

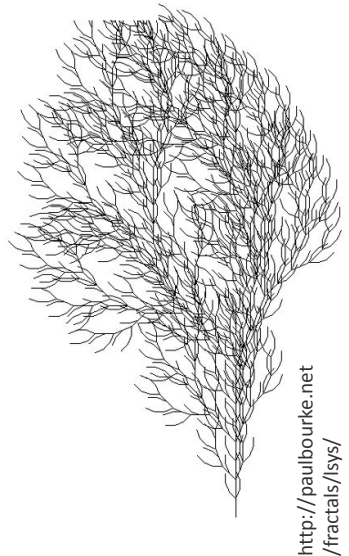
Implementierung eines einfachen Lindenmayer-Systems

Abschlusspräsentation

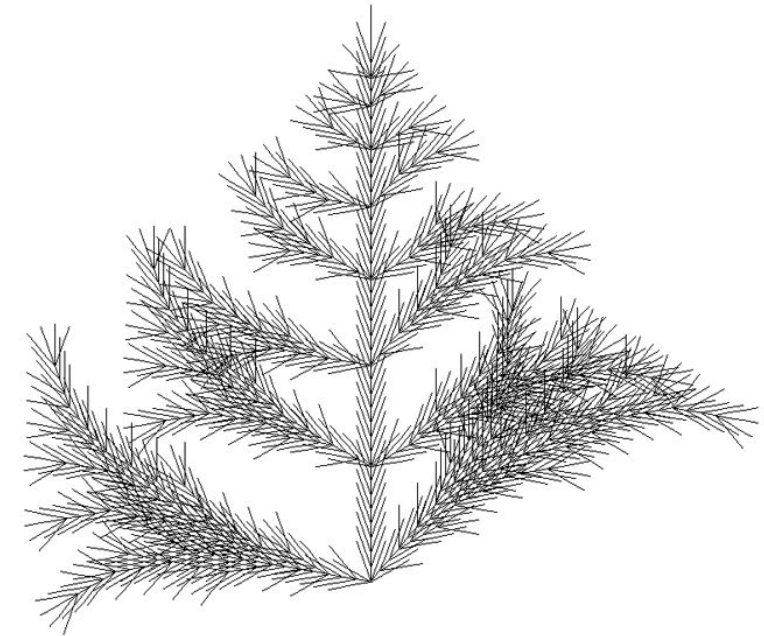
vorgelegt von: Sebastian Seidel, Aniket Rodrigues, Laurids von Emden

Leiter des Fachgebiets: Prof. Dr.-Ing. Clemens Gühmann

Betreuer: M. Sc. Daniel Thomanek



- ▶ Einführung und Projektziel
- ▶ Wie Lindenmayer-Systeme funktionieren
- ▶ Entwicklungsziele
- ▶ Vorstellung der Software
- ▶ Softwaretest
- ▶ Fazit und Ausblick



Allgemein übergeordnete Projektziele:

- Möglichkeit der Prädiktion des Pflanzenwachstums entlang von Bahnstrecken
- Voraussagemöglichkeit, wann die Bepflanzung die geometrischen Grenzen des Regellichtraums erstmalig verletzen wird
- Zeitplanung von Pflanzenschnitt und Prognostizierung, von Maßen für die zurückzuschneidende Biomasse

□ Lindenmayer-Systeme können diese Projektziele auf lange Sicht erfüllen



<https://de.wikipedia.org/wiki/Lichtraumprofil>

Warum Lindenmayer-Systeme und wie funktionieren sie?

„After the incorporation of geometric features, plant models expressed using L-systems became detailed enough to allow the use of computer graphics for realistic visualization of plant structures and developmental processes.”

