

VIRTUAL REALITY IM CAVE
L-System Viewer

WS 2004/05

Christoph Spielmann 0155049

Programmierung

• OpenSG-Anwendung

Der Großteil der Zeit wurde in die Programmierung des OpenSG L-System-Viewers gesteckt. Die Anwendung besteht aus zwei Klassen (Turtle und Interpreter) und einer Datei (01hello.cpp), die das Setup von GLUT (= [OpenGL Utility Toolkit](#)), des SimpleSceneManagers und anschließend die Interpretation und Darstellung des L-Systems übernimmt.

Die Klasse Turtle kapselt alle für die Turtle-Grafik nötigen Daten. Diese Daten sind einerseits die aktuelle Position und Orientierung der Schildkröte und andererseits die Linienlänge für die Linien, die diese Schildkröte zeichnen muss, und den Änderungswert für die Orientierung, wenn die Turtle gedreht wird. Weiters enthält diese Klasse Methoden um die aktuelle Position der Schildkröte als [Pnt3f](#) zu ermitteln, eine Methode um die Schildkröte zu rotieren (mit einem Parameter in welche Richtung die Schildkröte gedreht werden soll) und eine Methode um die Turtle um einen Schritt zu bewegen.

Die Klasse Interpreter speichert die Daten, die für die Interpretation des L-Systems wichtig sind. Erstens muss natürlich eine Instanz der Klasse Turtle vorhanden sein, um das System zeichnen zu können. Um Bäume und Pflanzen möglichst realistisch darstellen zu können, müssen „Verzweigungsstrukturen“ berücksichtigt werden. Dies wird im L-System selbst durch Verwendung von '[' und ']' sichergestellt und bei der Interpretation wird vom Interpreter die aktuelle Turtle auf einen Stack gepusht, wenn ein '[' gefunden wird und das oberste Element des Stacks vom Stack entfernt und als aktuelle Turtle verwendet, wenn ein ']' entdeckt wird. Weiters wird die maximale Rekursionstiefe gespeichert, eine Liste mit allen Regeln verwaltet und das Axiom des L-Systems gespeichert. Als öffentliche Methoden enthält die Klasse natürlich eine Methode, die das L-System interpretiert und als Ergebnis einen [GeometryPtr](#) liefert. Eine Instanz der Klasse GeometryPtr ist ein sogenannter „Kern“ (für Informationen bez. der Struktur von OpenSG siehe [OpenSG-Tutorial](#)) der verwendet wird, um Geometriedaten zu speichern. Als private Methoden enthält diese Klasse erstens eine Methode, die die Struktur des zu zeichnenden L-Systems analysiert und gleichzeitig die internen Datenstrukturen (Axiom und Regeln) des Objekts füllt. Desweiteren ist eine rekursive private Methode vorhanden, die dann schlussendlich wirklich die Aufgabe hat, das gegebene L-System zu interpretieren und zu zeichnen.

L-System Editor

Die Entwicklung des L-System Editors war im Gegensatz zur Programmierung der OpenSG-Applikation relativ einfach.

Der Editor selbst besteht aus zwei Klassen: EditLsysForm und ViewLsysForm. Beide Klassen erben von System.Windows.Forms.Form. Das Design der Benutzeroberfläche wurde mit dem Formulardesigner von Visual Studio .Net 2003 erstellt. Die Programmierung selbst beschränkte sich auf die Ausprogrammierung der, von der Oberfläche benötigten, Event-Prozeduren. Das einzige wirklich erwähnenswerte ist die Art wie die die OpenSG-Anwendung vom L-System Editor aufgerufen wird: Es wird ein eigenes Process-Objekt für die OpenSG-Applikation instantiiert, der Name des zu zeichnenden L-Systems und die im Formular eingestellte Rekursionstiefe als Kommandozeilenargumente für die Applikation

angegeben und anschließend der Prozess gestartet.

