

# Gothy, eine interaktive 3D-Comic-Figur

Studienarbeit

vorgelegt von:

Janet Seifert

Universität Koblenz-Landau  
Institut für Computervisualistik  
Arbeitsgruppe Computergrafik

Betreuer und Prüfer: Prof. Dr. Stefan Müller

Koblenz, den 28. Juni 2005

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>2</b>
1.1	Motivation der Arbeit . . . . .	2
1.2	Ziel der Arbeit . . . . .	3
<b>2</b>	<b>Entwicklung in Cinema4D</b>	<b>5</b>
2.1	Modellierung . . . . .	5
2.2	Animation . . . . .	7
2.2.1	Augen . . . . .	7
2.2.2	Ohren . . . . .	11
2.2.3	Rumpf . . . . .	12
2.2.4	Arme . . . . .	14
2.2.5	Kopf . . . . .	15
<b>3</b>	<b>Präsentation in Flash</b>	<b>16</b>
3.1	Erste Schritte . . . . .	16
3.2	Aufbau der Figur . . . . .	17
3.3	Laufen . . . . .	18
3.4	Leben . . . . .	21
3.4.1	Augen . . . . .	21
3.4.2	Kopf . . . . .	22
3.4.3	Hintergrundgeschehen . . . . .	22
3.5	Intro . . . . .	23
<b>4</b>	<b>Fazit und Ausblick</b>	<b>25</b>

# Kapitel 1

## Einleitung

### 1.1 Motivation der Arbeit

Mit der ersten Skizze fängt alles an. Ein kleines Schaf. Nicht irgendein beliebiges Schaf, sondern eines mit Persönlichkeit, Charakter und Geschmack. Es mag alten Wave von The Cure, Bauhaus und The Mission. Es trägt ausschliesslich schwarze Mäntelchen mit hohem Stehkragen, liebt Fledermäuse und alte Dinge. Es ist wirklich ein ausgeglichenes Schaf und nicht so leicht aus der Ruhe zu bringen. Wenn es ausreichend Tee und Kekse gibt, ist die Welt für Gothy in Ordnung.

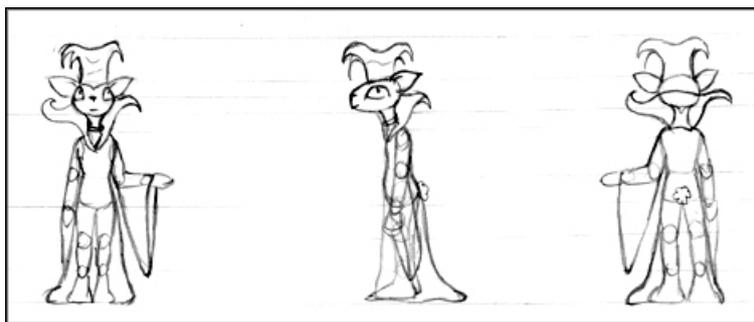


Abbildung 1.1: Gothy Entwurf

Dieses Charakterschaf taucht immer öfter im Skizzenbuch auf und wird schließlich zu einer der beiden Galionsfiguren der Homepage [www.vecona.de](http://www.vecona.de). Dort wird selbstentworfenene, extravagante Kleidung in einer Comic-Umgebung präsentiert. Die Figur findet man als 2D-Zeichnung in verschiedenen Räumen und wird von ihr durch die Seite geführt.



Abbildung 1.2: Gothy in 2D

Der Besucher kann mit ihr interagieren, indem er bestimmte Bereiche einer Seite mit dem Mauszeiger überfährt. Die Reaktion der Figur beschränkt sich dabei auf eingblendete Sprechblasen mit Hinweisen.

## 1.2 Ziel der Arbeit

Ziel der Arbeit ist, diese Interaktion auszuweiten. Der Benutzer soll in der Lage sein, auch die Figur selbst zu bewegen und ihr somit Leben einzuhauchen. Diese Funktion soll auf eine Seite beschränkt sein und nicht der Navigation dienen.

Die Darstellung der Szene erfolgt in 2D. Die einzelnen Bewegungen werden als Animation gespeichert, in der fertigen Applikation durch Mausklick aktiviert und abgespielt.

Die Modellierung und Animation erfolgt mit MAXON Cinema4D Release 8, die Präsentation wird mit Macromedia Flash MX 2004 erstellt.

Auch wenn das Projekt mehrmals diesen Zyklus durchläuft, werden diese Schleifen zu einem universalen Durchlauf zusammengefasst.

# Kapitel 2

## Entwicklung in Cinema4D

### 2.1 Modellierung

Die Figur ist aus einfachen geometrischen Objekten, wie Würfeln und Kugeln aufgebaut. Ihre endgültige Form erhalten die Elemente durch Verschiebung, Skalierung und Drehung der einzelnen Flächen, Kanten und Punkte.

Zum Modellieren der Frisur wird Matrix-Extrude verwendet. Dafür werden einzelne Polygone auf der Haaroberfläche ausgewählt und abgetrennt. Durch variieren der Einstellungen werden wirre Strähnen erzeugt und es entsteht Gothys zerzaustes "Vogelnest".

