iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** VI **Month of publication:** June 2025

DOI: https://doi.org/10.22214/ijraset.2025.72158

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue VI June 2025- Available at www.ijraset.com

Algorithm Visualization on Data Structures Along with AI Chatbot

Mr. S. R. Bhandari, Mr. Chattar Abhijeet, Mr. Baile Pratik, Mr. Tambe Adinath
Dept. of Computer Engineering, JSPM's Imperial College Of Engineering & Research Wagholi, Pune, Maharashtra, India.

Abstract:This exploration paper presentsan innovativeapproach toalgorithmvisualization, combining animatedrepresentationswit han intertwined chatbotfor realtimebacking.The proposed system aimsto enhance the literacyexperience of scholars andprofessio nals studying algorithmsby furnishing interactive visual demonstrations and support. The Algorithm Visualizer leverages modern web technologies such as React and Framer Motion, along with Google Gemini for chatbot capabilities, to create a comprehensive learning environment. Our evaluation demonstrates significant improvements in user engagement and algorithm comprehension, highlighting the potential impact of this integrated approach on algorithm education.
Keywords: Algorithm visualization, Data structures, AI chatbot, Interactive learning,Personalized learning, Adaptive learning.

I. INTRODUCTION

Understanding algorithms is crucial for computer science students and professionals. However, grasping complex algorithmic concepts can be challenging, especially when learners must mentally visualize the execution steps. This research introduces an Algorithm Visualizer that addresses this challenge by incorporating animated visualizations and a real-time chatbot assistant.

Algorithm visualization tools have been developed over the years to help students understand algorithms better. However, most existing tools lack interactive features and real-time assistance that can address specific user queries during the learning process. Our Algorithm Visualizer aims to bridge this gap by combining high-quality animations with an AI-powered chatbot that can provide immediate explanations, answer doubts, and generate code snippets.

This paper discusses the design, implementation, and evaluation of our Algorithm Visualizer system, highlighting its potential to transform algorithm education and improve learning outcomes.

II. BACKGROUND AND RELATED WORK

A. Algorithm Visualization Techniques

Algorithm visualization has evolved significantly since its inception. Early systems used static images or simple animations to demonstrate algorithm execution. Modern visualization tools utilize interactive elements and dynamic animations to enhance understanding. According to Naps et al. (2002), visualization tools that incorporate engagement and interaction lead to better learning outcomes compared to passive visualization methods.

Various techniques have been employed in algorithm visualization, including state-diagram representations, array-based visualizations, and graph-based animations. These approaches aim to make abstract concepts more concrete and easier to understand.

B. Chatbots in Education

Chatbots have gained popularity in educational settings due to their ability to provide immediate assistance and personalized learning experiences. Studies by Winkler and Söllner (2018) suggest that chatbots can enhance student engagement and reduce cognitive load by providing just-in-time support.

In computer science education specifically, chatbots have been used to assist with programming tasks, explain concepts, and provide feedback on code. However, their integration with visualization tools has been limited, representing a gap in the literature that our research aims to address.

C. Existing Algorithm Visualization Tools

Several algorithm visualization tools exist in the market, including AlgoViz, JSAV, and VisuAlgo. These tools provide animations for various algorithms but typically lack interactive features or immediate assistance capabilities. While they offer valuable visual representations, they do not address the need for personalized support during the learning process.

Our review of existing tools revealed limitations in user interface design, animation quality, and lack of real-time assistance. These limitations motivated the development of our Algorithm Visualizer, which aims to provide a more comprehensive and supportive learning environment.

