Visual Parsing Algorithms for an AR Learning System

Pushpita Saha, Matthew L. Furber, MFA, and Dr. Paul W. Bible

DEPAUW UNIVERSITY Est. 1837

Introduction

This research addresses the challenge of making programming concepts accessible to young children through a digital tangible learning system. Traditional input devices can be difficult for children under five, so we propose using tactile command cards and a machine vision system to create intuitive navigation games.

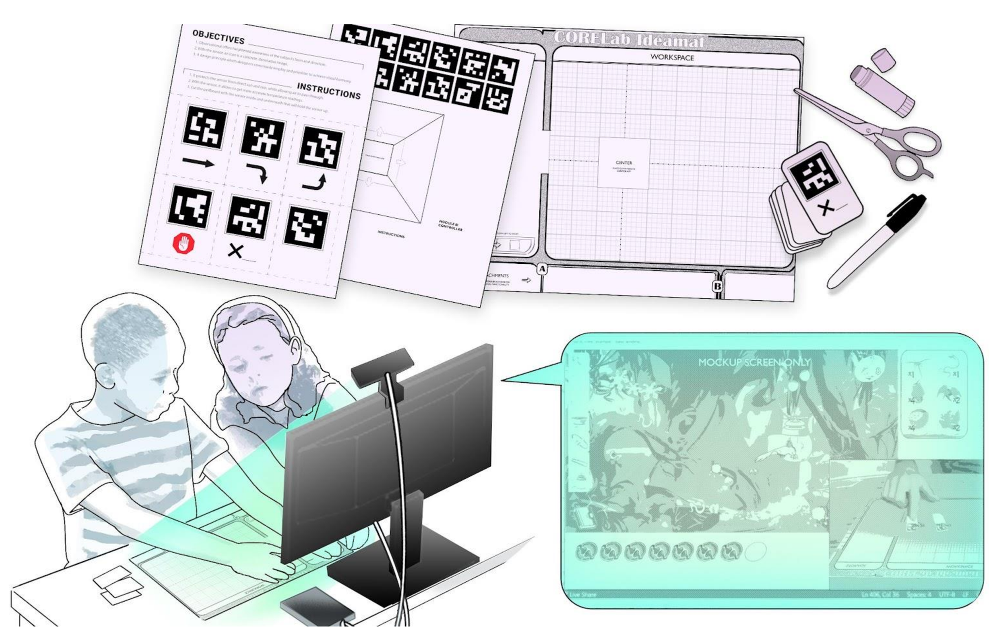

Figure 1 Machine vision system work model showing webcam as a scanner for QR-marked command cards on the table that our algorithms will translate as an executable sequence of a character's navigational plan

This approach simplifies programming for young children by using physical cards on a table, which are processed by algorithms to control an on-screen character, integrating physical interaction with digital feedback to make learning more accessible and engaging. This work explores algorithms to connect the cards in the desired way.

Objective: To develop and evaluate visual parsing algorithms that translate the spatial arrangement of physical cards into commands

Methods

The set of visual parsing algorithms is being developed in C++ with the videogame library Raylib 5.0 [4]. Visual Studio Code (version 1.86) serves as the IDE.

Input Processing → Sorting Algorithm → Graph Connectivity

We are also exploring different weighting schemes on the Minimum Spanning Tree for improved accuracy and flexibility to accommodate other connection uses.

Command Linking Algorithms

The algorithms connect command cards (forward, turn left, etc.) sequentially from left to right, checking if a modifier (e.g. a digit) is placed on top. Our first, hand-engineered algorithm uses a range-based method for better modifier connection when placed perpendicularly (best case). The second algorithm applies Kruskal's Minimum Spanning Tree with a priority queue and union-find data structure, while the third is a custom MST with a refined distance-based weighting function.