Die einzelnen Körperteile sind aus mehreren Elementen zusammengesetzt. Der Kopf besteht aus der Grundform, Augen, Nase und Haaren, die Arme aus Oberarm, Unterarm und Hand. Damit keine harten Kanten entstehen und eine geringe Anzahl von Stützpunkten ausreicht, werden sie Hypernurbs-Objekten untergeordnet. So werden die Objektoberflächen innerhalb dieses "Containers" interpoliert und können später separat bearbeitet werden. Das heisst, wenn Punkte innerhalb dieses Hypernurbs-Objektes verändert werden, bleiben alle Punkte ausserhalb dieses Objekt-Zweiges unverändert. Das soll sich beim Animieren als sinnvoll erweisen.

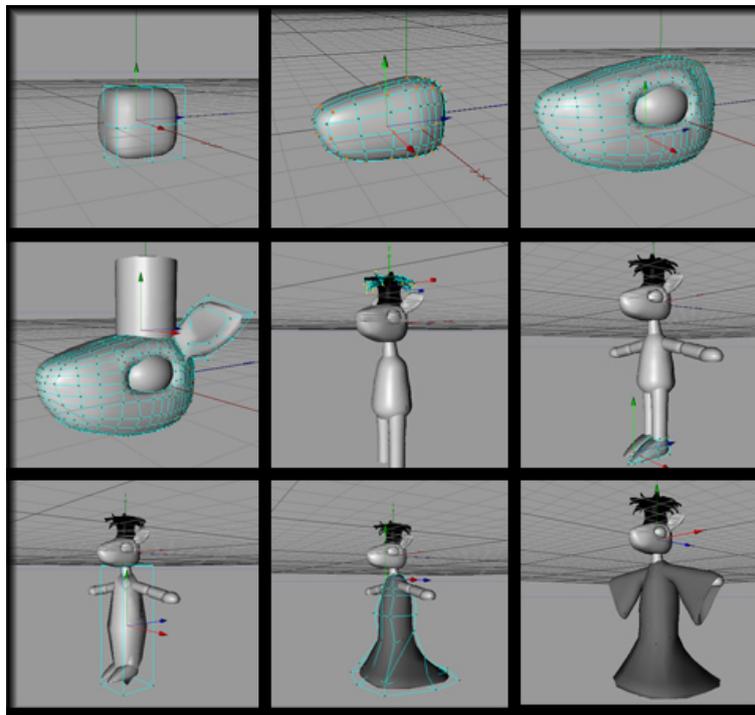


Abbildung 2.1: Modellierung von Gothy in Cinema4D

Am Ende der Modellierung werden der Figur Texturen zugewiesen. Der Mantel erhält einen leicht strukturierten Jeansstoff, die Haut ein helles Schaffell. Die Augen sind weiss und leicht glänzend, die Haare schwarz und matt.

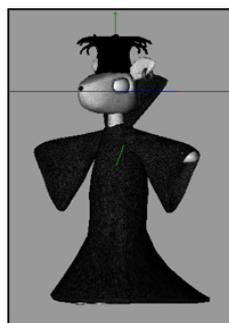


Abbildung 2.2: Das fertige Modell mit Texturen

## 2.2 Animation

### 2.2.1 Augen

Die erste Animation ist der Lidschlag. Dazu wird ein weiteres Objekt in Form des Auges erstellt, etwas vergrössert und geteilt. Die beiden Halbschalen werden so angepasst, dass sie auch bei Rotation das Auge nicht durchdringen. Diese Arbeit erfordert etwas Fingerspitzengefühl, da der Augapfel keine Kugel, sondern eiförmig ist. Ausserdem soll die Linie, an dem die Lider zusammen treffen, im unteren Teil des Auges und nicht in der Mitte liegen.

In der Zeitleiste sind drei Keyframes nötig, um das Auge einmal zu schliessen und wieder zu öffnen. Der Startframe mit offenem Auge, ein mittlerer Frame mit geschlossenem Auge, das letzte Bild muss wieder mit dem Startbild identisch sein. Ein Klick auf "play" und das Schaf blinzelt, die Zwischenbilder werden automatisch erzeugt.

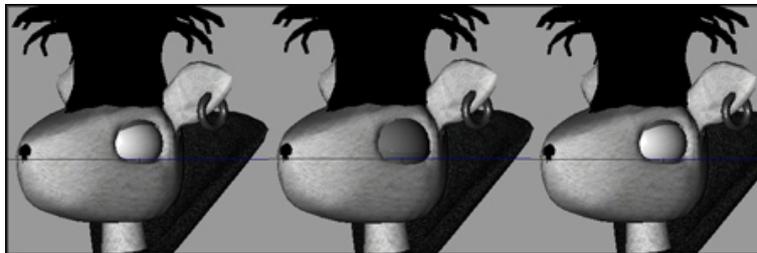


Abbildung 2.3: Lidschlag Keyframes

Der erste Test wird als \*.avi-Datei gespeichert. Das Video wird gerendert und zu jedem Einzelbild eine Alpha-Maske erstellt. Das sieht zwar zuerst wie das gewünschte Ergebnis aus, lässt sich aber nur schwer weiter verwenden. Für die Arbeit in Flash sind Einzelbilder sinnvoller, weil einzelne Teilsequenzen in verschiedenen Kombinationen aneinandergesetzt werden sollen.

Trotzdem muss das Schaf freigestellt sein, damit es sich ohne störenden Rand in die Szene einpasst. Es muss also in jedem Bild der Alpha-Kanal enthalten sein.