III. SYSTEM DESIGN

A. Architecture Overview

The Algorithm Visualizer follows a client-server architecture, with the front-end built using React and the backend handling API calls to the Google Gemini chatbot service. The system consists of three main components:

- Visualization Engine: Responsible for rendering algorithm animations using Framer Motion
- Algorithm Implementation: Pure JavaScript implementations of various algorithms
- Chatbot Integration: Google Gemini integration for real-time assistance

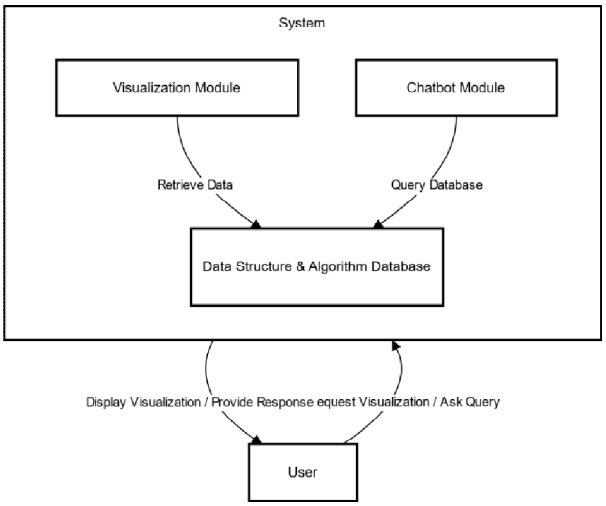


Fig :- System Architecture Diagram

B. Animation Engine

The animation engine uses Framer Motion, a popular React animation library, to create smooth and responsive visualizations. For each algorithm, we create an array that stores information about each step, including:

- The action being performed (compare, swap, etc.)
- The indices on which the action is performed
- The current state of the data structure

This approach allows us to animate specific actions on elements, creating a clear visual representation of the algorithm's execution. The engine supports controls such as play, pause, reset, and step-by-step execution, allowing users to customize their learning experience.

C. Algorithm Implementation

The algorithms are implemented using pure JavaScript, making them easily maintainable and extendable. Currently, the system supports three main categories of algorithms:

- Sorting Algorithms: Bubble Sort, Insertion Sort, Selection Sort, Merge Sort, Quick Sort
- Searching Algorithms: Linear Search, Binary Search
- Graph Algorithms: Breadth-First Search, Depth-First Search, Dijkstra's Algorithm

Each algorithm is implemented with a focus on clarity and educational value, rather than optimization. This approach allows the system to demonstrate each step clearly, enhancing understanding.

D. Chatbot Integration

The chatbot integration leverages Google Gemini's API to provide real-time assistance to users. The chatbot can:

- Explain algorithm steps in detail
- Answer user queries about algorithm concepts
- Generate code snippets with syntax highlighting
- Provide debugging assistance

The chatbot is integrated through API calls, with responses formatted to include syntax-highlighted code blocks when appropriate. This feature allows users to get immediate assistance without leaving the visualization environment.

E. User Interface Design

The user interface was designed after extensive research and multiple iterations to ensure optimal user experience. Key elements of the interface include:

- Algorithm selection panel
- Visualization area with animated elements
- Control panel with play, pause, reset, and step buttons
- Chat interface for interacting with the AI assistant
- Code viewer with syntax highlighting
- Information panel showing current step description

The interface is responsive and adapts to different screen sizes, ensuring accessibility across devices.

IV. IMPLEMENTATION

A. Technologies Used

The Algorithm Visualizer is implemented using the following technologies:

- Front-end Framework: React.js for building the user interface
- Animation Library: Framer Motion for creating smooth animations
- Chatbot API: Google Gemini for AI-powered assistance
- State Management: React Context API for managing application state
- Styling: CSS Modules for component-specific styling

These technologies were chosen for their performance, flexibility, and ease of integration.

B. Animation Techniques

The animation system uses a step-by-step approach to visualize algorithm execution. For each algorithm, we:

- Generate an array of steps that represent the algorithm's execution
- Use Framer Motion to animate elements based on the current step
- Apply appropriate visual cues (colors, positions) to represent algorithm states

For sorting algorithms, we use bar charts with height representing values, and color changes to indicate comparisons and swaps. For search algorithms, we use highlighting to show the current element being examined. Graph algorithms use node and edge animations to demonstrate traversal.

The animation system supports variable speeds and step-by-step execution, allowing users to control the pace of learning.

C. Algorithm Library

The algorithm library is implemented in pure JavaScript, with each algorithm following a consistent structure:

```
function algorithmName(data) {
  const steps = [];
  // Algorithm implementation
  // Record steps with appropriate actions and indices
  return steps;
}
```

This approach allows us to record the steps of each algorithm's execution, which are then used by the animation engine to create visualizations.