Prusinkiewicz, P. & Lindenmayer, A. (2004) „The Algorithmic Beauty Of Plants“

- ▶ L-System ist mathematischer Formalismus zur Erzeugung von Fraktalen
- ▶ L-System = Ersetzungssystem: Rekursives Ersetzen eines Objekts durch bestimmte vordefinierte Regeln
 - (Iteration einer bestimmten Schleife mit vorgegebenen Formeln)
 - ▶ Ergebnis: Zeichenkette (abstrakte Liste)
- ▶ Grafische Darstellung der Zeichenkette: Mittels Turtle-Grafik
 - ▶ Stifttragender Roboter auf Zeichenebene führt einfache Kommandos aus

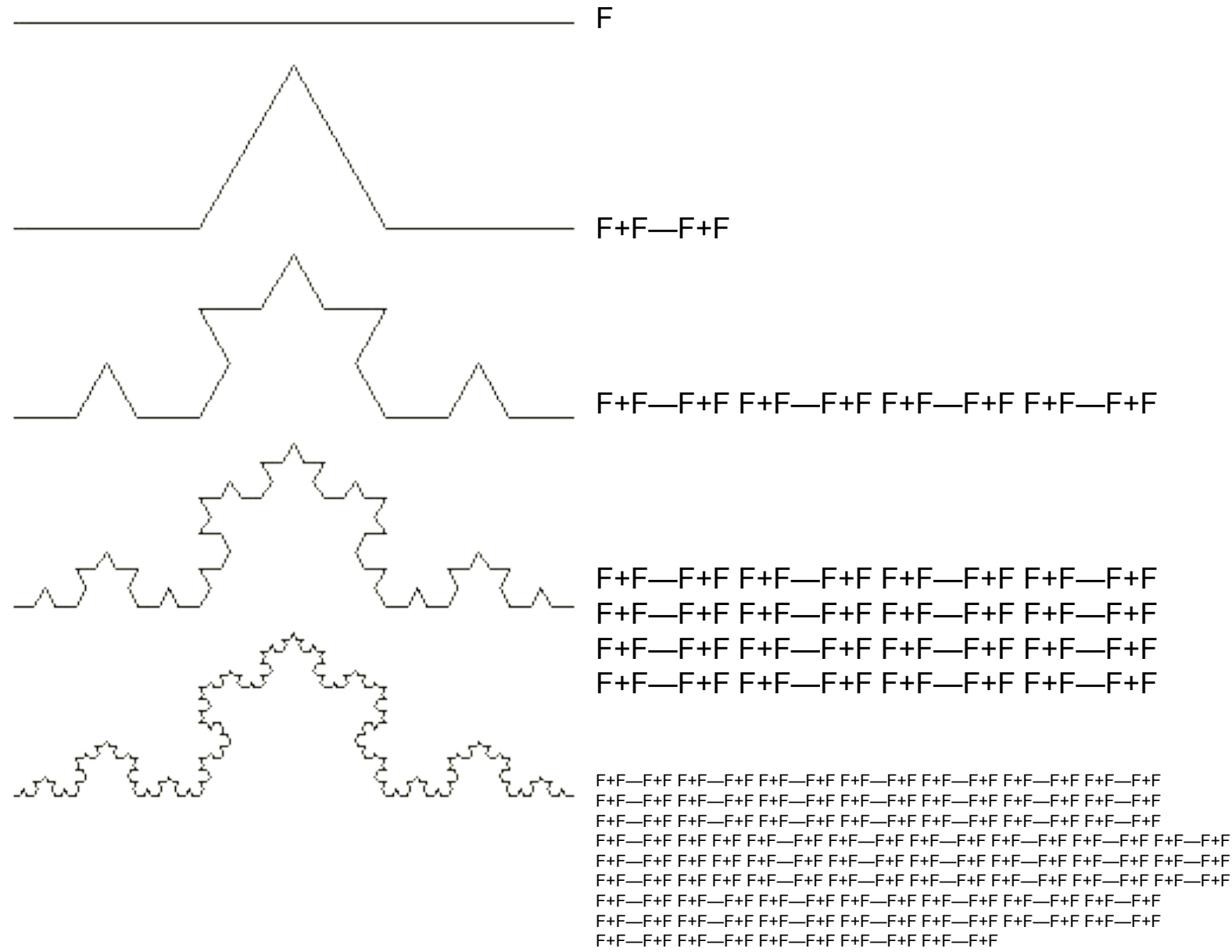
Beispiel: Koch-Konstruktion

Startwert: F

Ersetzungsbefehl: $F \rightarrow F+F--F+F$

Winkel: 60°

Iterationen: 4

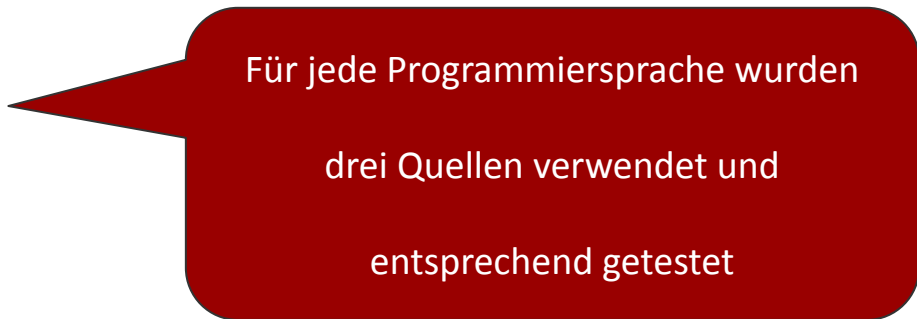


Entwicklungsziele

► Recherche zu vorhandenen Lindenmayer-System Implementierungen

► Gewählte Programmiersprachen (GitHub):

- 1) Python
- 2) Java
- 3) JavaScript
- 4) Delphi (auch Object Pascal genannt)



Für jede Programmiersprache wurden
drei Quellen verwendet und
entsprechend getestet

► Auswahlkriterien betrachteten Quellcode:

- 1) Performance
- 2) Strukturellen Aufbau