Eine Lösung wäre das \*.targa-Format, doch dieses lässt sich nicht in Flash importieren. Eine alternative Lösung liefert das \*.psd-Format, denn so kann man in Photoshop eine Aktion schreiben, die den Hintergrund des Bildes entfernt und es mit Transparenz als \*.png-Datei speichert. Dabei lässt sich das Bild sogar noch trimmen, das heisst, die überstehende transparente Bildfläche wird entfernt. Mit diesen \*.png-Sequenzen ist Flash einverstanden.

Die entstandenen Dateien sind relativ gross und für die Anwendung im Internet nicht optimal. Die Idee ist, die Figur nicht für jede Aktion komplett neu zu rendern, sondern einzelne Körperteile bei Bedarf auszutauschen. Wenn nur ein Lidschlag ausgeführt werden soll, muss also nur das Auge neu in eine statische Figur eingefügt werden.

Wie aber rendert man die einzelnen Objekte, ohne dass es zu falschen Überdeckungen, fehlenden oder doppelten Elementen kommt? Einzelne Objekte auszuschalten, bringt nicht das gewünschte Ergebnis, denn vorher verdeckte Elemente liegen plötzlich frei, z.B. der Halsansatz beim Weglassen des Rumpfes.

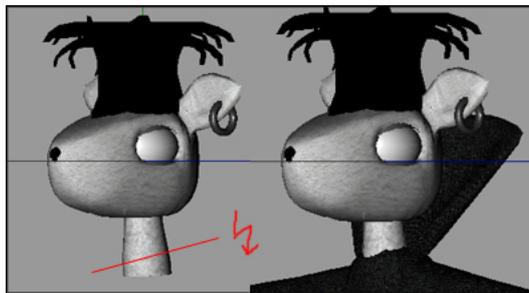


Abbildung 2.4: Fehlende Überdeckung beim Rendern einzelner Elemente

Ein Trick aus 3D Studio Max ist das Alpha-Material ohne Farbe. Dieses Material sorgt dafür, dass die damit belegten Objekte beim Rendern nicht mehr zu sehen sind und die von ihm verdeckten Teile trotzdem korrekt entfernt werden. Die Tücke steckt im Detail, denn auch wenn diese Teile im gerenderten Bild unsichtbar sind, tauchen sie in der Alpha-Maske wieder auf. Ist im fertigen Bild nur der Körper zu sehen, bleibt im Alpha-Layer die ganze Figur bestehen, so dass der Rumpf nicht korrekt freigestellt werden kann. Somit ist dieses Verfahren in Cinema4D nicht anwendbar.

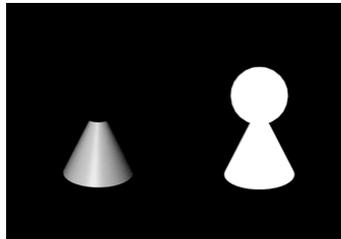


Abbildung 2.5: Fehlende Elemente tauchen im Alphakanal wieder auf

Ein weiterer Ansatz ist die Verwendung von Boolean-Objekten. Hier werden zwei Objekte durch Subtraktion verknüpft. Das Ergebnis entspricht leider auch nicht den Anforderungen. Objekt 1 wird zwar nicht mehr gezeichnet, doch dort, wo das Objekt 2 abgeschnitten wird, entsteht eine 3D-Schnittfläche. Diese Schnittfläche führt beim Zusammenfügen der einzelnen Ebenen wieder zu Überdeckungen.

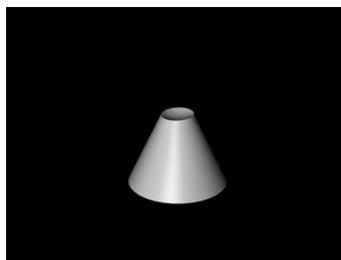


Abbildung 2.6: Bei Boolean-Objekten entstehen unerwünschte Schnittflächen

Die Lösung heisst Multipass-Rendering. Dazu wird jedem Objekt ein Render-Tag mit eindeutigen Namen und Objekt-Kanal zugeordnet. Dieser Objekt-Kanal kann in den Render-Voreinstellungen unter "Multipass-Rendering" ausgewählt werden. Ausgegeben wird zum einem das übliche gerenderte Bild mit vollem Alpha-Kanal, darüber hinaus noch ein zweites Bild mit dem gewünschten Alpha-Kanal, allein mit den gewählten, sichtbaren Objekten.

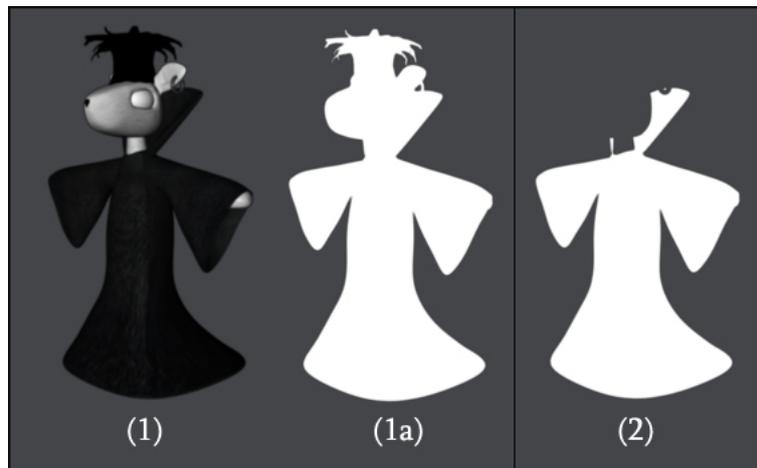


Abbildung 2.7: Gerendertes Bild (1) mit Alpha-Kanal (1a) und separates Multipass-Bild mit Alpha-Kanal (2)