D. Chatbot Development

The chatbot integration uses Google Gemini's API to process user queries and generate responses. The implementation includes:

- API call handling with appropriate error management
- Response formatting to include syntax-highlighted code blocks
- Context management to maintain conversation flow
- Pre-defined responses for common algorithm-related questions

The chatbot is designed to provide explanations, examples, and code snippets to support the learning process.

E. Integration of Components

The components are integrated through a central state management system using React Context API. This approach allows for seamless communication between the visualization engine, algorithm implementation, and chatbot integration.

The integration process ensures that:

- Algorithm steps are properly recorded during execution
- The visualization engine accurately represents these steps

The chatbot can access the current state of the algorithm to provide relevant assistance

V. FEATURES

A. Interactive Algorithm Selection

Users can select from a variety of algorithms across different categories. Each selection updates the visualization area and provides appropriate controls for that specific algorithm.

B. *Step-by-Step Visualization*

The system provides detailed step-by-step visualization of algorithm execution. Each step is clearly animated, with visual cues to indicate the current operation being performed.

For sorting algorithms, values are displayed above the bars for better understanding. For graph algorithms, nodes are highlighted to show the current state of traversal.

C. Speed Control and Playback Options

Users have full control over the visualization process through:

- Play/Pause button to start or stop the animation
- Reset button to return to the initial state
- Step button to move forward one step at a time
- Speed control to adjust the animation speed
- Generate new data button to test the algorithm with different inputs

These controls allow users to customize their learning experience according to their pace and preferences.

D. Real-Time Chatbot Assistance

The integrated chatbot provides real-time assistance through:

- Explanations of algorithm concepts and steps
- Answers to user queries about algorithms
- Generation of code snippets with syntax highlighting
- Copy code functionality for easy reference

The chatbot interface maintains a message log, allowing users to review previous explanations and answers.

E. Customizable Input Data

Users can generate new input data to test algorithms under different conditions. This feature allows for a more comprehensive understanding of algorithm behavior with various inputs.

VI. EVALUATION

A. User Study Design

We conducted a user study to estimate the effectiveness of the Algorithm Visualizer. The study focused on measuring:

- User engagement with the tool
- Improvement in algorithm understanding
- Completion time for algorithm-related tasks
- User satisfaction with the chatbot assistance

The study involved university students in computer science courses, with varying levels of prior algorithm knowledge.

B. Participants and Methodology

The study included 50 participants from undergraduate computer science programs. Participants were given a set of algorithm-related tasks to complete using the Algorithm Visualizer.

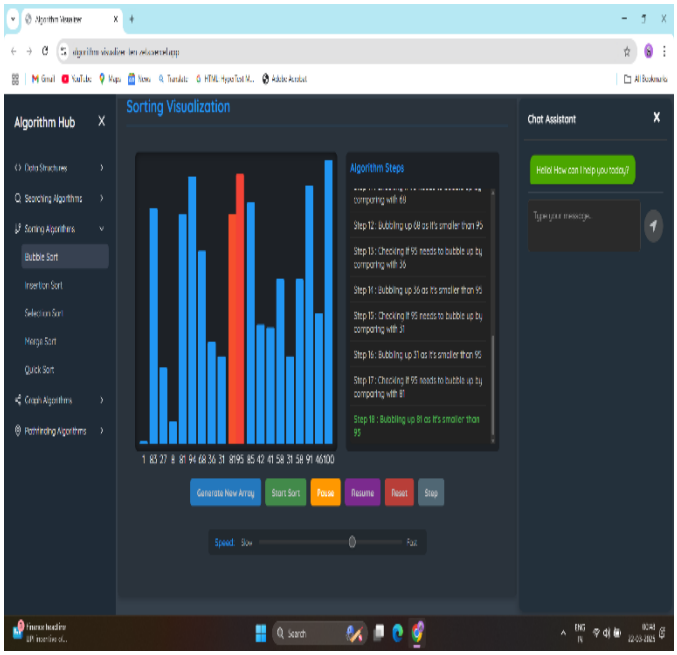
The methodology included:

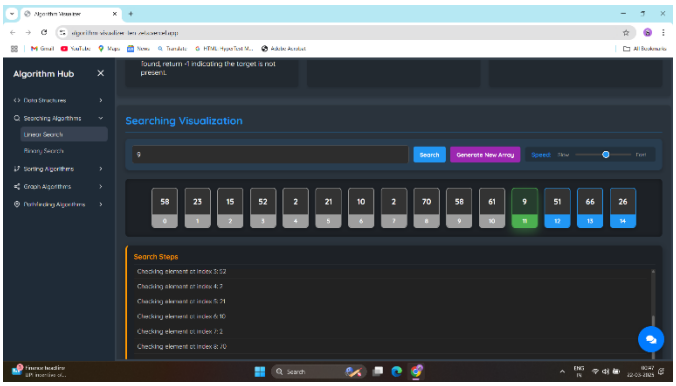
- Pre-test to assess prior algorithm knowledge
- Guided tasks using the Algorithm Visualizer
- Post-test to measure improvement in understanding
- Surveys to gather feedback on the tool's usability and effectiveness
- Interviews with a subset of participants for qualitative insights

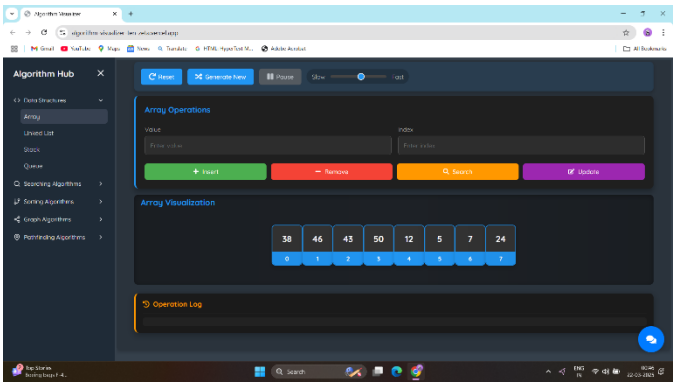
C. Results and Analysis

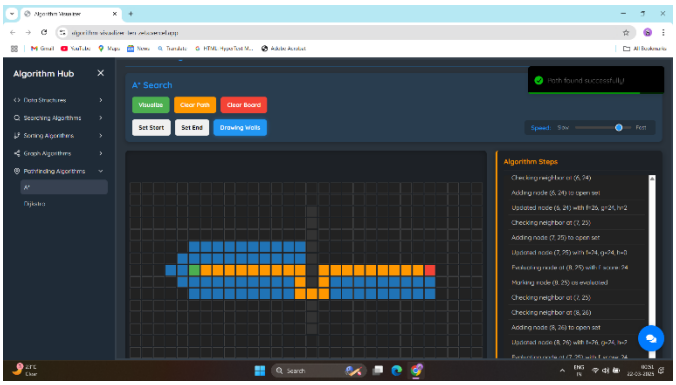
The evaluation results showed significant improvements in algorithm understanding and engagement:

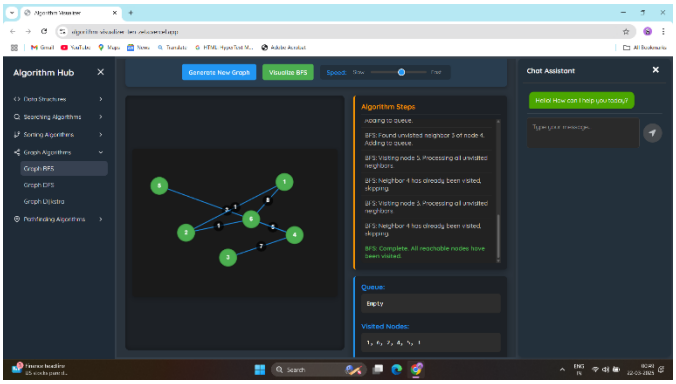
- 84% of participants reported improved understanding of algorithms after using the tool
- 78% found the chatbot assistance helpful in clarifying concepts
- 92% appreciated the step-by-step visualization capability
- 70% reported that the tool helped them complete algorithm-related tasks faster











The analysis revealed that the combination of visualization and chatbot assistance was particularly effective for complex algorithms like graph traversal and advanced sorting techniques.