- 3) Übersichtlichkeit des Quellcodes
 - a) Einfaches Verständnis
 - b) Anpassbarkeit

Entwicklungsziele

- ▶ Arbeit mit der Software:
Anpassen einer ausgewählten Implementierung
 - ▶ Visuelle Darstellung von Pflanzenwuchs in verschiedenen Wachstumsstadien
 - ▶ Gesamtes Wachstum als kontinuierliche Simulation
 - ▶ Erstellung einer GUI
 - 1) Eingabefeld
 - 2) Grow-Button: Für die kontinuierliche Simulation
 - 3) Prozentsatz des simulierten kontinuierlichen Pflanzenwachstums
 - 4) Show-Step: Einsehen des Wachstums nach jeweiliger Iteration
 - 5) Reset-Button

- ▶ Dokumentation der Rechercheergebnisse, des Evolutionsprozesses der Software, Erstellen von Wartungs- und Benutzerhandbuch

Vorstellung der Software – Benötigte Librarys

IDE: Spyder v5.1.5 (Anaconda 3)

Python: v3.9.7

1. Erstellung und Bearbeitung der GUI

- ▶ Library: `tkinter`

- ▶ Command: `conda install -c anaconda tk`

2. Exportieren des Bildes vom Canvas (= Zeichenfläche des Turtles)

- ▶ Library: `ghostscript`

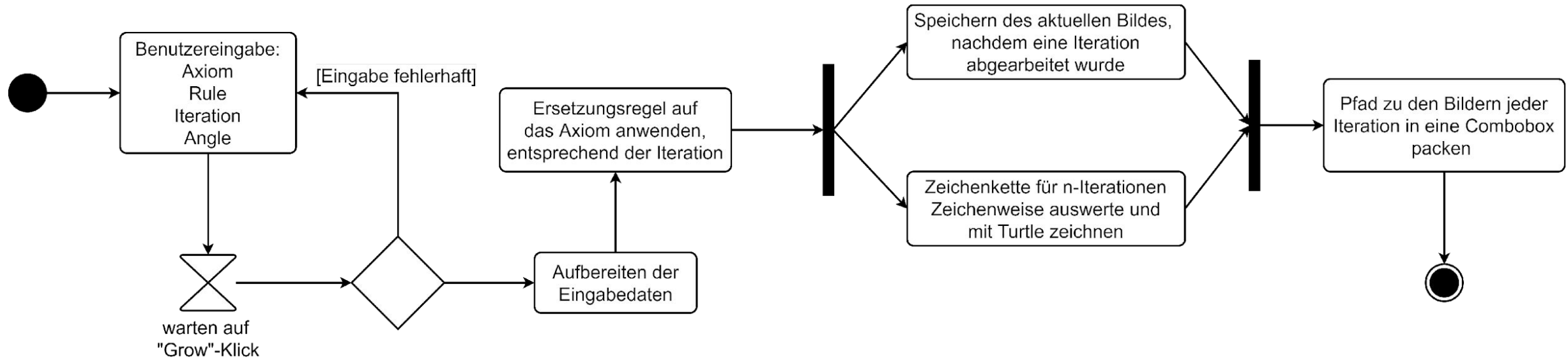
- ▶ Command: `conda install -c conda-forge ghostscript`