Beide Bilder können nun per Hand in einem Grafik-Programm verarbeitet werden. Der Alpha-Kanal aus dem Multipass-Bild wird als zusätzlicher Kanal in das gerenderte Bild eingefügt. Die unerwünschten Objekte können damit so herausgeschnitten werden, dass Überdeckungen erhalten bleiben, auch wenn die verdeckenden Elemente selbst nicht mehr zu sehen sind.

Um diese mühselige Handarbeit abzukürzen, lässt sich der Vorgang als Aktion speichern, so dass nur noch der Dateiname des Multipass-Bildes und der gewünschte Speicherort angegeben werden muss.



Abbildung 2.8: Ergebnisbild nach Entfernen des Hintergrunds

### 2.2.2 Ohren

Als nächstes soll das Schaf mit den Ohren wackeln. Sie sind Bestandteil des Symmetrie-Objektes Kopf. Die Ohringe dagegen nicht, denn sie haben unterschiedliche Formen und sind aus mehreren Teilen zusammengesetzt. Durch die fehlende Symmetrie können sie also nicht Unterobjekte des Kopfes sein und müssen einzeln behandelt werden. Alle einzelnen Teile aber völlig synchron zu animieren ist sehr aufwändig. Aus diesem Grund kommen alle Elemente in ein gemeinsames FFD-Objekt.

Es ist ein Punktgitter-Objekt, das beliebig aufgeteilt und verformt werden kann. Das Verschieben der einzelnen Punkte verändert synchron alle enthaltenen Objekte. Allerdings nur, wenn das FFD-Objekt noch eine übergeordnete Instanz besitzt, sonst hat das Verschieben des Punktgitters keine Auswirkung auf die enthaltenen Elemente. Ein übergeordnetes Null-Objekt erfüllt diese Anforderung.

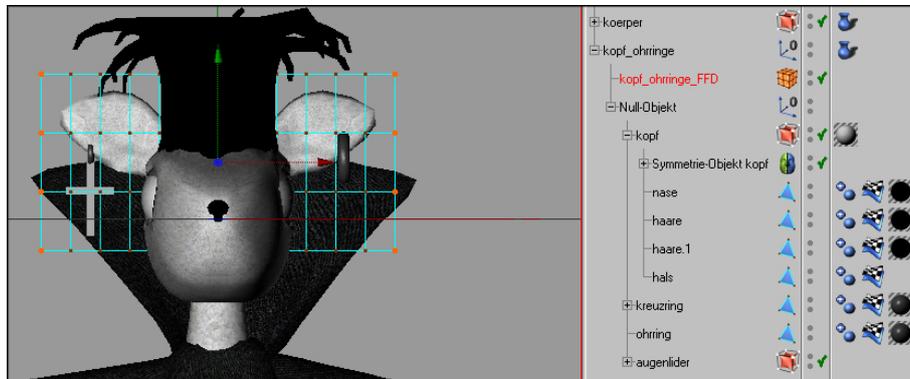


Abbildung 2.9: Mit einem FFD-Objekt können Kopf und Ohrringe synchron verformt werden

### 2.2.3 Rumpf

Das Laufen wird auf ähnliche Weise simuliert wie das Ohrwackeln. Dazu wird der untere Teil des Mantels mit dem Körper in ein FFD-Objekt gesteckt und ein Walkcycle erstellt. Das ist eine vollständige Schrittsequenz, bei der Start- und Endbild genau übereinstimmen. Beliebiger oft wiederholt, entsteht aus diesem Zyklus eine Laufbewegung.

Um einen Schritt anzudeuten, werden die Punkte so verschoben, dass jeweils ein Bein nach vorn und ein Bein nach hinten steht. In der Vorschau sieht das zwar aus wie erwartet, doch nach dem Rendern entstehen an der Ansatzstelle der Beine unschöne Knicke.

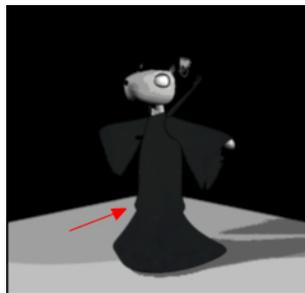


Abbildung 2.10: Laufartefakte

Diese Artefakte werden beseitigt, indem die ganze Figur einem grösseren Verformobjekt übergeben wird. Das hat den Vorteil, dass sich nicht nur die Beininstellung verändert, sondern der ganze Körper mitbewegt wird. Es wird eine zusätzliche Drehung des Unterkörpers eingefügt und der Oberkörper in die Gegenrichtung gedreht. Schon sieht der Bewegungsablauf deutlich realistischer aus. Auch die Artefakte sind verschwunden.

Die Laufbewegung wird für das Drehen der Figur weiter verwendet, das Schaf läuft dabei einen Halbkreis auf der Stelle. Die Keyframes des FFD-Objektes bleiben erhalten, zusätzlich wird noch die Kamera animiert. Dem Betrachter erscheint das so, als ob die Figur selbst gedreht wird.

