.g., Scratch, because it lacks animations and graphics.

For this, we set out to design our own representation of FOL focussed on a clear structure, encouraging exploration and collaboration, and preventing syntax errors. This resulted in a generic blocks-based notation for FOL, inspired by the success of formalisms such as Scratch. Because of the notation's "pegs-and-slots" design, blocks can only be connected if it results in a syntactically correct sentence, which also prevents type errors. Instead of having to work on a computer through a GUI, we laser cut the blocks out of wood, resulting in a tactile, hands-on experience that lends itself well to experimentation and collaboration. It also gives the notation a puzzle-like feel, making it more intuitive and fun to play around with.

Based on this notation, we created FOLL-E (First-Order Logic Learning Environment). We adapted the generic blocks-based notation to a domain-specific version, centered around the design of simple robots. Children are asked to create a formula using these blocks that distinguishes three "good" robots from

* This research received funding from the Flemish Government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" programme. The authors would also like to thank Kaat De Bock (www.kaatdebock.be) for their robot illustrations.

three "bad" robots. Similar to Scratch, the tool is "always live": a Raspberry Pi continually scans the blocks that have been placed down by detecting the attached ArUco markers [4]. Once these form a complete formula, it is translated into FOL and fed to IDP-Z3 [2], a reasoning engine for FOL, which then verifies whether the good robots are satisfiable, and whether the bad robots are unsatisfiable. These results are shown to the children by placing green check marks or green cross marks next to each robot. As an additional form of feedback, the consequences of the formula are also derived and shown on a big robot in the centre of the screen.

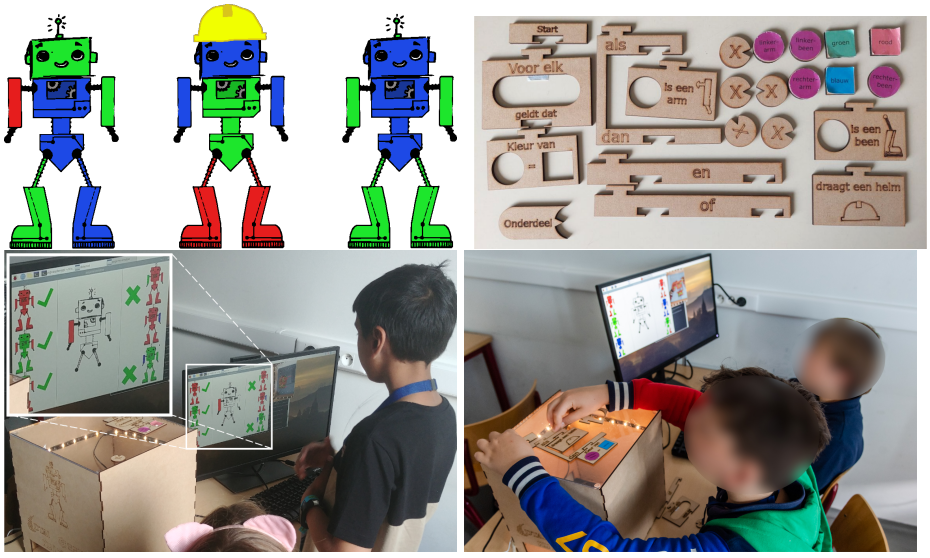

Fig. 1. Top-left: possible robot designs. Top-right: All blocks available in FOLL-E. Bottom: photos taken during a workshop.

In this way, it gives immediate feedback resulting in high interactivity, allowing the children to quickly explore the effects of specific blocks. Moreover, the feedback is purely visual in the form of the check/cross marks and the centre robots, which we feel makes FOLL-E more intuitive and enticing.

System requirements

- Raspberry Pi, PiCam, SD card
- LED strip
- Laser cut blocks with ArUco markers
- A computer monitor and keyboard
- Box with see-through pane on top

During our demo, anyone will be freely able to play around with DALL-E.

References

1. Bell, T., Vahrenhold, J.: CS Unplugged—How Is It Used, and Does It Work? In: Böckenhauer, H.J., Komm, D., Unger, W. (eds.) Adventures Between Lower Bounds and Higher Altitudes: Essays Dedicated to Juraj Hromkovič on the Occasion of His 60th Birthday, pp. 497–521. Springer International Publishing, Cham (2018). https://doi.org/10.1007/978-3-319-98355-4_29
2. Carbonnelle, P., Vandevelde, S., Vennekens, J., Denecker, M.: IDP-Z3: A reasoning engine for FO (.). arXiv preprint arXiv:2202.00343 (2022)
3. Fraser, N.: Ten things we've learned from Blockly. In: 2015 IEEE Blocks and beyond Workshop (Blocks and Beyond). pp. 49–50 (2015). https://doi.org/10.1109/BLOCKS.2015.7369000
4. Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F., Marín-Jiménez, M.: Automatic generation and detection of highly reliable fiducial markers under occlusion. Pattern Recognition **47**(6), 2280–2292 (Jun 2014). https://doi.org/10.1016/j.patcog.2014.01.005
5. Lane, D.: Machine Learning for Kids: A Project-Based Introduction to Artificial Intelligence. No Starch Press (2021)
6. Rauber, M.F., Gresse von Wangenheim, C.: Assessing the Learning of Machine Learning in K-12: A Ten-Year Systematic Mapping. Informatics in Education (2022). https://doi.org/10.15388/infedu.2023.11
7. Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., Kafai, Y.: Scratch: Programming for all. Communications of The Acm **52**(11), 60–67 (Nov 2009). https://doi.org/10.1145/1592761.1592779