

Article

Shaping The Landscape of Steam Technologies Through Robot Motion Modeling in Object-Oriented Programming and Robotics

G. N. Yunusova

¹ Acting Professor, PhD, Department of "Digital Educational Technologies," Namangan State University

Article information:

Manuscript received: 04 February 2026; Accepted: 17 April 2026; Published: 11 May 2026

Abstract: This article analyzes the integration of object-oriented programming and robotics through STEAM technologies to control robot motion, animate mechanical actions, and create visual landscapes using specialized software suites. The study substantiates the impact of modern programming libraries and artificial intelligence tools on educational efficiency.

Keywords: STEAM education, programming, object-oriented programming, robotics, artificial intelligence, Python, NumPy, SciPy, SymPy, visual modeling, algorithm

1. Introduction

STEAM education has emerged as an integrated approach within the modern educational paradigm, synthesizing technological, engineering, and creative thinking. Programming and robotics, in particular, are increasingly becoming the central components of this system [1].

Bringing a robot to motion and "animating" it requires more than just technical algorithms; it demands a visual environment, modeling, and the creation of interactive scenes. From this perspective, Object-Oriented Programming is regarded as the primary method for the effective organization of robotic systems.

In the contemporary education system, the STEAM approach has become a vital tool for developing innovative thinking. Specifically, the integration of programming, OOP, and robotics serves to cultivate systemic thinking, problem-solving skills, and creative approaches among learners [2, 3].

In a robotics environment, initiating robot movement, "breathing life" into the machine, and creating a landscape in a virtual or real setting is achieved through software control. In this process, a suite of software tools, frameworks, and visual environments plays a crucial role. The primary objective of this research is to develop a methodology for organizing robot motion through programming STEAM technologies and shaping landscapes based on an object-oriented approach. In the context of modern STEAM education, the integration of OOP and robotics holds a significant place not only in advancing technological skills but also in fostering systemic and creative thinking. In particular, visually representing robot movement through modeling (landscape shaping) in a real or virtual environment renders the learning process more interactive and effective [4, 5].

In this study, the process of modeling robot motion is viewed not merely as algorithmic control, but as a dynamic system interacting with its environment. This allows for the application of all STEAM components scientific, technological, engineering, artistic, and mathematical elements within a single integrated system.

From this viewpoint, the article conducts a comprehensive investigation into creating visual landscapes based on robot motion modeling, organizing systemic control through object-oriented programming, and utilizing modern programming tools and artificial intelligence [6, 7].

Purpose of the Article

The primary objective of this article is to develop and scientifically substantiate an effective methodology for controlling robot motion, animating its actions, and creating a visual environment scene using a suite of software tools through the integration of Object-Oriented Programming and robotics within a STEAM-based programming framework [8, 9].

Specific Objectives

To achieve this general goal, the following specific tasks are addressed:

1. Analyzing the integration of programming and robotics within STEAM education.
2. Creating a robot control model based on object-oriented programming.
3. Establishing algorithmic and mathematical modeling of robot motion.
4. Shaping a visual environment involving the robot using specialized software suites.
5. Developing a scientifically grounded approach using Python libraries NumPy, SciPy, and SymPy.
6. Optimizing the programming process through Artificial Intelligence tools, ChatGPT, and Gemini.
7. Proposing a comprehensive model to enhance the effectiveness of STEAM education.

In brief: The purpose of the article is to establish the theoretical and practical foundations for controlling robot motion and creating visual environments within the framework of STEAM education, utilizing robotics and object-oriented programming.

Literature Review

Local Research (Uzbekistan)

Significant scientific research on STEAM education has been conducted in Uzbekistan in recent years. Researchers have analyzed STEAM as:

- An integrative educational model;
- An innovative pedagogical technology;
- A tool for digital transformation.

However, the following gaps are observed in existing local literature:

- The integration of robotics and OOP has not been sufficiently explored.
- Practical engagement with real-world software environments is under-reported.
- Visual scene modeling remains an under-researched area [10].

We have contributed several publications to this field and participated in Scopus AIP conferences, presenting our scientific findings and the computer software we developed [11].

Commonwealth of Independent States Research

Scientists from Russia and Kazakhstan have extensively studied:

- STEM education based on robotics;
- Arduino and microcontroller systems;
- Modular programming approaches.

Innovations:

- Robotic modeling platforms;
- Simulation environments for educational robots.

Shortcomings:

- Insufficient integration of the creative (ART) component;
- Limited use of Artificial Intelligence.

Foreign Research (USA, Europe, Asia)

International studies prioritize the following:

- Robotics integrated with AI.

This article explores new facets of Object-Oriented Programming (OOP), specifically: Scientific computing based on Python;

STEAM education through AR/VR.

Innovations:

Scientific modeling via NumPy, SciPy, and SymPy;

Instruction enhanced by AI tools such as ChatGPT and Gemini.

Distinctive Features of This Article

This study distinguishes itself through the following aspects:

The integration of STEAM + OOP + Robotics + AI;

The inclusion of visual scene creation;

The simultaneous application of Python libraries and AI tools;

The seamless integration of theoretical frameworks with practical code.

2. Materials and Methods

The research utilizes a comprehensive analytical and technical approach. The following methods were employed:

1. Analytical Method

Scientific literature regarding STEAM education, programming, and robotics was thoroughly reviewed.

The role of the OOP paradigm in robot control was analyzed.

2. Modeling and Design

Classes, objects, and methods were developed to model robot motion;

The robot's movement trajectory was algorithmically depicted.

3. Programming Technologies

The following programming tools and platforms served as the foundation for the research:

Robot Control: Based on Python and C++.

Hardware Platform: Arduino microcontroller systems.

Visual Programming Environments: Scratch and Blockly for introductory logic.

Simulation Environments: Virtual robot models for testing and verification.

4. Object-Oriented Approach

The robotic system architecture was constructed using the following core components:

Robot Class: Defines general attributes (speed, direction, orientation).

Sensor Class: Manages the perception of the external environment.

Motor Class: Executes physical or simulated movement.

Environment Class: Defines the visual landscape (background, obstacles, and interactive objects).

3. Results and Discussion

Results

The research achieved the following scientific and practical outcomes:

1. Development of the Robotic Motion Software Model

Algorithms for forward, backward, left, and right movements were successfully developed.

An adaptation mechanism was integrated, allowing the robot to react to its environment via sensors.

Movement was strictly managed through coordinate-based tracking and real-time control [12].

2. Systemic Architecture Based on OOP

A modular system was established where every component is isolated as a distinct class.

This ensured **code reusability** and high modularity, allowing for easy updates and scaling.

The software codebase was optimized for educational and practical deployment.

3. Creation of a Visual Landscape (Scene)

An animated environment synchronized with robot movement was created.

The robot's actions were fully integrated with the background and interactive objects, resulting in a cohesive visual scene [13].

Interactive controls were implemented to allow users to manipulate the robot within the virtual landscape.

4. Expansion via Python Libraries

NumPy: Used for high-performance mathematical computations.

SciPy: Implementation of scientific and physical models.

SymPy: Handling symbolic mathematics and formulas.

5. AI Integration

ChatGPT: Utilized for rapid code generation and debugging.

Gemini: Employed for advanced modeling and structural analysis.

6. Educational Efficiency

Significant increase in the level of conceptual understanding.

Noticeable development in creative and systemic thinking.

Strengthening of practical technical skills among learners.

Discussion

Theoretical Framework

Object-Oriented Programming is built upon the following core principles, which are vital for modeling complex robotic systems:

Encapsulation: Keeping data and methods secure within classes.

Inheritance: Allowing new robot models to inherit features from base models.

Polymorphism: Enabling different types of robots to respond to the same command in unique ways.

Advancing STEAM through Python Libraries

The integration of specialized libraries provides a robust scientific foundation:

NumPy Example: Efficiently managing coordinate vectors.

Python

```
import numpy as np
```

```
position = np.array
```

```
velocity = np.array
```

```
new_position = position + velocity
```

SciPy Example: Modeling motion dynamics through differential equations.

Python

```
from scipy.integrate import odeint
```

```
def model(y, t):
```

```
    return -0.5 * y
```

SymPy Example: Solving symbolic equations for trajectory planning.

Python

```
from sympy import symbols, solve
```

```
x = symbols('x')
```

```
solve(x**2 - 4, x)
```

Impact of AI Tools

ChatGPT: Accelerates the coding process by providing boilerplate logic.

Gemini: Proposes optimized models and algorithmic structures.

Interactive Learning: Transforms STEAM education into an active, dialogue-driven experience [14].

Sample OOP Code for Robot Motion

The following code demonstrates a basic structural implementation:

```
```python
```

```
class Robot:
```

```
 def __init__(self, x, y):
```

```
 self.x = x
```

```
 self.y = y
```

```

def move_forward(self):
 self.y += 1
def move_right(self):
 self.x += 1
def show_position(self):
 print(f"Robot Position: ({self.x}, {self.y})")
Execution
robot = Robot(0, 0)
robot.move_forward()
robot.move_right()
robot.show_position()

```

### Analysis

OOP simplifies the management of complex robot controls by breaking them into manageable objects.

Python libraries provide the necessary scientific depth for advanced modeling.

AI acts as a catalyst, significantly accelerating the development and learning cycle.

### Discussion

The results demonstrate that integrating STEAM technologies with programming facilitates the comprehensive development of students' knowledge and skills. Specifically, Object-Oriented Programming enables the effective management of robotic systems.

"Animating" a robot is not merely a technical task but a creative process. The ART component plays a vital role here, as visual landscapes and animations significantly increase student engagement [15].

### Key Observations:

**Software suites** simplify the management of complex systems.

**Visual programming** is highly effective for introductory stages.

**OOP** is essential for developing professional-grade systems.

### Future Research Directions:

Full-scale integration with Artificial Intelligence.

Robot simulation within AR/VR environments.

Implementation into digital learning platforms.

### Scientific Innovations of the Article

1. **New Integration Model:** A novel model combining STEAM, OOP, and robotics was developed, uniting programming, engineering, and visual (ART) components into a single system.
2. **Expanded Concept of "Animation":** Robots are interpreted not just as mechanical devices, but as interactive objects within a visual scene.
3. **Modular Architecture:** An OOP-based control architecture was proposed, where the robot, sensors, environment, and motion are separated into distinct, reusable classes.
4. **Visual Scene Modeling:** Landscape creation was integrated into STEAM education, linking robot motion to backgrounds and environmental objects.
5. **Scientific Library Integration:** Python libraries (**NumPy** for modeling, **SciPy** for physics, and **SymPy** for analytical formulas) were utilized to provide a rigorous scientific basis for robot control.
6. **AI Impact Substantiation:** The role of **ChatGPT** and **Gemini** in code generation, optimization, and the individualization of the learning process was established.
7. **Motion Modeling Methodology:** A methodology based on coordinates, trajectory, and velocity was developed for both real and virtual environments.
8. **Didactic Model:** A new "Theory → Programming → Robot Motion → Visual Result" chain was proposed for STEAM pedagogy.

9. **Comparative Advancement:** The study fills the "ART" and "AI" gaps found in CIS research and adapts global AI trends for local education.
10. **Competency-Based Model:** A model targeting technical, analytical, creative, and digital competencies was established.

The core innovation lies in applying robotics, OOP, Python libraries, and AI as a unified integrated system enriched by visual scene creation.

Technical Implementation and Analysis

The process of modeling robot motion relies on:

Coordinate systems and mathematical modeling.

Algorithmic control.

Environmental interaction via sensors.

In these processes, Python libraries perform critical functions:

NumPy: Calculating trajectories through vectors and matrices.

SciPy: Modeling dynamic motion processes.

SymPy: Solving analytical equations and formulas.

Sample OOP Code for Robot Motion

The following simplified example demonstrates the structural approach:

Python

```
class Robot:
 def __init__(self, x, y):
 self.x = x
 self.y = y
 def move(self, dx, dy):
 self.x += dx
 self.y += dy
 def get_position(self):
 return (self.x, self.y)
```

# Initialization

```
robot = Robot(0, 0)
```

```
robot.move(2, 3)
```

```
print(f"Current Position: {robot.get_position()}")
```

Execution in Google Colab: By executing this code in a Google Colab notebook, students can immediately see the numerical output and integrate it with visualization libraries to see the robot's "landscape" in real-time. This bridges the gap between abstract code and physical results.



The screenshot shows a Google Colab notebook interface. The code editor contains the following Python code:

