User Guide

This chapter provides a complete guide to using the developed L-System application. It is intended to help first-time users navigate the interface, understand the parameters involved, and generate various fractal structures, particularly plant-like forms, with ease.

0.1 Overview of the Interface

Upon launching the application, the user is presented with a graphical interface that consists of three primary sections:

- Control Panel (Left Side): Contains all input fields, predefined L-system selector, a checkbox for custom drawing commands, and the generate button.
- Drawing Panel (Right Side): Occupies the majority of the screen and is responsible for displaying the generated fractal output.
- Menu Bar (at the top): Contains file and help menu.

If the control panel content exceeds the window height, a vertical scroll bar will appear to allow easy access to all input fields.

0.2 Alphabet and Command Interpretation

This application uses a fixed internal alphabet that maps specific characters to turtle graphics commands. Users can define L-System rules using these characters directly, or assign their own characters to drawing commands through the Command input.

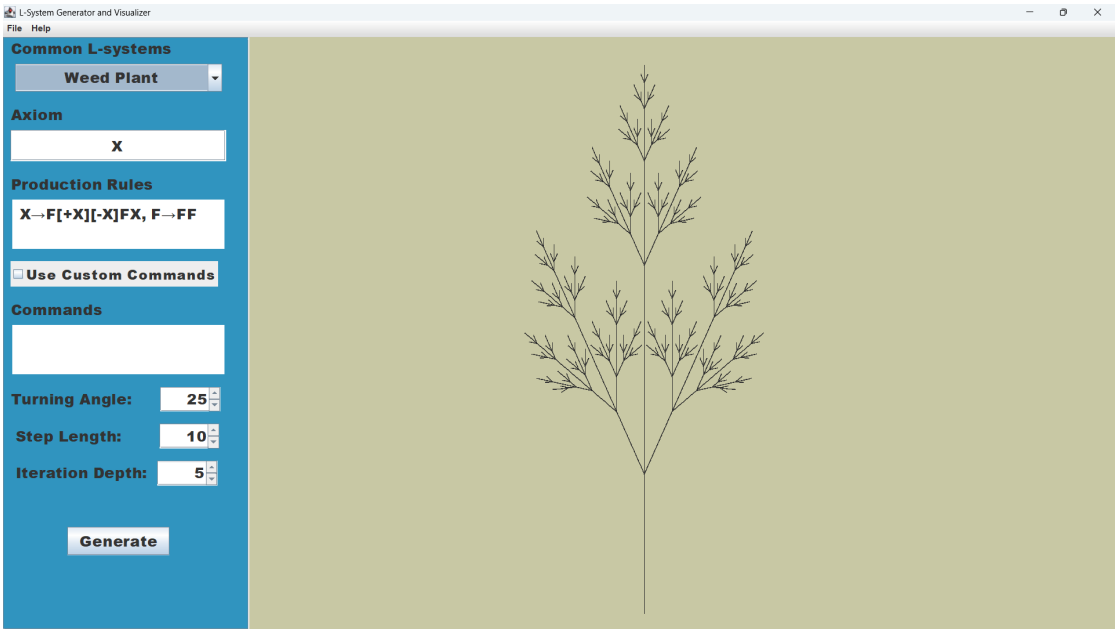


Figure 0.1: screenshot of the full GUI layout

Default (Predefined) Alphabet and Interpretation

The following characters (see table 0.1) are hard-coded into the application with built-in interpretations. If used in rules, they will automatically be processed using the associated turtle graphics behavior.

Characters	Command	Interpretation
A, B, F, G, X, Y	MOVE	Each character interpreted as move forward and draw a line
+	TURNRIGHT	Turn right by the specified angle
-	TURNLEFT	Turn left by the specified angle
[	PUSH	Save the current state (position and angle)
]	POP	Restore the most recently saved state

Table 0.1: Predefined Characters and Their Interpretations

Custom Command Mapping

In addition to the default characters, users can define their own command mappings using the Commands input field. This allows users to assign turtle commands to custom characters.