Benutzerführung

Aufbau einer L-System-Datei:

Eine vom Viewer und Editor verarbeitbare Datei hat folgenden Aufbau:

Angle=90; <= Der Winkel, um den die Turtle bei einer Drehoperation gedreht wird
Axiom=L; <= Das Axiom (der „Startpunkt“ der Interpretation) dieses L-Systems
Danach folgt eine beliebige Anzahl von „Regeln“ (jede Regel muss in einer eigenen Zeile stehen, damit sie vom Editor richtig gelesen werden kann!!). Eine Regel hat wie folgt auszusehen: <Buchstabe OHNE F,f>={<Buchstabe>|+|-}; ({} steht für eine beliebige Anzahl). Die Zeichen 'F' und 'f' dürfen nicht als Bezeichner für eine Regel verwendet werden, da es sich bei den beiden Zeichen um „Steuersymbole“ für die Schildkröte handelt:

- F heißt: Bewege Turtle und zeichne Linie
- f heißt: Bewege Turtle aber zeichne KEINE Linie

Der Inhalt einer Beispieldatei sehe z.B. so aus (dieses L-System zeichnet die Hilbert-Kurve!):

```
Angle=90;  
Axiom=L;  
L=+RF-LFL-FR+;  
R=-LF+RFR+FL-;
```

L-System-Editor:

Der L-System-Editor ist eine in C# entwickelte Applikation, mit deren Hilfe man L-Systeme erstellen und bearbeiten kann. Falls auf dem System OpenSG und der L-System-Viewer installiert sind kann man auch aus dem L-System-Editor sofort das L-System betrachten.

Mit Hilfe der Textfelder **Angle** und **Axiom** werden Winkel und Axiom für das L-System angegeben. In das Textfeld Rule wird eine Regel eingegeben und wenn das Textfeld den Fokus verliert und die Regel gültig (für Gültigkeitsregel siehe oben) ist wird die Regel in die Liste mit den Regel eingefügt.

Mit den Buttons **Open** und **Save** kann ein L-System-File geladen bzw. abgespeichert werden. Beim Öffnen eines L-System-Files wird die Datei gelesen und wenn der Inhalt der Datei korrekt ist, in die zugehörigen Textfelder und die Regelliste eingetragen. Beim Abspeichern eines L-Systems wird das erstellte L-System in eine Datei geschrieben.

Mit Hilfe der **ViewLsysForm** (Illustration 2) kann man sich das eben im Editor erstellte L-System auf dem Bildschirm darstellen lassen (Die Voraussetzung dafür sind weiter oben genannt). Mit dem Button **Choose...** muss man die, bei der Kompilierung der OpenSG-Anwendung erstellte, ausführbare Datei auswählen. Wie der Name schon vermuten lässt, wird der Spinner **Recursion depth** dazu verwendet um die Rekursionstiefe für die Interpretation einzustellen.

Wenn das L-System dargestellt wurde (es öffnet sich ein eigenes Fenster mit dem

Titel L-System Viewer (siehe Illustration 3), in dem das gezeichnete L-System dargestellt ist), kann man mit Hilfe der Maus das L-System rotieren. Falls man den L-System-Viewer von Hand starten will, ist es wie folgt zu machen: <Name der ausführbaren Datei> <Name der L-System-Datei> <Rekursionstiefe>!