Die Animation erfolgt mit Hilfe von Kamera-Keyframes, die einen Pfad um die Figur bilden. Die Endposition ist dabei die an der Z-Achse gespiegelte Startposition.

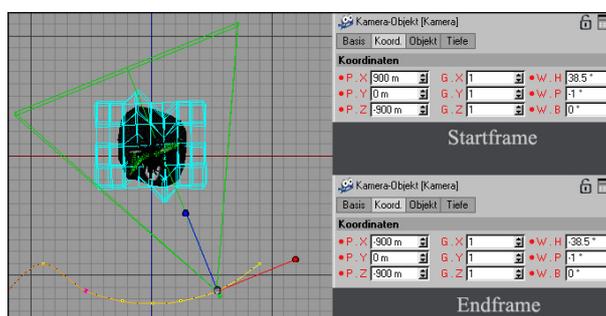


Abbildung 2.11: Die Drehung der Figur erfolgt durch Animation der Kamera

### 2.2.4 Arme

Anschliessend werden die Arme animiert. Hier würde ein Verformobjekt nicht ausreichen, denn die Arme sind nicht, wie die Beine, unter weit fallendem Stoff versteckt. Sie werden zum realistischen Bewegen mit Bones versehen und mit der Inversen Kinematik-Funktion verformt.

Die Knochenkette besteht pro Arm aus fünf Elementen: Oberarm, Unterarm, Hand und zwei Extra-Knochen zum Verformen der weiten Ärmel. Der Wirkungsbereich der einzelnen Knochen kann festgelegt werden, so dass der Arm beim Verschieben der Hand nicht ziellos mitschaukelt, sondern durch einzelne Ankerpunkte manuell gesteuert werden kann.

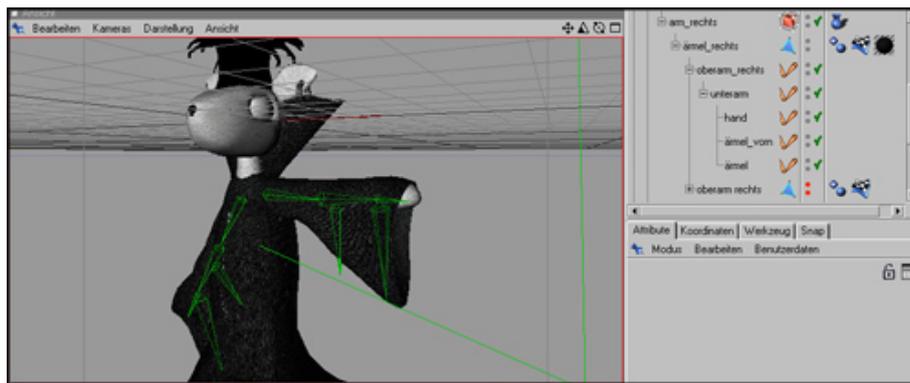


Abbildung 2.12: Arme mit Bones

Für das Schaf sind mehrere Armbewegungen, wie Zeigen und Winken vorgesehen. Um das Laufen realistischer wirken zu lassen, wird auch hier noch das Mitschwingen der Arme ergänzt.

Die Arbeit mit Bones bringt ein neues Problem mit sich. Der Arbeitsspeicher ist unzureichend und die Arme werden beim Rendern einfach weggelassen. Die Ursache liegt in der Anzahl der Hypernurbs-Unterteilungen, denn diese wird durch die enthaltenen Bones-Objekte automatisch erhöht. Die ehemals grob unterteilten Flächen verwandeln sich plötzlich und ohne Zutun in ein feinmaschiges Netz. Stellt man die Anzahl der Unterteilungen manuell kleiner, funktioniert auch das Rendern wieder.

### **2.2.5 Kopf**

Das Bewegen des Kopfes ist deutlich unkomplizierter und erfolgt durch Drehen des Hypernurbs-Objektes. Der Kopf soll alle Positionen, wie links, rechts, oben, unten und geradeaus ansteuern können, insgesamt also neun verschiedene Stellungen. Nach einem kurzen Innehalten soll er sich wieder zur Ausgangsposition zurück bewegen. Mehrfach genutzte Wege werden nur einmal animiert, die verschiedenen Bewegungen werden später aus mehreren Einzelsequenzen zusammengesetzt.

# Kapitel 3

## Präsentation in Flash

### 3.1 Erste Schritte

Die 2D-Oberfläche, in der die Szene stattfindet, wird in Macromedia Flash erstellt. Zur Übung wird erst eine animierte Zeichnung angefertigt. Die einzelnen Bilder werden in \*.png-Format gespeichert. So kann man mit transparenten Bereichen arbeiten und erhält einen ersten Eindruck der freigestellten Figur im Raum. Die Szene beginnt mit dem unbewegten Startbild. Bei Mausklick auf die Figur wird die Animation bis zum nächsten Klick als Schleife abgespielt.



Abbildung 3.1: Erste Test-Animation

## 3.2 Aufbau der Figur

Der zweite Schritt ist der Test mit der gerenderten Sequenz. Auch hier werden die einzelnen Bilder als \*.png-Sequenz importiert und auch die Aktionen bleiben dieselben. Das Problem ist die Grösse der entstandenen Datei, denn die gerenderten Ausgangsbilder sind deutlich grösser als die Zeichnungen.