3. Laden der gespeicherten Bilder

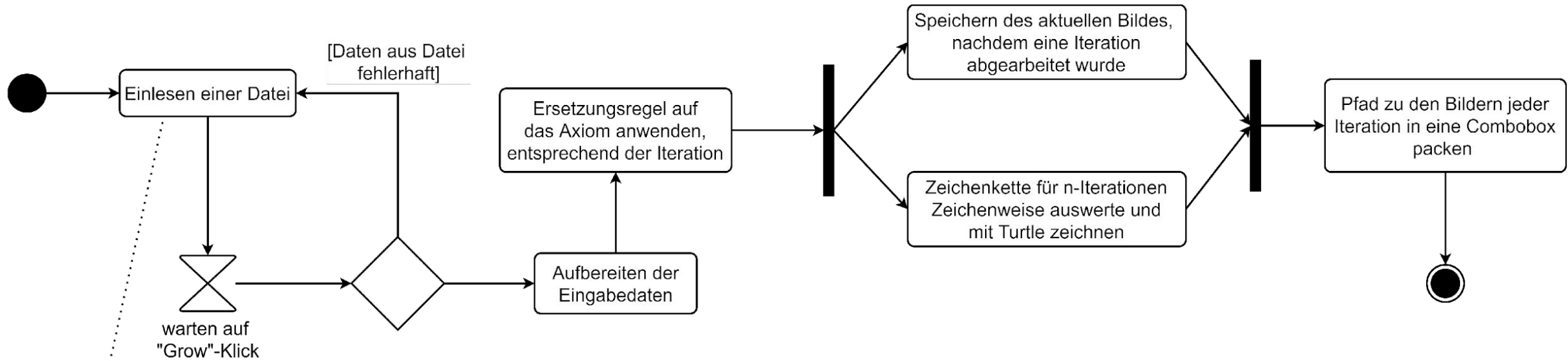
- ▶ Library: `pillow`

- ▶ Command: `conda install -c conda-forge pillow`

Vorstellung Software – Ablauf (Benutzereingabe)



Vorstellung Software – Ablauf (zukünftig)



Es müsste aber noch geklärt werden wie die Syntax sein soll mit die Regeln aus einer Datei geladen werden können z.B.

```

[Zeile] : [Inhalt]
[1]      : F          {Axiom}
[2]      : 4           {Iteration}
[3]      : 22.5        {Angle}
[4]      : F=.....   {Rule 1}
[5]      : ...=.....  {Rule 2}
[...]
```

► Falsche Eingabe

Lindenmayer-System

Missing Entry

The "Axiom" must not be empty.

OK

Axiom:

Show Step:

Rule:

F=F[+F]F[-F]F

Iteration:

4

Angle:

25.7

Grow

Reset

Lindenmayer-System

Wrong Format

"Rule" must contain "=", e.g. "F=FF+[+F-F-F]-[-F+F+F]".