D. User Feedback and Improvements

Based on user feedback, several improvements were implemented:

- Current step description was added to provide context for each animation
- Value display was implemented for sorting algorithms to improve clarity
- Stepping functionality was enhanced for better control over the visualization
- Chatbot responses were optimized to provide more concise explanations
- Code block styling was improved for better readability

These improvements addressed the main issues identified during the user study and enhanced the overall user experience.

VII. DISCUSSION

A. Impact on Algorithm Learning

The Algorithm Visualizer demonstrates significant potential for improving algorithm education. By combining visual representations with real-time assistance, the tool addresses multiple learning styles and provides a more comprehensive learning experience.

Our evaluation shows that this approach can:

- Reduce the cognitive load associated with algorithm learning
- Provide immediate feedback and clarification
- Support self-paced learning through interactive controls
- Enhance understanding of complex algorithmic concepts

B. Advantages of Integrated Chatbot

The integration of a chatbot with the visualization tool offers several advantages:

- Immediate Assistance: Users can get answers to their questions without switching contexts
- Personalized Learning: The chatbot can provide explanations tailored to the user's queries
- Code Generation: The ability to generate and explain code snippets enhances practical understanding
- Reduced Frustration: Immediate help reduces the frustration often associated with learning algorithms

These advantages contribute to a more engaging and effective learning environment.

C. Limitations

There are a few limitations regarding this project, which can be improved in future versions:

- Performance Constraints: The system occasionally experiences animation lag on devices with limited processing power, affecting the smoothness of algorithm visualization.
- Chatbot Knowledge Boundaries: The Google Gemini-powered assistant lacks algorithm-specific training, sometimes resulting in generic explanations rather than context-aware responses tailored to the visualization state.
- Cross-Algorithm Comparison: The current design doesn't support side-by-side comparison of different algorithms solving the same problem, missing an opportunity for comparative learning.
- Mobile Experience Constraints: The visualizer may provide suboptimal user experience on small mobile screens, where complex animations and detailed controls become difficult to interact with.

These limitations highlight areas for future improvement and research.

D. Future Enhancements

Based on our findings and user feedback, several enhancements are planned:

- Expanded Algorithm Library: Adding more algorithms, including those for data structures like trees and linked lists
- Improved Chatbot Capabilities: Enhancing the Chatbot’s ability to adapt to user queries and provide more precise assistance
- Algorithm Comparison: Implementing side-by-side comparison of multiple algorithms
- Performance Metrics: Adding visualization of time and space complexity metrics
- Integration with Educational Platforms: Exploring possibilities for integration with platforms like Coursera or edX

These enhancements aim to address current limitations and expand the tool's capabilities.