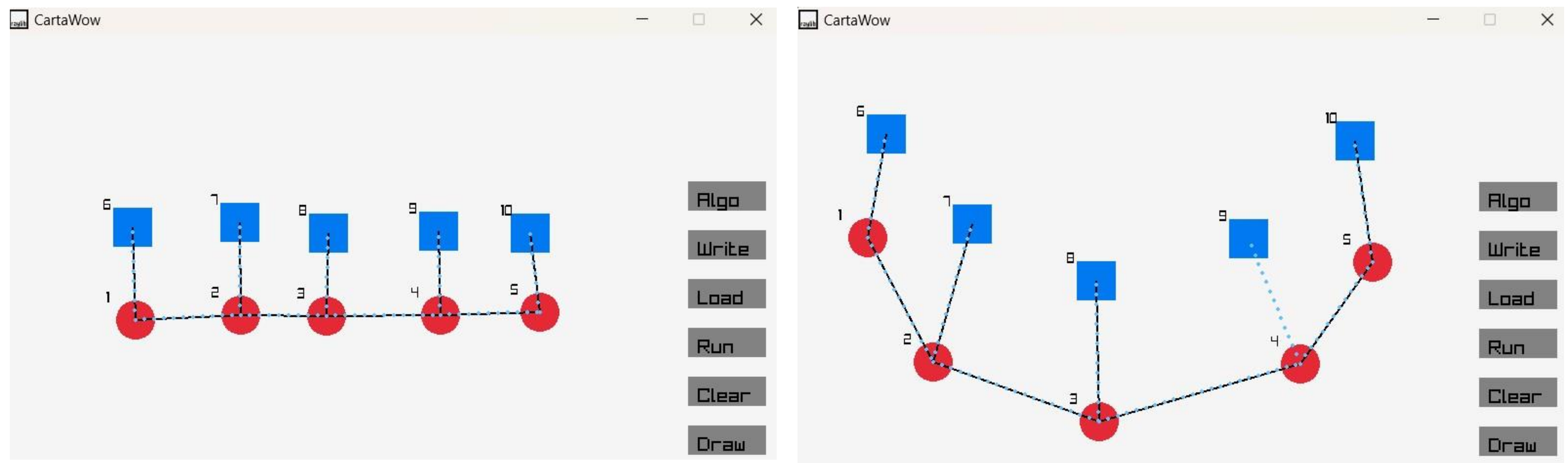

Figure 2. Our hand-engineered algorithm running on two test cases. Red circles represent commands, blue squares represent modifiers, dotted lines are expected edges, and black lines are result edges.

Performance Analysis: Raspberry Pi vs. Windows

We tested our three algorithms on two architectures: Raspberry Pi 4 and Windows 64-bit. The machine vision system is designed to run on Raspberry Pi for cost-effectiveness. Running each algorithm 100 times across five test cases showed that despite the Raspberry Pi's lower processing power, performance metrics are comparable to Windows, with minimal differences in execution times.

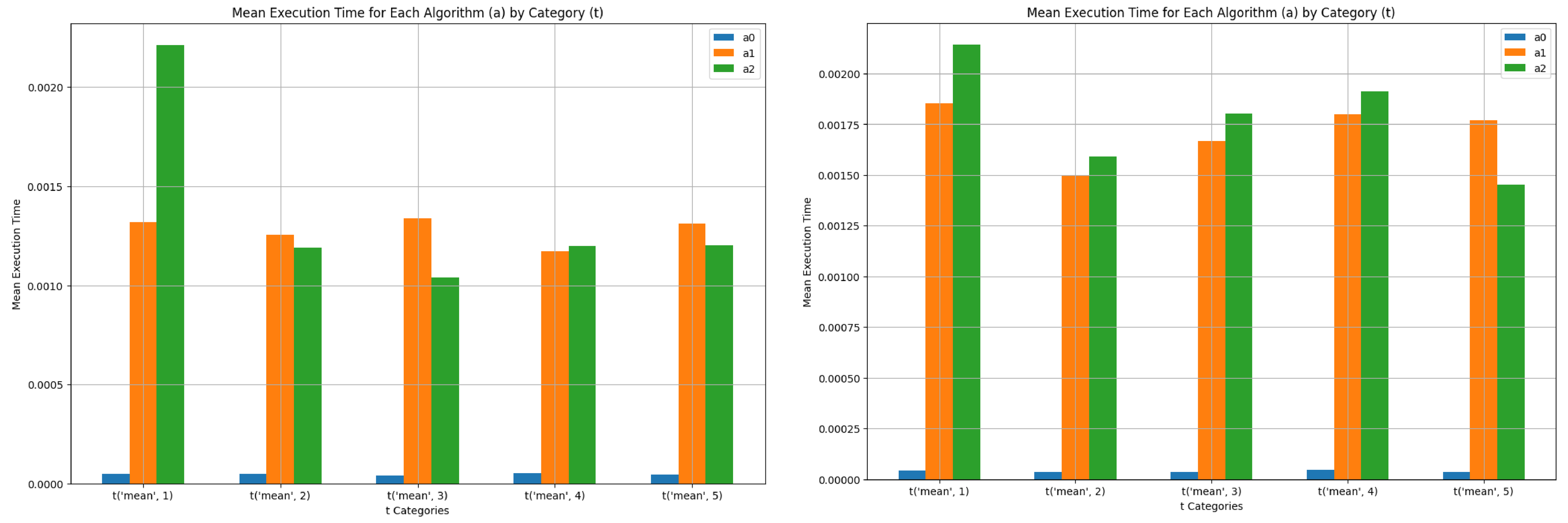

Figure 3. Comparison of mean execution times (y-axis) on Raspberry Pi (left) vs. Windows (right) for 5 test cases (x-axis) per system, with 3 bars per test case representing the 3 algorithms: hand-engineered (a0), Kruskal's MST (a1), and a custom MST with refined distance-based weighting (a2).