OK

Axiom:

F

Show Step:

Rule:

F->F[+F]F[-F]F

Iteration:

4

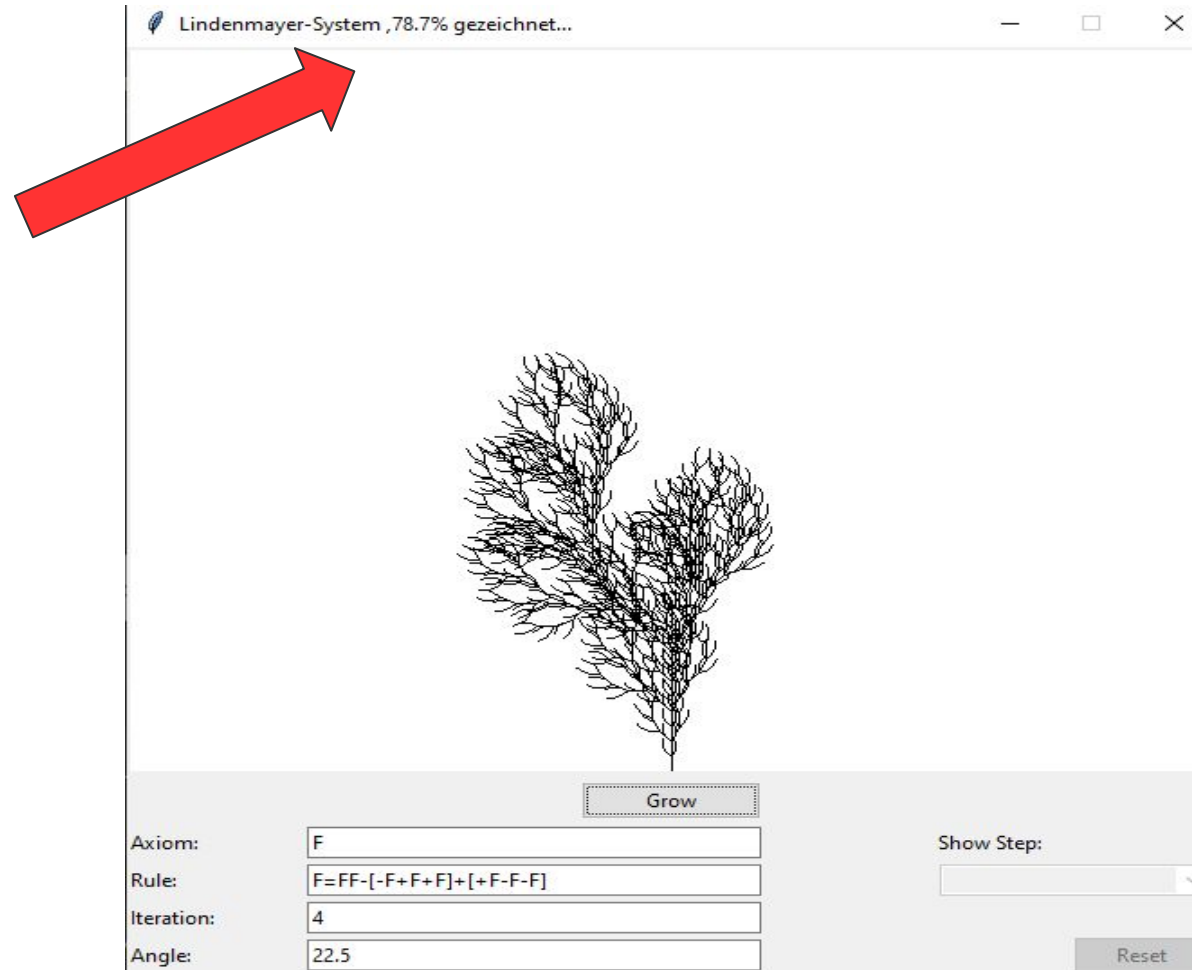
Angle:

25.7

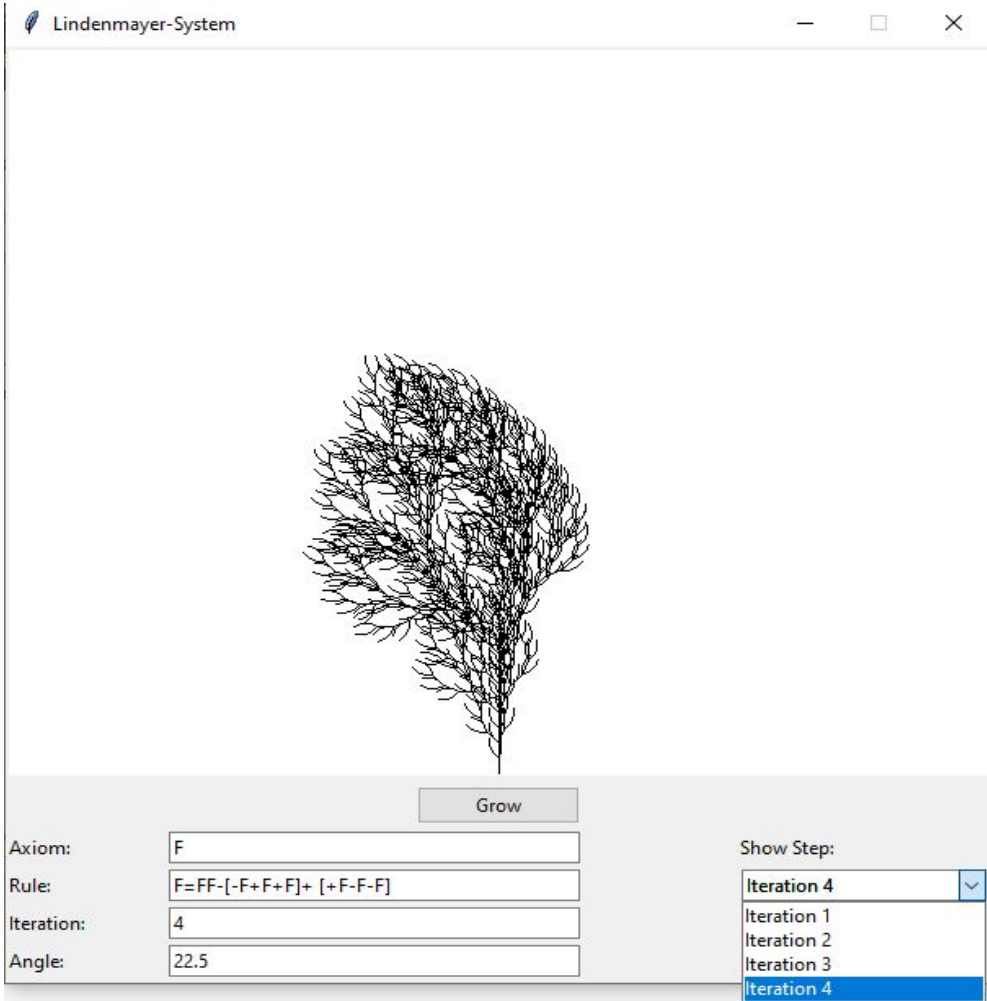
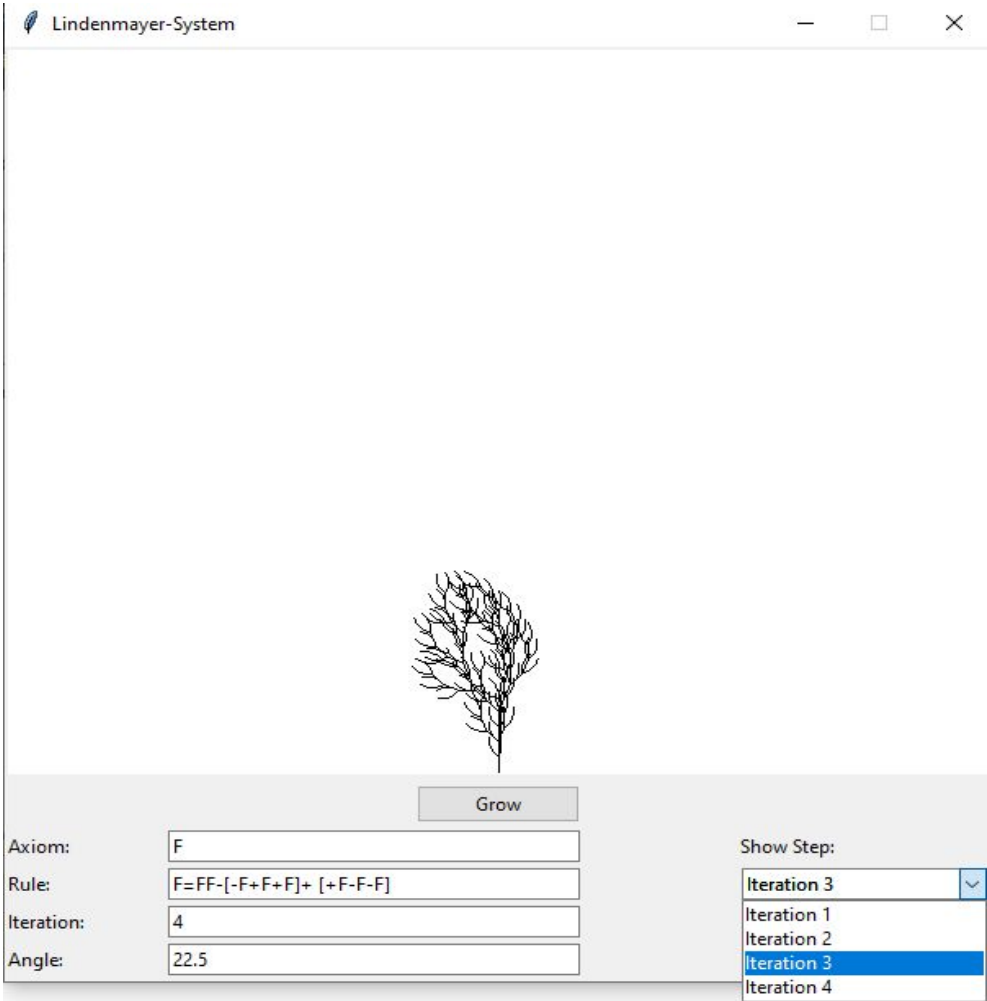
Grow