Die Idee, eine starre Figur im Hintergrund zu lassen und jeweils nur die einzelnen bewegten Körperteile zu verändern, bringt die geforderte Speicherplatzeinsparung. Die Figur wird durch Abfragen aus Einzelementen zusammengesetzt, die auf verschiedenen Ebenen angeordnet sind und simultan abgespielt werden. Hinzu kommt, dass es zwei Ansichten vom Schaf geben muss, da es nicht nur in eine Richtung laufen soll. Diese Bibliothek von Elementen muss also auch noch in gespiegelter Form angelegt werden.

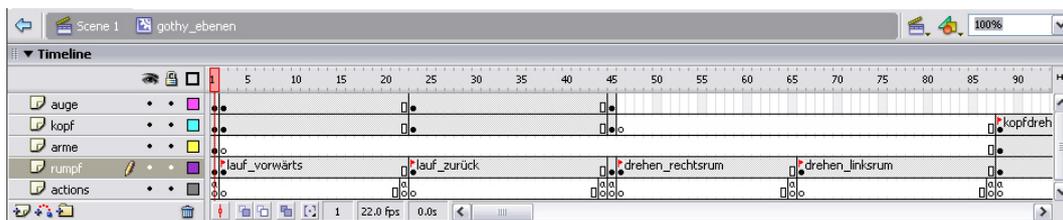


Abbildung 3.2: Movieclip "gothy" besteht aus mehreren Ebenen

Zuerst wird die stehende Figur in Körper, Kopf und Auge unterteilt. Auf diese Weise kann das Schaf gleichzeitig zwinkern und mit den Ohren wackeln, denn diese Animationen befinden sich auf unterschiedlichen Ebenen und können somit unabhängig voneinander angesteuert werden. Der Rumpf ist hier noch statisch. Im nächsten Schritt soll auch dieser Teil separat bewegt werden können. Das heisst, Ohrenwackeln und Blinzeln sind auch im Laufen möglich.

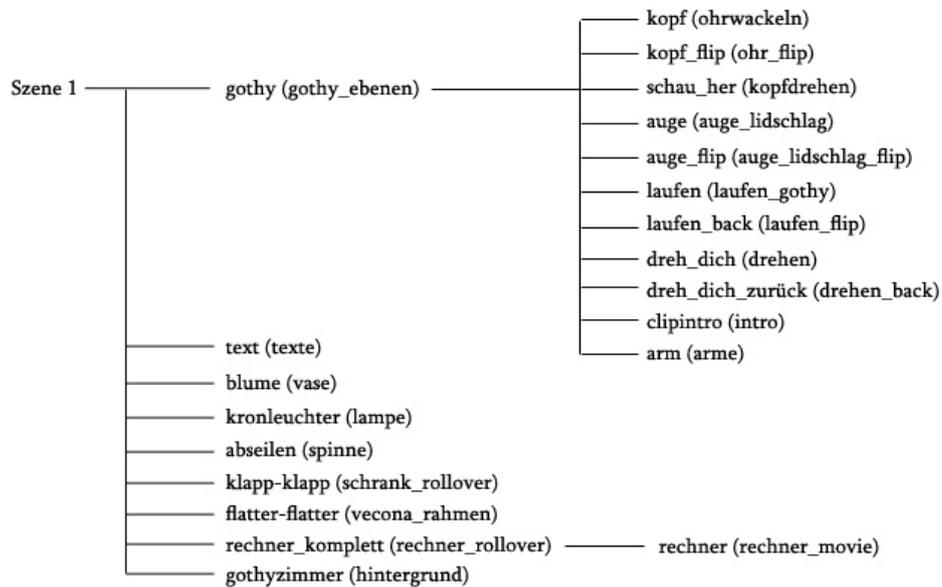


Abbildung 3.3: Movieclip-Hierarchie in Flash

### 3.3 Laufen

Um die Figur in der Szene zu bewegen, wird ein Motion-Tween eingefügt, der nach dem Festlegen von Start- und Endposition automatisch Zwischenbilder generiert.

Jetzt schwebt das Schaf noch unbewegt von rechts nach links. Auge und Ohr kann man schon per Mausklick bewegen, nur die eigentliche Laufbewegung fehlt. Diese wird im "rumpf" eingefügt und über Aktionen in "gothy" gesteuert.

Um diese Lösung etwas eleganter zu gestalten, wird nun das Tweening aus "gothy" direkt in die Hauptszene verlegt. Auf diese Weise kann man den kompletten Bewegungsablauf, bestehend aus Tweening und Laufbewegung von dort aus steuern. Man muss nicht in untergeordnete Movieclips klicken, um Änderungen am Pfad oder der Laufgeschwindigkeit vorzunehmen.

Zum Beispiel soll eine dritte Position hinzu kommen, die die Figur ansteuern kann. Um alle Verbindungen abzudecken, sind sechs vormodellierte Bewegungsabläufe nötig. Damit die Figur auf den drei Wegen nach rechts nicht rückwärts laufen muss, wird noch eine gespiegelte Laufbewegung in "gothy" eingefügt.

Das unschöne Umschalten der Bilder wird durch Drehungen ausgeglichen. Dabei wird die komplette Figur ersetzt und die Mausfunktionen für Auge und Ohr stehen für kurze Zeit nicht zur Verfügung. Dem Benutzer sollte das nicht auffallen, denn diese Szene ist nur eine halbe Sekunde lang.