The stability in mean and SD suggest that the performance differences between architectures are minimal and do not significantly impact reliability.

Jaccard Index: Higher the Better

The Jaccard Index measures similarity between sets, with 1 indicating a perfect match. We used it to compare the expected vs. result edge set output of three algorithms across five test cases to assess their suitability for our uses.

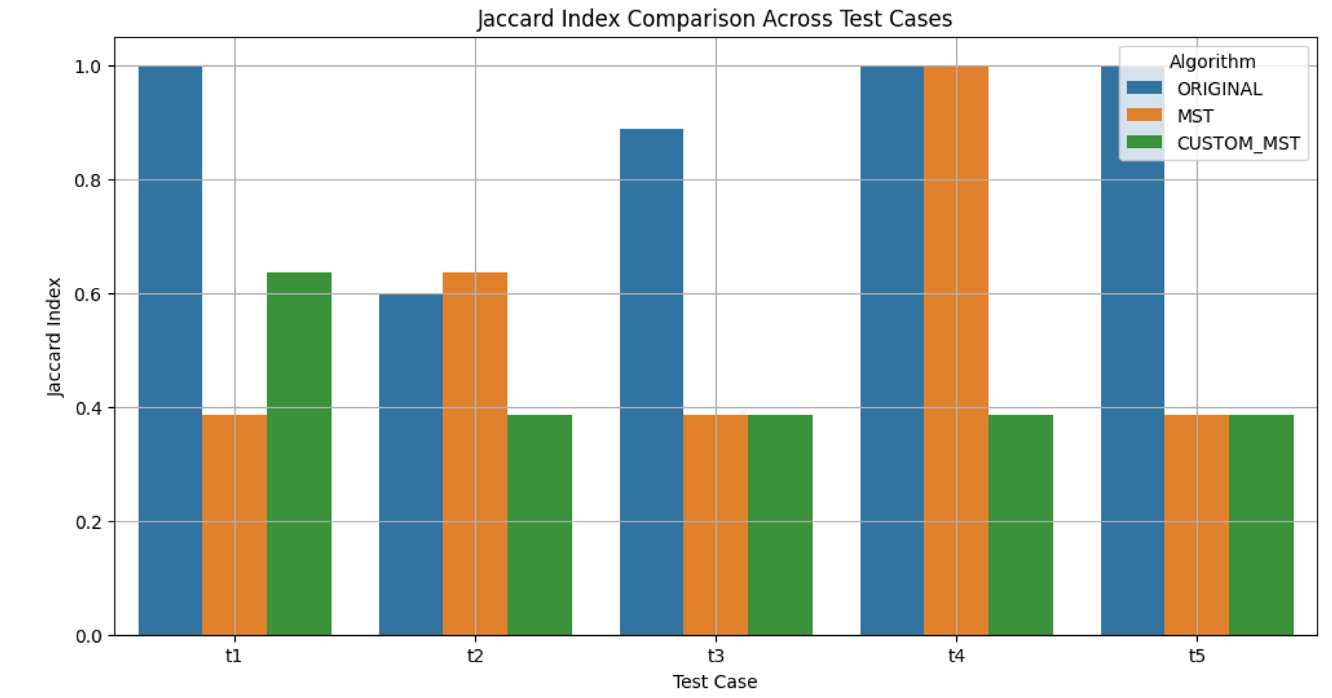

Figure 4. Jaccard Index (y-axis) for 5 test cases (x-axis), with bars representing the 3 algorithms. The hand-engineered original algorithm performed best, the MST had mixed results, and the custom MST showed consistent but lower performance.

Conclusion

Our system offers a cost-effective solution with efficient Raspberry Pi performance, providing an affordable AR platform for learning. The algorithms demonstrate potential in parsing and connecting tactile command cards but are not yet fully effective across all test cases. This AR-driven approach supports interactive programming education for young learners.

Future work will focus on improving error handling and scaling the system for wider educational use.

Acknowledgments

This work was supported by the Information Technology Associates Program and DePauw Student-Faculty Summer Research Grant

Let's Connect!

Questions? Suggestions? Ideas? Please send a message so we can connect!