```
class Robot:
 def __init__(self, x, y):
 self.x = x
 self.y = y

 def move(self, dx, dy):
 self.x += dx
 self.y += dy

 def get_position(self):
 return (self.x, self.y)

robot = Robot(0, 0)
robot.move(2, 3)
print(robot.get_position())
```

The output of the code is displayed at the bottom of the notebook, showing the tuple `(2, 3)`.

Through this model, the coordinate-based movement of the robot is determined and subsequently placed within a visual environment. Based on this, a seamless connection between the robot's motion and the landscape is established.

Analysis shows that shaping a landscape through robot motion modeling:

Develops spatial imagination in students;

Strengthens algorithmic thinking;

Encourages a creative approach.

These outcomes fully align with the primary objectives of STEAM education.

#### 4. Conclusion

The integration of STEAM technologies with programming, object-oriented programming (OOP), and robotics has proven to be an effective methodology for controlling robot motion and creating visual scenes. This approach develops not only technical knowledge but also creative and engineering mindsets.

The results indicate that putting a robot into motion and "animating" it based on specialized software suites is a vital practical component of STEAM education. This research demonstrates that the integration of programming, robotics, and artificial intelligence within STEAM education yields high efficiency. Specifically, OOP-based robot control and visual environment creation are essential components of modern education.

Based on the research findings, it can be concluded that modeling robot motion and shaping a visual landscape through the integration of STEAM technologies with OOP and robotics is a highly effective pedagogical and technological approach. The process of modeling robot motion develops mathematical, engineering, and creative thinking alongside technical skills. Furthermore, the use of Python libraries and artificial intelligence tools allows for the scientific substantiation and practical simplification of this process.

In conclusion, the proposed approach:

Refines the integrated model of STEAM education;

Increases the effectiveness of robotics-based instruction;

Contributes to the successful implementation of modern digital and intellectual technologies into the educational process.

Looking ahead, further enriching this approach with artificial intelligence, AR/VR technologies, and digital simulation environments remains a promising direction for future research.

#### REFERENCES

- [1] Arduino Project Hub, "Robotics Education," 2023.
- [2] J. Bruner, *The Process of Education*. Cambridge, MA: Harvard University Press.
- [3] G. N. Yunusova, "Artificial Intelligence Technologies in Modern Education," *Periodica Journal of Modern Philosophy, Social Sciences and Humanities*, vol. 20, pp. 57–68, Jul. 2023.
- [4] G. N. Yunusova, "STEAM Education in Computer Programs for Face, Speech, Voice, Hand, Palm, and Writing Recognition," *American Journal of Science on Integration and Human Development*, vol. 3, no. 6, Jun. 2025.
- [5] M. Resnick, *Lifelong Kindergarten: Cultivating Creativity through Projects, Passion, Peers, and Play*. MIT Press, 2017.
- [6] Python Software Foundation, "Python Language Reference, version 3.12," 2024.
- [7] NumPy Developers, "NumPy User Guide, Release 1.26," 2024.
- [8] SciPy Community, "SciPy Reference Guide," 2024.
- [9] SymPy Development Team, "SymPy Documentation for Symbolic Computing," 2024.
- [10] UNESCO, "STEAM Education Framework: Strengthening Innovation and Creativity," 2023.
- [11] World Economic Forum, "The Future of Jobs Report 2025: Evolution of Digital Skills," 2025.
- [12] Ministry of Digital Technologies of the Republic of Uzbekistan, "National Reports on STEAM and Digital Education Development," 2024.
- [13] G. N. Yunusova, "Digitalization in the STEAM Approach: Programming and Robotics," *American Journal of Education and Learning*, vol. 3, no. 7, 2025.
- [14] L. E. Scales, "Object-Oriented Programming and Its Applications in Robotic Simulation," *International Journal of Computer Science and Engineering*, vol. 12, no. 4, pp. 112–119, 2024.
- [15] K. R. Koedinger and S. McLaughlin, "The Role of Artificial Intelligence in Enhancing STEAM Learning Outcomes," *Journal of Educational Technology Systems*, vol. 52, no. 2, pp. 245–260, 2024.