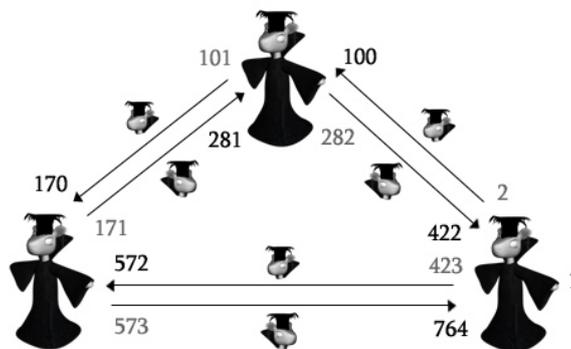


Abbildung 3.4: Pfade mit Frame-Nummern und Richtungsangaben

Die einzelnen Bewegungsabläufe werden durch Anklicken verschiedener Objekte im Raum ausgelöst. Man erkennt sie daran, dass sie beim Überfahren mit dem Mauszeiger reagieren. Wählt man durch Anklicken ein Möbelstück im Raum aus, läuft die Figur dorthin. Steht sie bereits am gewählten Objekt, findet ein anderes Event statt.

Dann werden eine Sprechblase und ein Menü eingeblendet, mit deren Hilfe Gothy erklärt, welche Funktion die Gegenstände in der eigentlichen Anwendung haben. So kann man sich vorstellen, wie die Figur später einmal als Hilfe-Funktion oder Agent fungiert, der dem Besucher einer Webseite Hilfeleistung gibt. Die eingeblendeten Felder verschwinden nach kurzer Zeit und können durch erneutes Anklicken wieder aufgerufen werden.

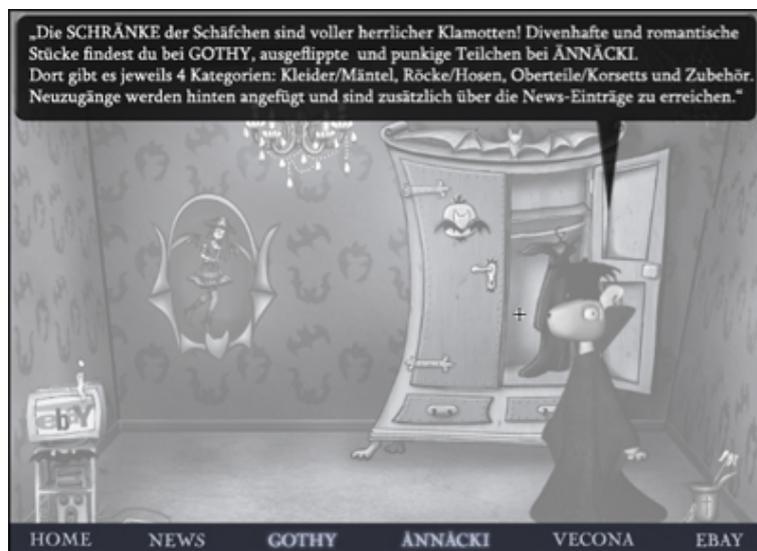


Abbildung 3.5: Sprechblase und Menü werden eingeblendet

## 3.4 Leben

### 3.4.1 Augen

Die stehende Figur würde viel lebendiger wirken, wenn die Augen ständig in Bewegung wären. Zusätzlich zum Lidschlag entstehen neun weitere Bilder mit verschiedenen Pupillen-Stellungen und werden in die Augen-Zeitleiste eingefügt. Diese einzelnen Frames werden nun in einer OnEnterFrame-Funktion abhängig von der Mausposition angesteuert. Anfangs wurde die Position in der Mitte des Auge nie erreicht, weil dafür die Mauskoordinate punktgenau mit der Augenkoordinate übereinstimmen musste. Um diesen Bereich zu vergrößern, wird ein Toleranzbereich eingefügt. Das Schaf schaut wie gewünscht der Maus hinterher.

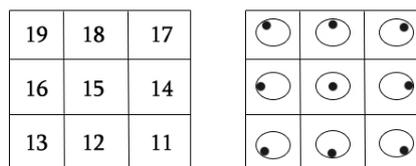


Abbildung 3.6: Für jede Augenposition wird ein Bild angefertigt

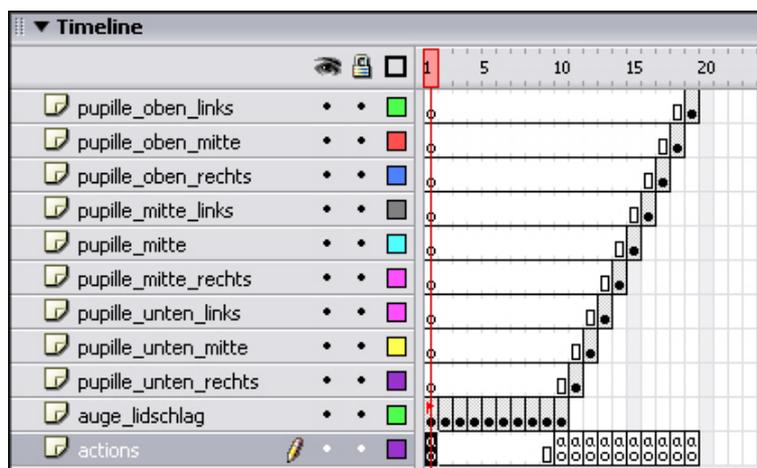


Abbildung 3.7: Umsetzung der Augenpositionen in Flash

### 3.4.2 Kopf

Klickt man mit der Maus in die Szene, soll sich der ganze Kopf dieser Position zudrehen. Das funktioniert nach demselben Prinzip wie bei der Pupillenstellung, nur dass die Funktion nicht direkt auf dem angesprochenen Objekt, dem Kopf, sondern auf dem Hintergrund des Bildes ausgeführt wird. Die Bewegung wird also von einem anderen Movieclip aus gesteuert.