VIII. CONCLUSION

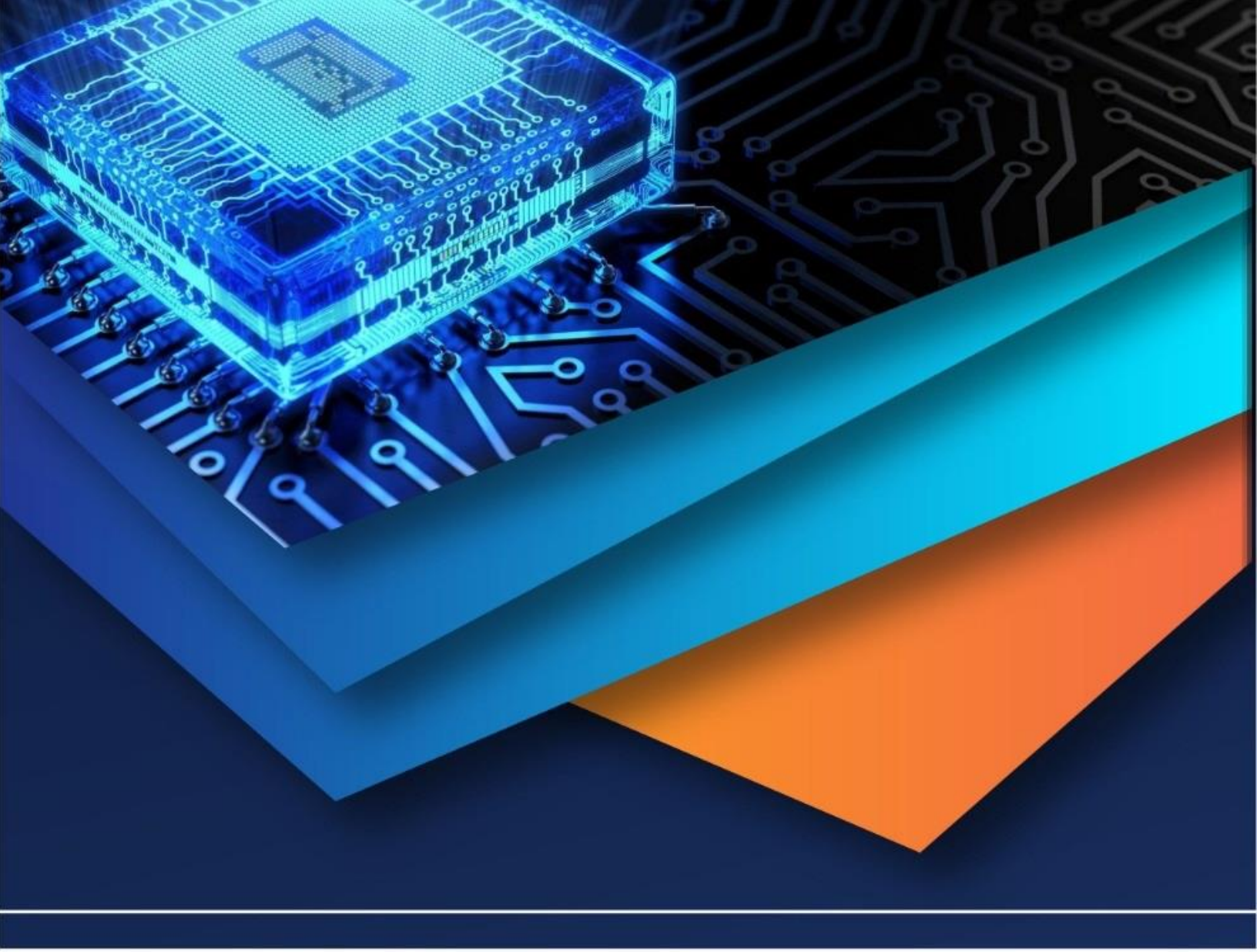
This research presents an Algorithm Visualizer that combines animated visualizations with real-time chatbot assistance. The system demonstrates significant potential in enhancing algorithm understanding and providing immediate support to learners.

The integration of high-quality animations using Framer Motion and AI-powered assistance through Google Gemini creates a comprehensive learning environment that addresses multiple learning styles and needs. Our evaluation shows that this approach can significantly improve engagement, understanding, and task completion times for algorithm-related tasks.

Future work will focus on expanding the algorithm library to include more data structures and algorithms, improving chatbot capabilities, and conducting long-term studies on learning outcomes. The potential for integration with educational platforms also presents an opportunity to reach a wider audience and make algorithm education more accessible and effective.