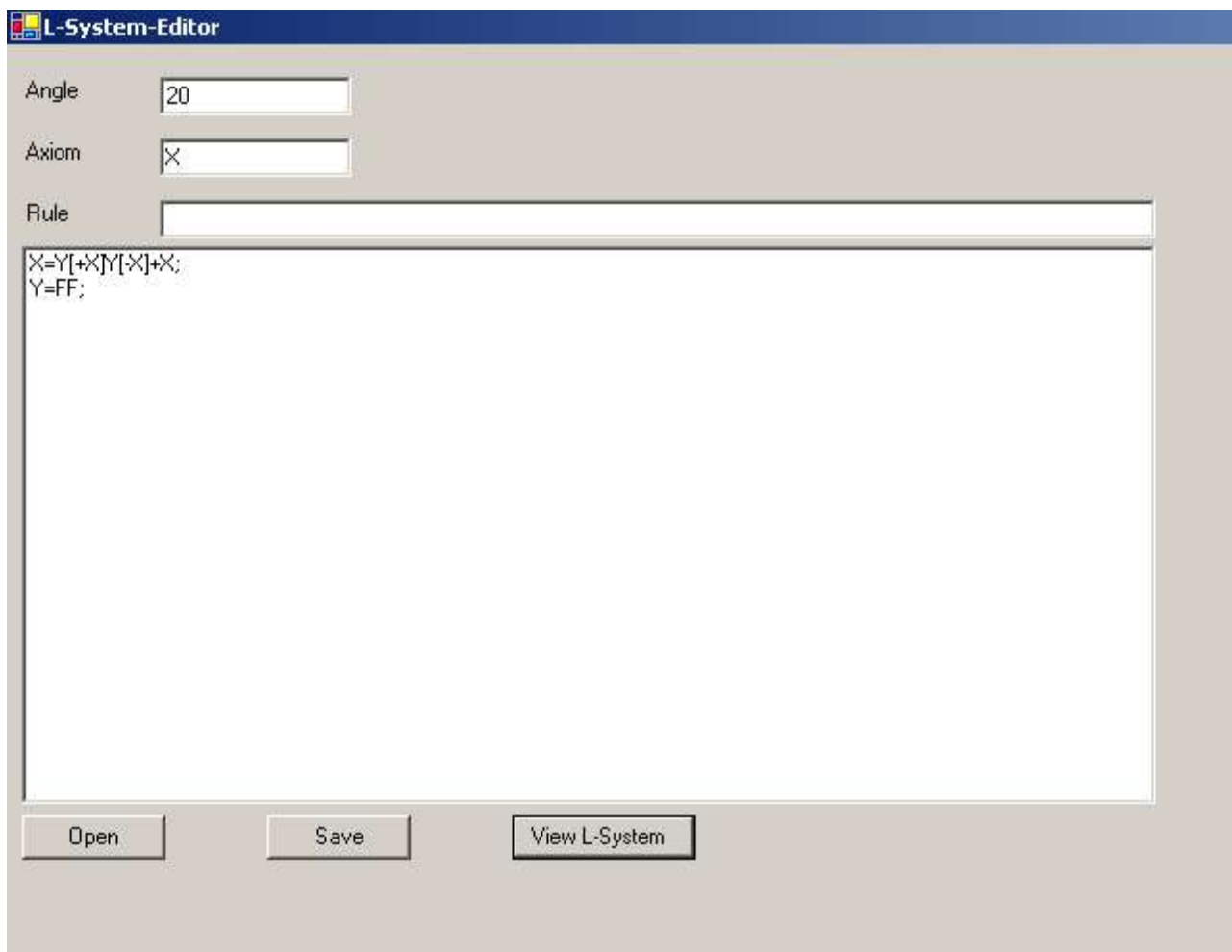


Illustration 1-Screenshot des L-System-Editors

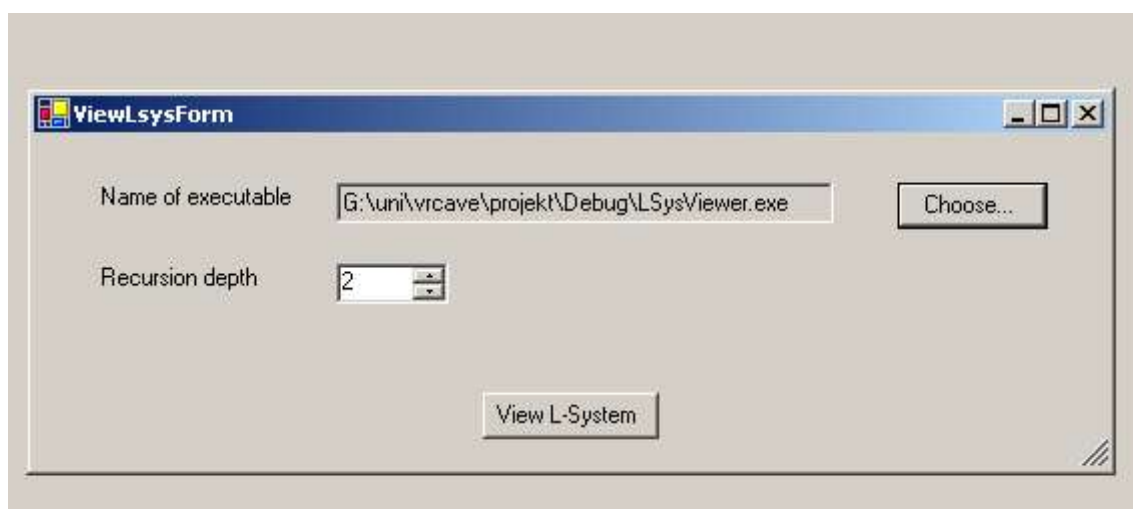


Illustration 2-Screenshot der ViewLsysForm

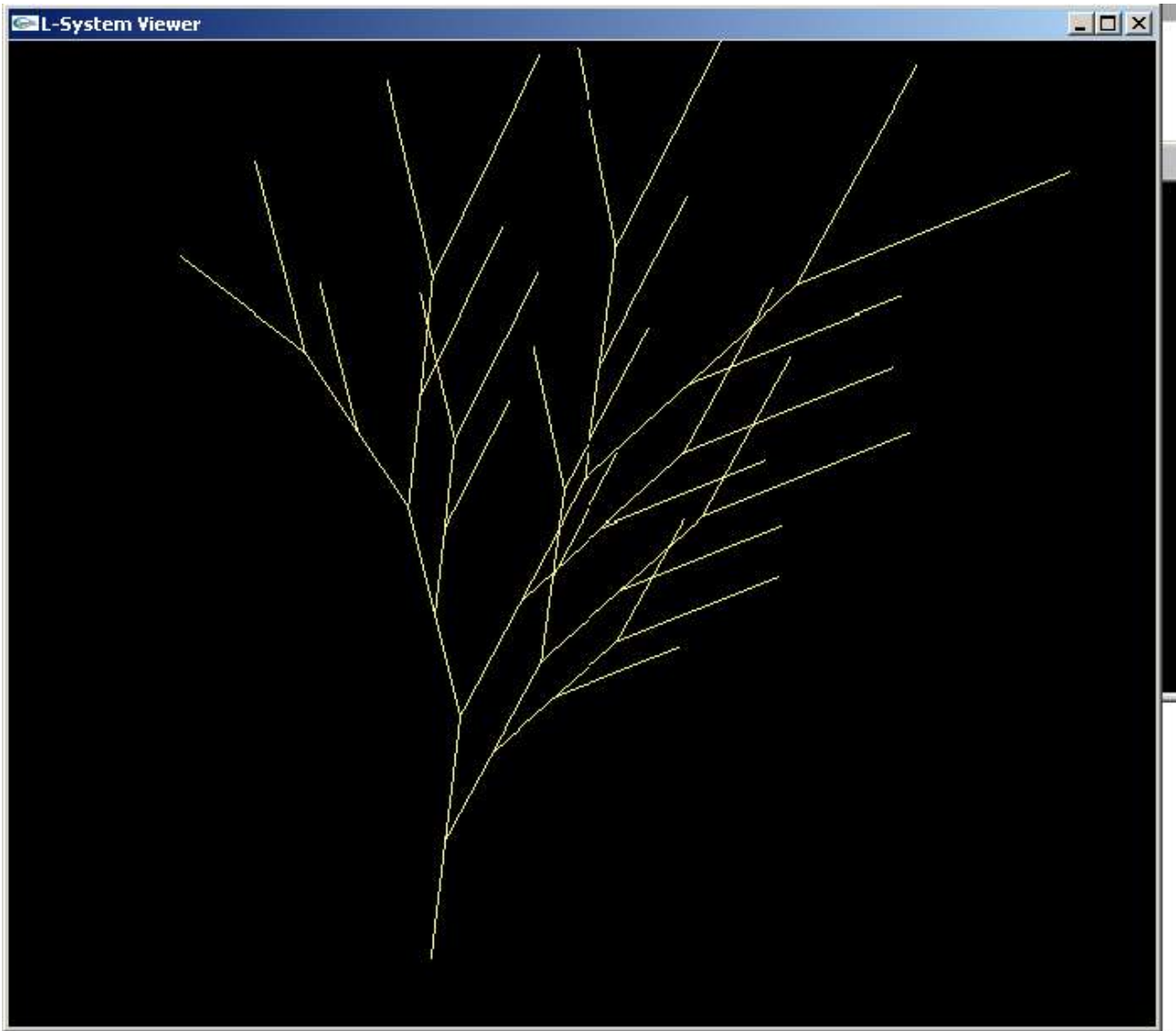


Illustration 3-Screenshot L-System Viewer