Reset

- Features der GUI (Prozentsatz des simulierten kontinuierlichen Pflanzenwachstums)



► Features der GUI (Show-Step: Einsehen des Wachstums nach jeweiliger Iteration)



► Simulation des Pflanzenwachstums (Beispiel 1):

- Axiom: F
- Rule: $F = FF + [+F - F - F] - [-F + F + F]$
- Iteration: 4
- Angle: 22.5



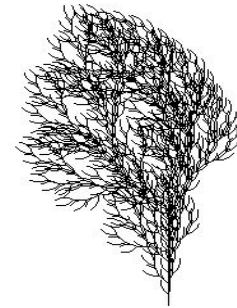
Iteration 1



Iteration 2



Iteration 3



Iteration 4

► Simulation des Pflanzenwachstums (Beispiel 2):

- Axiom: F
- Rule: $F = F[+F]F[-F]F$
- Iteration: 4
- Angle: 25.7



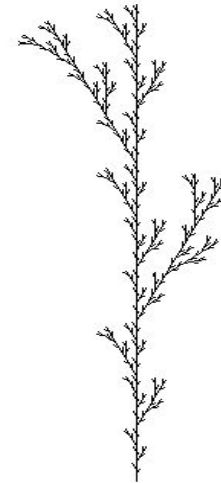
Iteration 1



Iteration 2



Iteration 3



Iteration 4

Fazit und Ausblick

- ▶ Recherche zu: Simulation von Pflanzenwuchs ✓
- ▶ Erste Implementierung einer Software ✓
- ▶ Dokumentation der Ergebnisse (✓) *bis 14.2.*
- ▶ Projektplanung und zeitlicher Ablauf ✓

- ▶ Mit Dokumentation Grundlage für weitere Gruppen gelegt
- ▶ Software Modular und übersichtlich aufgebaut
- ▶ GUI leicht erweiterbar
- ▶ Aufkommende Themen:
 - ▶ „Mehrere Bäume in einem Bild“
 - ▶ Bezug zu Bahnverkehr herstellen, Regellichraum einbeziehen

Vielen Dank für Ihre Aufmerksamkeit!