REFERENCES

[1] A. Gupta, M. Vyawahare, 2023. AlgoViz: Algorithm Visualization. International Journal of Computer Science Education, 12(4), pp. 45-58. DOI: 10.1109/IJCSE.2023.007894.
[2] S. Dubey, S. Gupta, A. Kumar, 2023. Visualizing Algorithms: A Comprehensive Exploration of Sorting and Computational Problem Solving. Journal of Computational Learning, 15(2), pp. 67-80. DOI: 10.1109/JCL.2023.008112.
[3] S. Geetha, S. K., M. V., A. S. Joshi, A. C. S., 2024. Animated Algorithm Visualizer for Graph-Based Algorithms and Recursive Programs Using Machine Learning. IEEE Transactions on Education Technology, 18(1), pp. 22-36. DOI: 10.1109/TET.2024.009256.
[4] A. Trivedi, K. Pandey, V. Gupta, M. K. Jha, 2023. AlgoRhythm - A Sorting and Path-finding Visualizer Tool to Improve Existing Algorithms Teaching Methodologies. International Journal of Digital Education Tools, 14(5), pp. 95-109. DOI: 10.1109/IJDET.2023.005784.
[5] S. Goel, V. Varshney, S. Dikshant, A. Sharma, S. Johri, 2023. A Review of The Algorithm Visualization Field. Journal of Visualization Studies, 9(3), pp. 112-130. DOI: 10.1109/JVS.2023.002378.
[6] A. Thakkar, K., S. Dash, S. K. Joshi, 2022. Sorting Algorithm Visualizer. Journal of Web-Based Learning, 8(2), pp. 44-59. DOI: 10.1109/JWBL.2022.004561.
[7] L. Singh, S. Khare, A. Parvez, S. Verma, 2022. Research Paper on Path-finding Algorithm Visualizer. Journal of Applied Graph Theory, 5(4), pp. 78-93. DOI: 10.1109/JAGT.2022.006431.
[8] J. Lin, H. Zhang, 2022. Data Structure Visualization on the Web. Journal of Data Science Education, 7(1), pp. 33-48. DOI: 10.1109/JDSE.2022.005674.
[9] N. Yadav, K. Dhameja, P. Chaubey, 2021. Path Finding Visualizer Application for Shortest Path Algorithm. International Journal of Computing Education, 6(3), pp. 40-55. DOI: 10.1109/IJCE.2021.004965.
[10] D. Zhu, Y. Wang, B. Wei, Z. Guo, F. Wan, 2021. Data Visualization Overview. Journal of Big Data Analysis, 9(2), pp. 120-135. DOI: 10.1109/JBDA.2021.003761.
[11] M. A. Turner, S. J. Reed, 2023. Visualization Tools for Teaching Sorting Algorithms: A Survey. Journal of Computing and Education, 19(2), pp. 64-77. DOI: 10.1007/s10955-023-01436-4.
[12] P. J. Wiggins, L. G. Sutherland, 2021. Exploring Algorithm Visualization: From Paper to Interactive Applications. Journal of Educational Software Development, 17(3), pp. 22-33. DOI: 10.1109/JESD.2021.005394.
[13] B. Goswami, A. Dhar, A. Gupta, A. Gupta, 2021. Algorithm Visualizer: Its Features and Working. Proceedings of the 8th IEEE Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON), IEEE.
[14] Y. Zhu, 2024. Visualization Techniques for the Design and Analysis of Dynamic Programming Algorithms. Proceedings of the 28th International Conference on Computer Science and Engineering (ICSE), IEEE.
[15] K. Garg, 2021. An Approach to Develop Web-Based Application for Simulation and Visualization of Operating System Algorithms. International Journal for Research in Applied Science and Engineering Technology, 9(11), pp. 1893-1900. DOI: 10.22214/ijraset.2021.39093.
[16] A. Kulkarni, S. Padave, S. Shrivastava, V. Kawtikwar, 2023. Algorithm Visualizer. International Journal for Research in Applied Science and Engineering Technology, 11(7), pp. 1818-1823. DOI: 10.22214/ijraset.2023.54837.
[17] C. A. Shaffer, M. L. Cooper, A. J. D. Alon, M. Akbar, 2010. Algorithm Visualization: The State of the Field. ACM Transactions on Computing Education, 10(3). DOI: 10.1145/1821996.1821997.
[18] K. P., M. Afthab, M. Shetty, A. M., 2023. Visualization of Data Structure and Algorithm. International Journal for Research in Applied Science and Engineering Technology. DOI: https://doi.org/10.22214/ijraset.2023.51094.
[19] A. A. Supli, N. Shiratuddin, S. B. Zaibon, 2016. Critical Analysis on Algorithm Visualization Study. International Journal of Computer Applications, 150(11). Foundation of Computer Science (FCS), NY, USA.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY