Einführung in die Informatik
Übung zu Modellierung und Simulation

15. Oktober, Herbst 2009

Diese Übung muss zu Beginn der Übungsstunde am 29. Oktober abgegeben werden. Beachten Sie genau die Angaben zum Material, das abgegeben werden soll. Die Übung kann in Zweiergruppen bearbeitet werden. Vergessen Sie nicht, Ihre Namen und Matrikelnummern in der Abgabe zu vermerken.

Praktische Aufgaben

In dieser Übung arbeiten Sie mit *Game Maker*, einem Werkzeug, um einfach und mit wenig Programmieraufwand Computerspiele zu erstellen. Allerdings verwenden wir es in dieser Übung, um mit Simulationen zu experimentieren. Game Maker bietet eine einfache Programmiersprache, die wir in dieser Übung verwenden. Bitte beziehen Sie das Programm frei von dieser Web Page: *http://www.yoyogames.com/gamemaker*. Beachten Sie, dass es nur auf dem Windows Betriebssystem läuft.

Game Maker bietet umfangreiche Dokumentation im "Help" Menu des Programms. Um sich mit den Grundkonzepten von Game Maker bekannt zu machen, empfiehlt es sich, zuallererst dieses Tutorial zu bearbeiten: *http://www.yoyogames.com/downloads/tutorials/first.zip*. Die folgenden Erklärungen gehen davon aus, dass Sie die Game Maker Begriffe *Objects* und *Events* in diesem Tutorial kennengelernt haben.

Die Übung benutzt die *Game Maker Language (GML)* Programmiersprache, welche ebenfalls im Hilfemenu dokumentiert ist. Die Syntax von GML ist sehr ähnlich wie zum Beispiel Java. Einige der wichtigsten Unterschiede und Ähnlichkeiten sind:

- Variablen müssen nicht deklariert werden. Variablen gelten als definiert, nachdem ihnen zum ersten Mal ein Wert zugewiesen wird. Bei der ersten Zuweisung zu einem Variablennamen muss kein Variablentyp angegeben werden.

- Felder (Arrays) werden automatisch alloziert. Beim ersten Schreibzugriff auf ein Element wird das Feld entsprechend vergrössert, falls dies nötig ist. Elemente von mehrdimensionalen Feldern werden gemäss dem zwei-dimensionalen Beispiel *a[i,j]* indiziert, wobei *a* der Name des Feldes ist, und *i, j* die Indizes.

- Klassen werden nicht direkt in GML deklariert. Das Konzept von Klassen entspricht den *Objects*, welche man über das grafische Interface konstruiert. Instanzen werden in GML mit der eingebauten Funktion *instance_create* erzeugt. Wie in Java greift man mit dem Operator "." auf Variablen einer Instanz zu.

- Die *Events*, die man für jedes Objekt definieren kann, sind die Methoden der Objekte. Die Behandlung der Events kann durch GML Code definiert, oder mit Hilfe des grafischen Interfaces zusammengebaut werden.
- Game Maker durchläuft ein Spiel Schritt für Schritt. In jedem Schritt werden für alle Instanzen diejenigen Events behandelt, die auf die Instanz zutreffen. Die Events *Step* und *Draw* werden in jedem Schritt immer behandelt.

Für die folgenden Aufgaben stellen wir auf der Web Page der Vorlesung Game Maker Projekte als Ausgangslage zur Verfügung. Wir empfehlen, dass Sie sich mit Hilfe dieser Beispiele einarbeiten. Zögern Sie nicht, Fragen und Probleme auf dem Webboard und in den Poolstunden aufzubringen.

1. Zwei-dimensionale Wärmeleitung

In dieser Aufgabe geht es darum, die zwei-dimensionale Wärmegleichung zu simulieren. Verwenden Sie als Ausgangslage das Game Maker Projekt *heat.gmk*. Die Simulation basiert auf einem einzigen Objekt *main*. Dieses Objekt speichert ein zwei-dimensionales Feld, welches die Wärmeverteilung auf einem Gitter darstellt. Die Veränderung des Gitters in jedem Simulationsschritt wird im Event *Step* berechnet. Die Darstellung des Gitters erfolgt im Event *Draw*, der am Schluss jedes Schritts abgearbeitet wird.

Aufgaben

Modifizieren Sie den GML Code im *Step* Event so, dass die Entwicklung der Wärmeverteilung mit dem finiten Differenzenverfahren wie in der Vorlesung berechnet wird. Um diese Änderungen vorzunehmen, expandieren Sie die Liste der *Objects*, öffnen Sie das *main* Objekt, wählen Sie den Event *step* und öffnen Sie den GML Programmtext durch Doppelklick auf *Execute a piece of code*.