Supported command values include:

- MOVE – Move forward and draw
- TURNRIGHT – Turn right by the specified angle
- TURNLEFT – Turn left by the specified angle
- PUSH – Save the current state (position and angle)
- POP – Restore the most recently saved state

0.3 Input Fields and Their Functions

1. **Axiom**

   Axiom is the initial state where the generation starts. The Axiom can be any characters from the set of the alphabet used but can't be empty or blank space. In first generation the axiom replaced by the rewriting rules set for the characters in the axiom which implies the axiom has a direct effect on the output of the string generated.

2. **Production Rules**

   Production Rules are the rewriting rules in which we define single character to be replaced with string replacement. The production rules can be more than one rule.

   - Defines the production rules in the form of character→replacement string
   - Multiple rules are separated by comma (,).
   - Example: F→FF, X→F[+X]F[-X]+X

3. **Turning Angle**

   - The angle (in degree) used for turning left or right when interpreting the drawing commands.
   - It significantly affects the shape and orientation of the generated structure.

4. **Iteration Depth**

   - The number of times the production rules are recursively applied to the axiom.

- Higher values result in more complex structures and increases the processing time.

5. **Step Length**
    - Specifies the length(in pixel) of each drawing step when characters interpreted as "move forward and draw".
    - A higher value results in longer lines and a lower value creates more compact drawings.
6. **Common L-Systems**
    - A drop-down menu that provides a list of common L-system configurations such as the Koch curve, Dragon curve, and plant-like structures.
    - Selecting one automatically triggers drawing generation and fills in the axiom, rules, and input parameters.
7. **Custom Drawing Commands (Optional)**
    - Enabled/disabled via a checkbox.
    - Allows users to define their own interpretation of characters.
    - Defines commands in form of character=command and separate by comma(,)
    - Example: F=MOVE, +=TURNRIGHT, -=TURNLEFT, [=PUSH,]=POP

0.4 Generating L-system: Step-by-Step Example

Using predefined L-system

- From the drop down, select any L-system.
- It will draw the output and fill the parameters automatically.

Let us consider generating a simple plant-like fractal by filling the parameters manually.

Step-1: Fill in the parameters as follows:

- Axiom: X

- Rules: X→F-[[X]+X]+F[+FX]-X, F→FF
- Turning Angel: 25
- Step Length: 5
- Iteration Depth: 5

Step-2: Click "Generate"

- The drawing panel will render a fractal resembling a branching plant structure (see figure 0.2).
- Make sure the use custom commands checkbox is not checked otherwise you need to provide a command for each character or you will get error.

Using Custom Commands
In addition to filling the other parameters manually we need to fill in the commands. To do so:
Step-1: Check the “Use Custom Drawing Commands” checkbox.

Step-2: Enter the custom command mappings as follows:

- F=MOVE, X=MOVE +=TURNRIGHT, -=TURNLEFT, [=PUSH,]=POP
- Make sure every characters in the axiom and rules have command mapping

0.5 Understanding the Effect of Parameters

This section explains how each input parameter affects the final output of the L-system visualization. Users can manipulate these parameters to explore how L-system complexity, orientation, and scale are influenced by different configurations.

Axiom

- Changing the axiom alters the base structure of the generated string.
- Using more than one character can initiate more complex, recursive growth depending on how the character is defined in the rules.

Rule

Figure 0.2: Screenshot showing the generated plant-like L-system

- Production rules define how each character in the string is replaced during each iteration
- Modifying a rule may change the output of the structure entirely.
- The output is highly affected by the rewriting rules, for example, the change of one character in the rewriting rule will completely change the output.
- For example let compare the following three rules without changing the other parameters.
    1. Rule 1: F→FF, X→F[+X]F[-X]+X
    2. Rule 2: F→FF, X→F[+X]F[-X]+
    3. Rule 3: F→F, X→F[+X]F[-X]+X
    4. Axiom: X, Turning Angel: 20, Step Length: 5 and Iteration Depth: 6

As we can see in the figures 0.3, the change of one character in the rule results in a very different result.

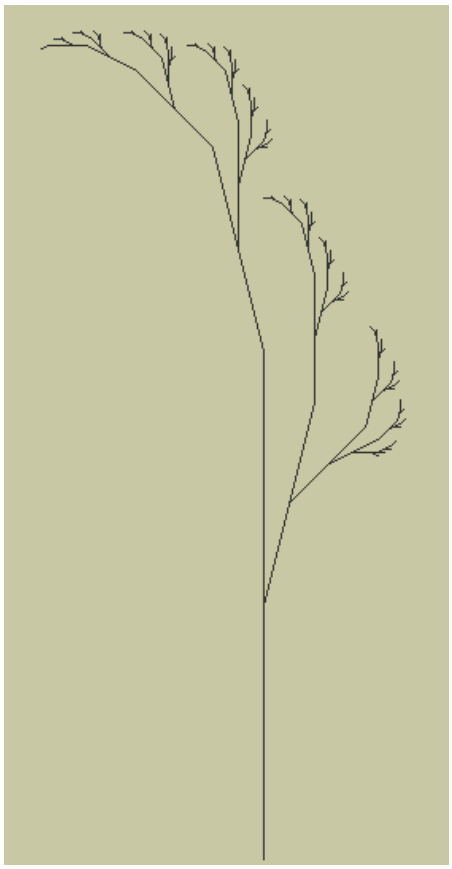
Turning Angle

- Changing the turning angle modifies the branching pattern.
- Example: If we change the turning angle from 25 to 15. The branches become narrower and the plant appears more vertical. Increasing to 45 makes the plant spread wider.(see figure 0.4)
- Other parameters; Axiom: X ; Rules: X→F-[[X]+X]+F[+FX]-X, F→FF ; Step Length: 5 ; Iteration Depth: 5

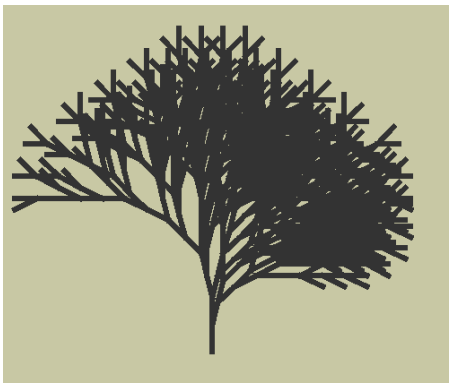
Iteration Depth

- Increasing iterations adds more recursive detail.
- Example: Use 3, 4, and 5 iterations. The plant becomes more complex and larger with the depth of iteration (see figure 0.5).
- Other parameters; Axiom: X ; Rules: X→F-[[X]+X]+F[+FX]-X, F→FF ; Step Length: 5 ; Turning Angel: 25
- Higher iterations slow down the rendering due to increased complexity.

(a) Rule 1



(b) Rule 2

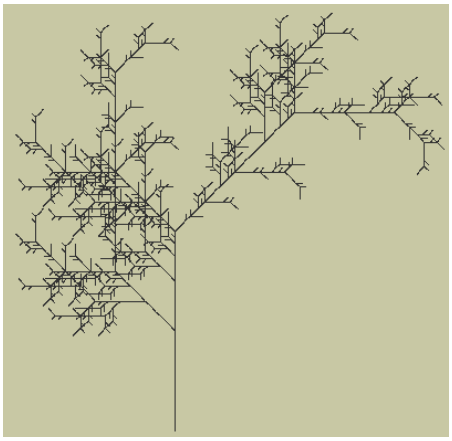


(c) Rule 3

Figure 0.3: L-system output for above example with Rule 1, 2 and 3

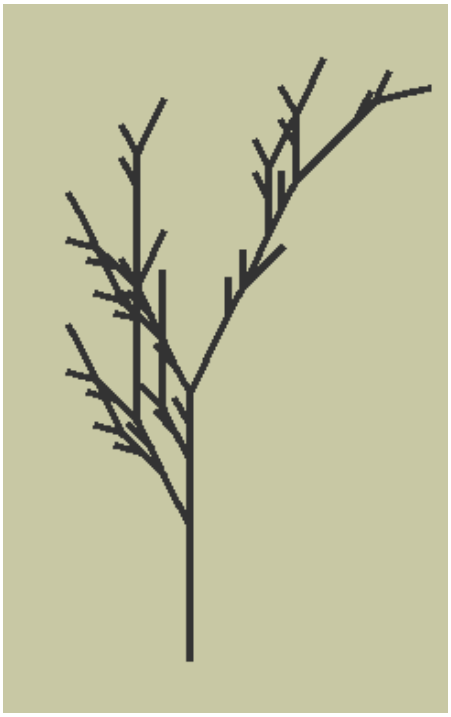
(a) Angle 15°

(b) Angle 25°

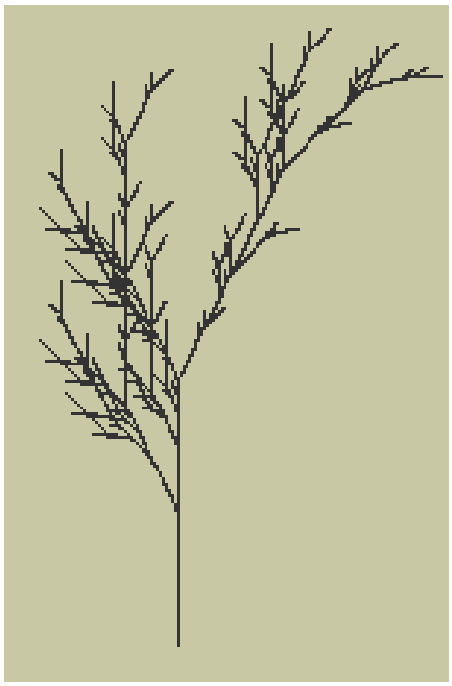


(c) Angle 45°

Figure 0.4: Output comparison for turning angle 15°, 25°, and 45°



(a) Three Iteration



(b) Four Iteration

(c) Five Iteration

Figure 0.5: Output comparison for iteration depth three,four, and five

Step Length

- A shorter step results in a more compact drawing.
- A longer step stretches the drawing, making it larger.(see figure 0.6)
- Other parameters; Axiom: X ; Rules: X→F-[[X]+X]+F[+FX]-X, F→FF ; Turning Angel: 25 ; Iteration Depth: 5

(a) Step Length three

(b) Step Length eight

Figure 0.6: Output comparison of step length three and eight

0.6 Menu Items

File Menu Items

- **Save:** Saves the current screen as an image file (PNG).
- **Exit:** Closes the application.

Help Menu Items

- **About:**Displays application version and author information.

- **User Guide:** Opens the user guide in PDF.

0.7 Error Messages and Troubleshooting

If an input is invalid, an error message will be shown in the dialog box.
Examples of errors include:

- Missing input parameter (e.g., missing Axiom or Rule)
- Incorrect rule format (e.g., missing $\rightarrow$)
- Incorrect command format (e.g., missing =)
- Exceeding the upper limit 50M characters

When an error occurs, read the message displayed and correct accordingly. For example, (see figure 0.7) after the first "F" symbol "$\rightarrow$" is missing, so click OK and correct the rule, then regenerate again.

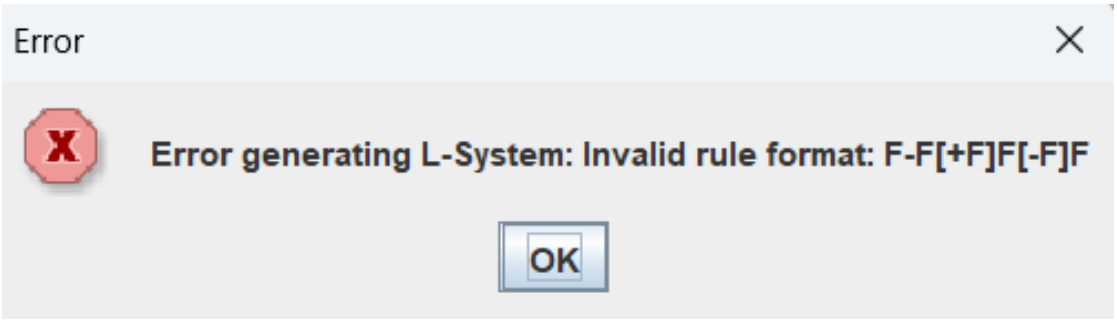


Figure 0.7: Screenshot showing error message for invalid rule format