The Analytical Geometry of the Plane

Course of Methods in Computer Science Education: Analysis

Manganaro Clizia Giorgia
id.2017897

A.Y. 2022/2023

Introduction

The main objective of this report is to provide an overview of the creation process for a Learning Unit named "The Analytical Geometry of the Plane" using studio.code.org. The platform allows students to learn coding using visual programming languages like Blockly or textual programming languages like JavaScript.

The primary aim is to enable students to gain a more comprehensive understanding of the equations in the plane of certain geometric figures, particularly the line, the circle, the ellipse, and the parabola. So, the students through the LU will have the opportunity to understand the equations of the geometric form and how the change of the coefficients influences the graphical presentation.

The platform allows for easy and intuitive creation of web applications. Therefore, at the end of the LU, students will create a web application at various levels where the user can input equations of different geometric shapes through a user interface. The program then verifies if the entered equations are correct for the current level and draws the corresponding shape on the Cartesian plane. The interface is composed of sections that guide the user through the different levels, each requiring the input of a specific type of geometric equation with its respective data. Additionally, there is a section where the user will input the values of coefficients (a, b, c) to correctly generate the equation of the desired function.

Prerequisites and Class

The learning unit is intended for a fourth-year class at a scientific high school during the math lectures. This choice not only aligns with the Italian ministerial programs but also addresses the challenge posed by the programming language used by studio.code.org, which is JavaScript.

Since JavaScript is a complex language, it is deemed necessary for students to have a solid foundation in programming. Through the support and guidance provided by studio.code.org's block-based programming interface, students are aided in overcoming the complexities of JavaScript. This approach ensures that students receive the necessary assistance while still being exposed to the fundamentals of programming.

It's possible to distinguish various prerequisites that the student must possess before proceeding with LU development that can be divided into two categories: Computer Science and Mathematics.

A solid foundation in **Computer Science prerequisites**, including programming language syntax, variables and data types, control structures like conditional statements and loops, functions, and event handling, is fundamental for students in order to proceed with the creation of the learning Unit. Finally, the last prerequisite required is that students possess the ability to effectively manage logical errors.

As for the **Mathematical Prerequisites**, students are expected to have a comprehensive understanding of the semantic meaning of formulas and equations, encompassing both their implicit and explicit forms. This understanding should extend to fundamental geometric shapes such as lines, circles, ellipses, and parabolas. Specifically, students should be proficient in interpreting the geometric significance of these mathematical entities.

Moreover, students need to comprehend the contextual relevance of formulas and equations. This involves recognizing and interpreting the semantic meaning of various symbols and terms embedded in the formulas. Additionally, they should possess the capability to manipulate both implicit and explicit forms of equations, showcasing versatility in algebraic operations.

Furthermore, proficiency in deriving specific formulas for geometric curves, such as those defining circles, ellipses, and parabolas, is essential. This involves a deep understanding of the underlying principles and properties associated with each geometric shape.

Learning Objectives and Motivations

Through this Learning Unit, students will learn how to work on studio.code.org and deepen their programming skills. They will understand how to interact with the graphical interface and manage events within the platform. This is crucial in developing students' problem-solving abilities and computational thinking.

From a mathematical perspective, students will have the opportunity to solidify their understanding of formulas for geometric figures in the plane. They will learn how these formulas can be graphically represented. This will allow them to gain a deeper understanding of mathematical concepts and apply them in practical contexts. This approach aims to ensure that students not only acquire a conceptual understanding of the topics covered but are also able to apply such knowledge in practical contexts. The guidelines that will be given to the student are as follows:

> *During the year, you studied plane geometry. Now, you have to create a web application using the studio.code.org platform to represent the figures you've learned. Specifically, you have to identify their equations, both in implicit and explicit forms, and demonstrate how the equation changes with variations in their coefficients (a, b, c). Finally, generate a graph to visualize them.*

The students can use the regex generator site in order to write and handle the equation inserted by the user.

Some formulas given:

Line

Implicit form of a line:

$$ax + by + c = 0$$

Explicit form of a line:

$$y = mx + q$$

Circle

Implicit form of a circle:

$$(x - h)^2 + (y - k)^2 = r^2$$

Explicit form of a circle:

$$y = \pm\sqrt{r^2 - (x - h)^2} + k$$

Ellipse

Implicit form of an ellipse:

$$\frac{(x - h)^2}{a^2} + \frac{(y - k)^2}{b^2} = 1$$

Explicit form of an ellipse:

$$y = k \pm \frac{b}{a}\sqrt{a^2 - (x - h)^2}$$

Parabola

Explicit form of a parabola:

$$y = ax^2 + bx + c$$

Structure of the Lectures

The learning unit will be delivered during the last two months of the fourth year. It is deemed appropriate to conduct the LU over 8 lessons with the following structure:

- **Familiarize with the environment:** Two lessons will be allocated to familiarize students with studio.code.org, providing them with the necessary foundational knowledge and skills to navigate the platform effectively;
- **User Interface:** One lesson will be dedicated to implementing the management of the user interface and input of geometric equations. This session will focus on ensuring students understand how to interact with the application and input their geometric data accurately;

- **Implementation of the app:** Four lessons will be focused on implementing formulas of various geometric figures such as lines, circles, ellipses, and parabolas. These sessions will delve into the mathematical principles behind these shapes and guide students through the process of translating them into code within the studio.code.org environment.
- **Debugging and Evaluation:** The final lesson will be concentrated to debugging errors that may arise during the implementation process and conducting final evaluations.

Evaluation Grid

The evaluation grid will consist of various criteria that assess the students' performance and understanding of the learning unit. These criteria are: Programming Skills and Computer Science Concepts, Understanding of Mathematical Concepts, and Project Completion.

Grade	Programming Skills	Mathematical Concepts	Project Completion
Excellent	Programming concepts demonstrated with efficient solutions. Excellent translation of mathematical equations into visually graphical representations. Excellent implementation of user interface with seamless interaction and thorough input validation.	Advanced understanding with accurate application of mathematical concepts to solve all the equations. Proficient in interpreting the geometric significance of mathematical entities such as lines, circles, ellipses, and parabolas. Ability to manipulate both implicit and explicit forms of equations with versatility in algebraic operations.	Project completed with all required features implemented.
Good	Competent use of programming concepts, demonstrating understanding and effective implementation. Effective creation of graphical representations that reflect mathematical equations. Effective implementation of user interface with clear interaction and adequate input validation.	Clear understanding and application of mathematical concepts to solve the equations. Proficient in interpreting the geometric significance of mathematical entities. Ability to manipulate both implicit and explicit forms of equations with accuracy.	Project completed with required features implemented satisfactorily.
Sufficient	Basic application of programming concepts with some errors and limitations. Effective creation of graphical representations. Basic implementation of user interface, with some deficiencies in interaction and validation. Not all events are handled..	Basic comprehension of mathematical concepts, with some inaccuracies in formulas and results. Adequate understanding of the geometric significance of mathematical entities but with errors in interpretation. Ability to manipulate both implicit and explicit forms of equations, but with some limitations.	Project completed with half required features implemented.
Insufficient	Difficulty applying fundamental programming concepts, frequent errors. Lack of graphical representation. Ineffective implementation of user interface, lacking clarity and proper validation.	Limited understanding of key mathematical equations and difficulty in applying them. Limited interpretation of the geometric significance of mathematical entities. Difficulty in manipulating both implicit and explicit forms of equations.	Incomplete project with several required features missing or not functioning correctly.

Development

The application development followed the following logic: the application was designed with four main levels: Line, Circle, Parabola and Ellipse. To proceed to the next level, the user must input the correct equations given certain values/data. They are as follows:

1. For the line, the user must input an equation in explicit form to proceed to the next level, for example: $y = 2x + 1$.

2. For the circle, the user is asked to input the equation of the circle passing through the origin with a radius of 2.
   Solution: $x^2 + y^2 = 4$.

3. For the parabola, the user is asked to input the equation of a parabola with its vertex at the origin of the Cartesian axes and passing through the point (1,2).
   Solution: $y = 2x^2$.

4. For the ellipse, the user is asked to input the equation of the ellipse with vertices at the points (5,0), (-5, 0), (0,3), (0, -3).
   Solution: $\frac{x^2}{25} + \frac{y^2}{9} = 1$.

For each level, parameters will be calculated regardless of the values entered. This allows the user to understand whether the entered equation is correct or not and why. So, the following information will be displayed:

- **Level 1 (Line):**
  The user-entered equation represents a line. The equation is displayed in slope-intercept form. Additionally, the slope 'm' of the equation is shown.
  Equation: $y = mx + q$

- **Level 2 (Circle):**
  The user-entered equation represents a circle. The equation is displayed in general form along with information about the center and radius of the circle.

  Equation: $x^2 + y^2 + ax + by + c = 0$

- **Level 3 (Parabola):**
  The user-entered equation represents a parabola. The equation is displayed in standard form. The vertex (h, k) and focus (F) of the parabola are also calculated and displayed.

  Equation: $y = ax^2 + bx + c$

- **Level 4 (Ellipse):**
  The user-entered equation represents an ellipse. The equation is displayed in standard form, along with information about the semi-major and semi-minor axes of the ellipse. This equation allows users to observe how changing coefficients a and b affects the shape and size of the ellipse.

  Equation: $\left(\frac{x^2}{a^2}\right) + \left(\frac{y^2}{b^2}\right) = 1$

An additional section will be provided where users can input coefficients a, b, and c to observe their effects on the equations. This interactive feature aims to enhance the learning experience by allowing students to manipulate the coefficients and observe how it affects the equation's outcome.

By exploring variations in a, b, and c, students can develop a deeper understanding of the relationships within the equations.

Furthermore, two types of solutions have been identified: optimal and sufficient. Below is an explanation of the distinction between them.

Optimal Solution

Regarding the optimal solution, you can view the application at this link. In this case, all four levels are implemented with their respective functions for drawing the equations. (This can be seen from the images below.) When all levels have been solved, the end screen will be displayed.

Sufficient Solution

Regarding the Sufficient solution, you can access the application at this link.
The main difference between the optimal solution and the sufficient solution lies in the number of levels. In this case, the student only has to implement the line and the circle equations.
The Following images show how the application works:

Level: 1

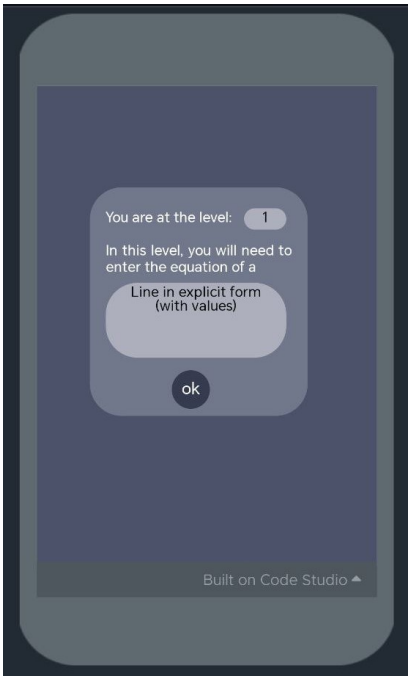

Figure 1: Explanation screen

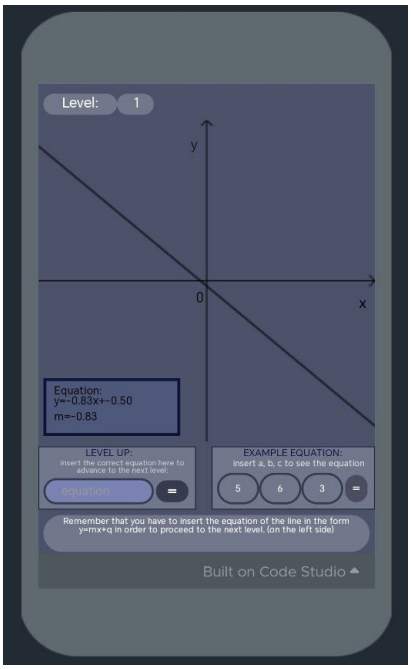

Figure 2: Inserting a, b, c coef.

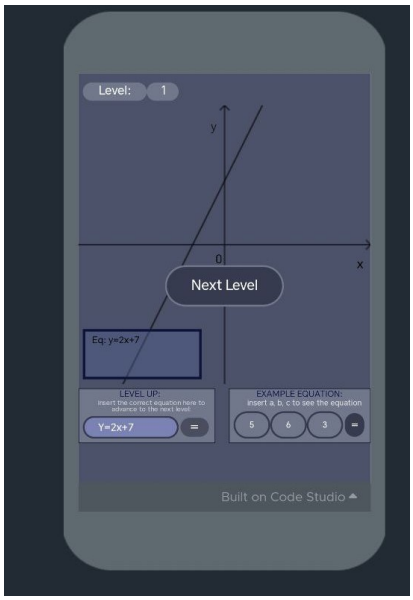

Figure 3: Inserting an equation.

Level: 2

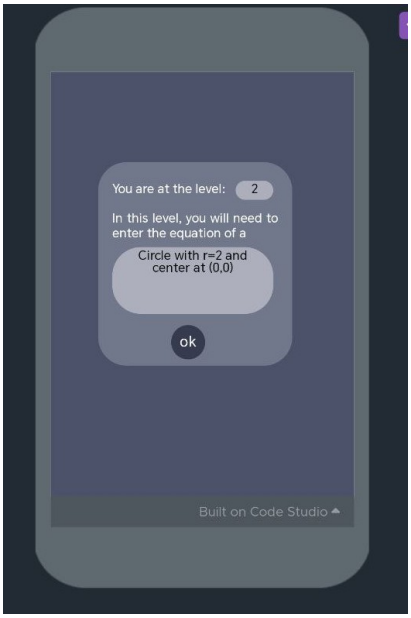

Figure 4: Explanation screen

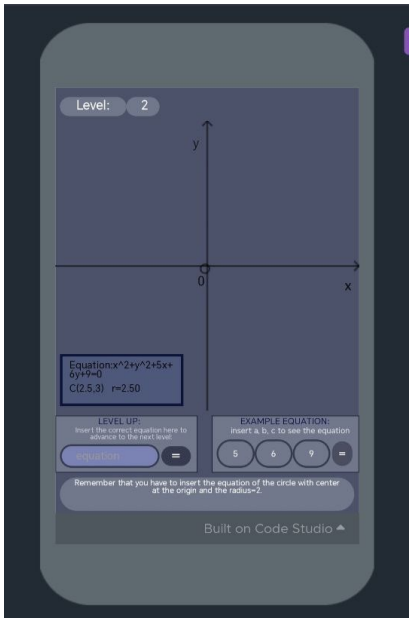

Figure 5: Inserting a, b, c coef.

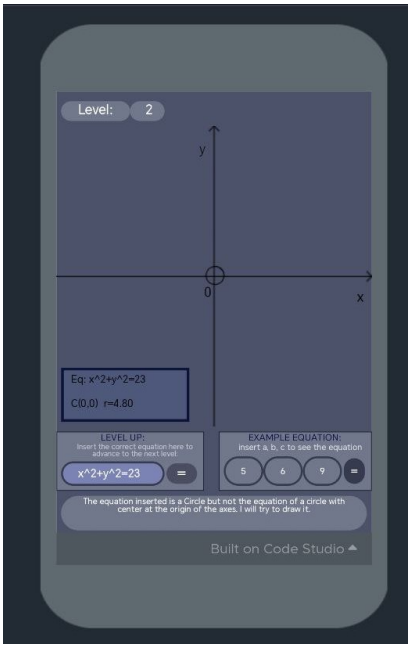

Figure 6: Inserting an equation.

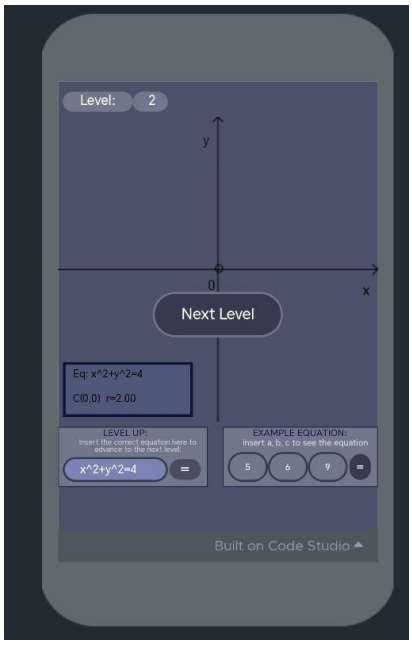

Figure 7: Drawing the right equation.

Level: 3

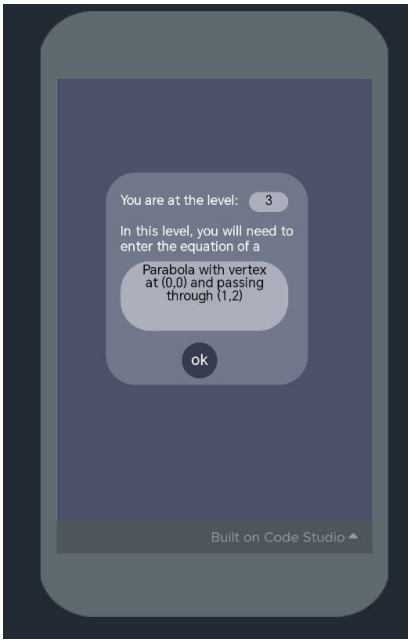

Figure 8: Explanation screen

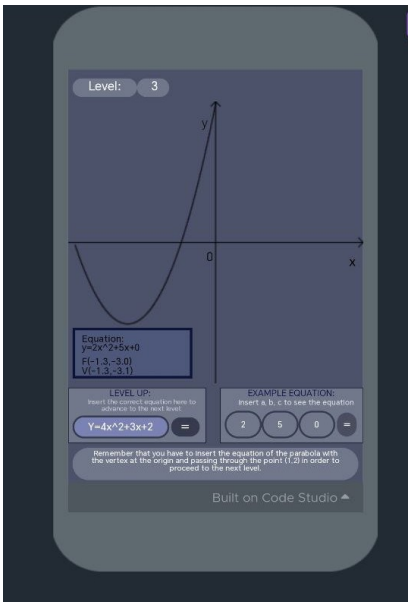

Figure 9: Inserting a, b, c coef.

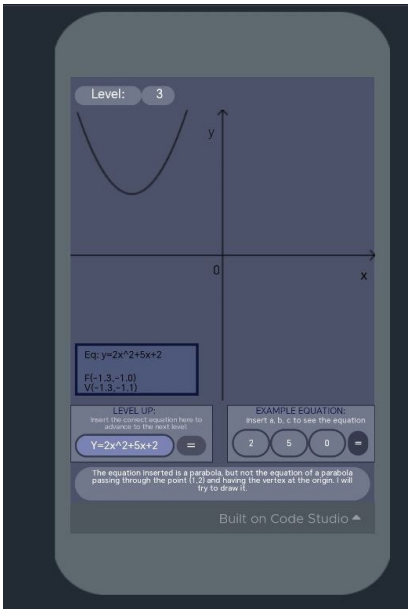

Figure 10: Inserting an equation.

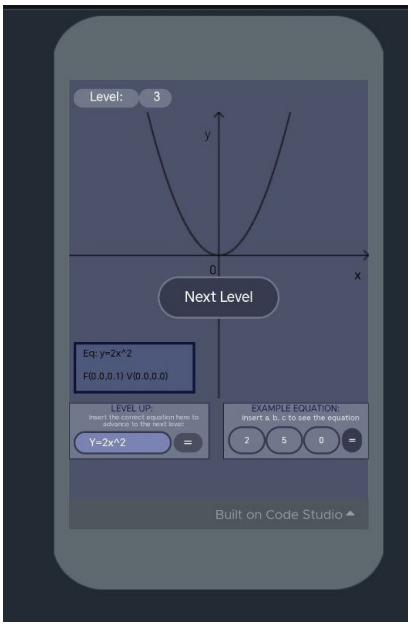

Figure 11: Drawing the right equation.

Level: 4

Figure 12: Explanation screen

Figure 13: Inserting a, b, c coef.

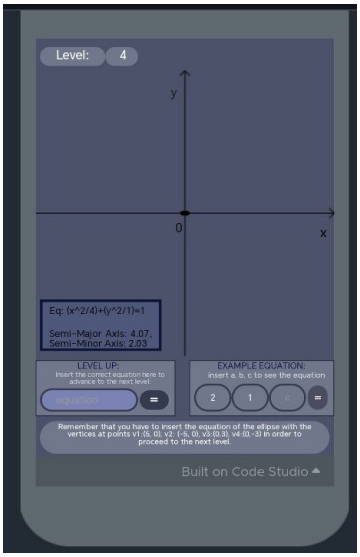

Figure 14: Inserting an equation

End Game

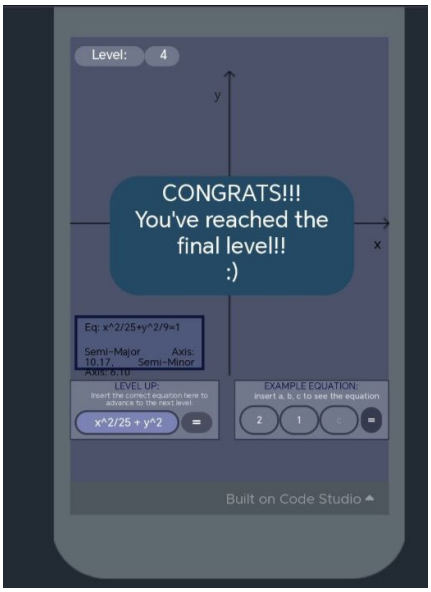

Figure 15: End Game!

References

1. studio.code.org
2. Regex generator site
3. Optimal Solution
4. Sufficient Solution