Beachten Sie, dass die Formel zur Berechnung jedes Wertes immer auf alle vier Nachbarn zugreift. Das bedeutet, dass in einer naiven Implementation an den Rändern auf nicht existierende Daten zugegriffen wird. Um dieses Problem zu lösen, müssen Sie zusätzliche *Randbedingungen* formulieren. Implementieren Sie die folgenden zwei Varianten:

- *Periodische Randbedingungen:* Man stellt sich vor, das zwei-dimensionale Gitter wiederhole sich periodisch. Anschaulich heisst das, dass das nächste Element links vom linken Rand dem Element links vom rechten Rand entspricht. Angenommen die Indizes laufen von 0 bis $n-1$, dann wird also der Index -1 auf den Index $n-1$ abgebildet. Umgekehrt entspricht der Index n, der ausserhalb des Feldes liegt, dem Index 0. Beachten Sie, dass die periodische Berechnung der Indizes mit Hilfe von Modulo-Operationen ohne Fallunterscheidungen erreicht werden kann.
- *Dirichlet Randbedingungen:* Man nimmt an, die Nachbarelemente ausserhalb des Bereichs hätten einen festen Wert. Wählen Sie den Wert null für Ihre Experimente.

Experimentieren Sie mit beiden Varianten, die Randbedinungen zu implementieren. Welche Unterschiede beobachten Sie?

Untersuchen Sie weiter den Einfluss des Parameters $r = k/h^2$, der in der Diskretisierung mit finiten Differenzen vorkommt. Beschreiben Sie Ihre Beobachtungen.

Abgabe

Abzugeben sind je eine Game Maker Datei für die Implementation der zwei verschiedenen Randbedingungen. Ausserdem sollen die Beobachtungen in Worten in einer Textdatei aufgeschrieben werden.

2. Schafherde

In dieser Aufgabe simulieren Sie eine weidende Schafherde. Verwenden Sie als Ausganslage die Game Maker Datei *sheep.gmk*. Die zwei wichtigsten Objekte der vorgegebenen Implementation sind die Schafe *sheep* und Grasflächen *grass*. Studieren Sie das Verhalten dieser Objekte, indem Sie sich ihre Events anschauen. Beachten Sie, dass die Verteilung von Schafen und Gras zu Beginn des Spiels im Raum *room0* spezifiziert ist.

Aufgaben

Die vorgegebene Simulation ist "unrealistisch", weil die Schafe nur Grass fressen, sich aber weder vermehren noch sterben. Erweitern Sie das Verhalten der Schafe folgendermassen:

- Jedes Schaf speichert eine Variable *full*, der anzeigt wie stark es mit Gras vollgefressen ist. Die Variable soll bei der Erzeugung eines Schafs, das heisst im Event *Create*, auf einen bestimmten Wert gesetzt werden. Der Wert wird in jedem Schritt erhöht, wenn das Schaf Gras findet. Er wird in jedem Schritt um einen gewissen Futterverbrauch verkleinert, wenn kein Futter gefunden wird.

- Wenn ein Schaf genügend Nahrungsreserven hat, das heisst *full* überschreitet einen gewissen Schwellwert, dann pflanzt es sich fort. Sie können dazu mit der GML Funktion *instance_create()* eine neue Instanz eines Schafes erzeugen. Gleichzeitig soll der Wert *full* des "Mutterschafs" wieder unter den Schwellwert reduziert werden.

- Wenn ein Schaf über längere Zeit kein Futter findet, dann wird sein *full* Wert unter null sinken. In diesem Fall stirbt das Schaf. Sie löschen die Instanz eines Schafs mit der GML Funktion *instance_destroy()*.

Die folgende Liste enthält eine Übersicht über die Parameter, die das Verhalten der Schafe bestimmen:

- Der Zuwachs von *full*, wenn ein Stück Gras gefunden wird.

- Die Reduktion von *full*, wenn kein Gras gefunden wird.

- Der Schwellwert von *full*, der eine Geburt auslöst.

- Der Wert von *full* der Mutter nach der Geburt.

- Der Wert von *full* des Neugeborenen bei der Geburt.

Experimentieren Sie mit verschiedenen Werten für diese Parameter. Analysieren Sie jeweils die gesammelten Daten (Anzahl Schafe und Grasflächen zu jedem Zeitschritt), die in der Datei *sheep_data.txt* gespeichert werden, indem Sie die Daten als Kurven plotten. Beschreiben Sie Ihre Beobachtungen in ein paar Worten.

Bonusaufgabe: Erweitern Sie die Simulation um zusätzliche Effekte Ihrer Wahl. Sie könnten weitere Tierarten, wie zum Beispiel Wölfe, einführen, und Sie könnten weitere Verhaltensweisen implementieren. Sie könnten auch Benutzerinteraktion ermöglichen, oder die Simulation optisch verbessern, indem Sie eigene Sprites erstellen. Wir werden die beste Abgabe mit einem kleinen Preis belohnen!

Abgabe

Abzugeben ist die Game Maker Datei je für die erweiterte Simulation der Schafherde. Die analysierten Daten in Form von Kurven müssen als Dateien im PDF Format abgegeben werden. Für alle Kurven soll ersichtlich sein, mit welchen Simulationsparametern sie erzeugt wurden. Ausserdem sollen die Beobachtungen in Worten in einer Textdatei aufgeschrieben werden. Für die Bonusaufgabe sollen Sie einen kurzen Text abgeben, der Ihre Erweiterungen beschreibt, sowie die entsprechende Game Maker Datei.