Damit die Kopfbewegung natürlich wirkt, wird nicht der kürzeste Weg gewählt, sondern die Bewegung läuft über Zwischenstationen. Soll zum Beispiel der Kopf, ausgehend von der Startposition, nach rechts oben schauen, wandert er zuerst geradeaus, dann nach rechts und dort erst nach oben. Auf diesem Weg dreht er sich nach einem kurzen Verharren auch wieder zurück.

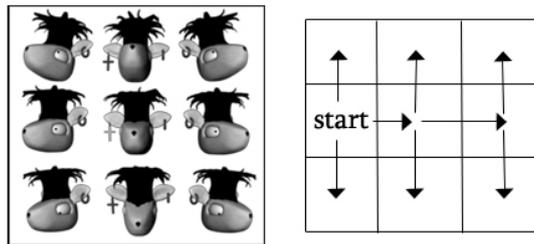


Abbildung 3.8: Der Kopf wird über Zwischenpositionen bewegt

### 3.4.3 Hintergrundgeschehen

Trotzdem wirkt die Szene noch etwas eintönig und steif. Ihre Atmosphäre erhält sie durch eine Random-Funktion, die in jedem einzelnen Frame ausgeführt wird. Einige kleine und nebensächliche Elemente im Raum werden durch eine Zufallszahl gesteuert und bewegt.

Eine Spinne seilt sich ab, die Blume wirft nach und nach ihre Blütenblätter ab, der Kronleuchter schaukelt oder das Schaf "lebt". Es zwinkert, wackelt mit den Ohren oder ... siehe da, es schaut her und winkt!

Zu jedem Element wird dafür ein eigener Movieclip erstellt. Die Steuerung erfolgt von der Hauptszene aus.

Damit der Ablauf nicht zu chaotisch wirkt, ist ein Puffer eingebaut, der Pausen zwischen den einzelnen Aktionen erzwingt.



Abbildung 3.9: Die vollständige Szene

### 3.5 Intro

Was jetzt noch fehlt, ist eine Eingangssequenz. Der Benutzer soll begrüßt werden und erfahren, wie er mit der Figur interagieren kann. Das Schaf stellt sich also kurz vor, winkt und führt mit einer Handbewegung und dem Hinweis mit der Maus zu klicken, in die Szene ein.

Damit der Benutzer sich auch wirklich angesprochen fühlt, wird das Schaf zu Beginn sehr nah gezeigt. Nach dem Winken läuft es nach hinten und wird ausgezoomt. Der Zoom wird bereits in 3D, durch Verändern der Kamera-Koordinaten, ausgeführt.

Mit Beginn der Laufbewegung kommt in Flash ein Motion-Tween hinzu. Anschliessend folgt eine Links-Drehung und Gothy steht in der Ausgangsposition. Die Random-Funktion setzt ein und ab jetzt ist die Interaktion mit der Figur möglich.



Abbildung 3.10: Begrüssung...



Abbildung 3.11: ... und Einführung in die Szene

# Kapitel 4

## Fazit und Ausblick

Das Ergebnis der Arbeit ist eine kleine abgeschlossene Umgebung, in der eine einzelne Figur agiert. Sie wirkt durch die zufälligen Aktionen lebendig und der Benutzer ist in der Lage, selbst durch Interaktion Reaktionen, wie zwinkern, winken oder laufen hervorzurufen.

Dieses Bewegungsarsenal ist beliebig erweiterbar. Das Schaf könnte Gegenstände aus dem Mantel holen und verwenden, sich hinsetzen, Dinge aufheben, also auch mit anderen Gegenständen der Szene interagieren.

Vorstellbar wäre es, sie als Agent auf einer Webseite oder in einer Software zu verwenden. Dort würde sie als Navigation dienen und dem Benutzer hilfreiche Tipps geben. Mit einer KI und Texterkennung versehen, könnte sie sogar Fragen beantworten.

Dem kleinen Schaf stehen alle Türen offen, doch vorerst erklärt es Veconas kleine Welt unter <http://www.uni-koblenz.de/~vecona/CG/gothy3d.html>.

# Literaturverzeichnis

- [1] Chris Debski. *Cinema 4D 8 Charakteranimation*. Galileo Press, 2003.  
ISBN: 3898423840
  
- [2] Robert Reinhardt, Jon Warren Lentz. *Flash 5 Bible*. John Wiley & Sons,  
2001. ISBN: 0764535153
  
- [3] *Cinema4D Release 9 Referenzhandbuch*. <http://www.maxon.de/>
  
- [4] *Cinema4D Forum*. <http://www.c4d-treff.de>