Runde Pfade

Ada Möhlmann, Neele Stoeckler

im Studiengang Mensch-Computer-Interaktion

23. Mai 2012

Universität Hamburg

Fachbereich Informatik

Arbeitsbereich WSV

Projekt

Animationswerkzeug zur Visualisierung sozialen Handelns

Dozenten

Dr. Carola Eschenbach

Prof. Dr. Christopher Habel

Felix Lindner

Inhaltsverzeichnis

1 Einleitung

Ob in der Industrie, in Krankenhäusern, im Museum oder im Service, Roboter werden in vielen Bereichen genutzt. Die Mensch-Roboter-Interaktion gewinnt zunehmend an Bedeutung. Gerade im häuslichen, sozialen Bereich, z.B. bei der Alten - oder Behindertenpflege, ist eine funktionierende Interaktion mit dem Menschen unabdingbar. Durch den demographischen Wandel und besonders durch den Anstieg der älteren Bevölkerung, die Hilfe benötigt, ergibt sich die Notwendigkeit, im Bereich der Home Assistance weiter zu forschen. Roboter sollen sich an dem sozialen Verhalten von Menschen orientieren und danach handeln. Sowohl Roboter als auch Menschen müssen das Verhalten des jeweils anderen verstehen und antizipieren können, um eine funktionierende Kommunikation zu erreichen. Für den Menschen wäre es wünschenswert, dass ein Roboter ihn nicht stört, sondern ihn in seiner Rolle als Assistent unterstützt. Ein Haushaltsroboter, der sich sehr schnell bewegt, um dadurch effizient seine eigentliche Aufgabe, Staubsaugen oder Ähnliches, korrekt auszuführen, kann leicht als aggressiv und nervös wahrgenommen werden. Da sich Menschen und Roboter die Umgebung teilen und sich oft im gleichen Raum aufhalten, ist es nötig, dass der Roboter sich adäquat verhält. Es gibt mehrere Möglichkeiten wie der Roboter Informationen, bspw. den aktuellen Status übermitteln kann, z.B. Sprache und Bewegung. Wir wollen im Folgenden jedoch nur auf die Bewegung als Modalität genauer eingehen.

Alltägliche Situationen wie Begegnungen auf dem Flur, das Betreten eines Raumes oder das Unterbrechen eines Gespräches wurden bereits in Studien behandelt. Saulnier, Sharlin und Greenberg [S. Greenberg, E. Sharlin, P. Saulnier, 2011] haben 2011 untersucht, wie die minimale nonverbale Unterbrechung eines Gesprächs durch Roboter von Menschen aufgenommen wird. Dabei haben sie festgestellt, dass die Geschwindigkeit und die Nähe zur Person ausreichen, um die Aufmerksamkeit des Menschen auf den Roboter zu lenken.

Nicht nur das Verhalten spielt hinsichtlich der Wahrnehmung eines Roboters durch den Menschen als menschenähnlich eine Rolle, sondern auch sein Aussehen. 1970 hat Masahiro Mori [Mori, 1970] das sogenannte „Uncanny Valley“ – Phänomen populär ge-

macht. Der Effekt besteht darin, dass die wahrgenommene Menschenvertrautheit nicht linear mit menschenähnlichem Aussehen steigt, sondern ab einem gewissen Grad ins Negative umschlägt. Das „Uncanny Valley“ – Phänomen enthält ebenfalls den Effekt der Bewegung auf den Grad der Vertrautheit, der laut Mori sogar noch stärker als der Effekt des Aussehens ist. Mori geht in seiner Originalarbeit nicht auf die Beziehung zwischen Aussehen und Bewegung ein, obwohl die Kombination beider Faktoren eine wichtige Eigenschaft eines Roboters ist. Gee, Browne und Kawamura (2005) [F.C. Gee, W.N. Browne, K. Kawamura, 2005] stellen die Frage nach der Konsistenz, d.h. nach der gemeinsamen Wirkung von Bewegung und Aussehen. Auf den Aspekt der Bewegung wird in diesem Bericht zu einem späteren Zeitpunkt ausführlicher eingegangen.

Aus den beschriebenen Experimenten lässt sich erkennen, dass ein großer Bedarf an Forschung in Bezug auf die Mensch-Roboter-Interaktion besteht, um das gegenseitige Verständnis zu verbessern. Reale Experimente mit Robotern sind zeitaufwändig und kostspielig. Mit Computersimulationen, die ein abstraktes Abbild der realen Welt liefern, bietet sich eine Alternative, mit der getestet werden kann. Eine mögliche Simulationsumgebung sind Heider-Simmel-Filme. Die Filme zeigen sich bewegende Objekte auf einer Leinwand, deren Wirkung auf Versuchspersonen untersucht wurde. Die Animationsstudien von Heider und Simmel [M. Simmel, F. Heider, 1944] von 1944 zeigen, dass einfachen, sich bewegenden Figuren Motive und Absichten unterstellt werden. Geometrische Figuren, die sich mit unterschiedlicher Geschwindigkeit auf einer Leinwand bewegen, werden von den Versuchspersonen mit Menschen und Tieren verglichen. Dabei spielen auch hier das Aussehen und die Bewegung eine Rolle. Große, sich schnell bewegende Objekte werden als bedrohlich wahrgenommen, während kleine Objekte, die sich nur zusammen mit anderen Objekten bewegen, als passiv und hilfsbedürftig angesehen werden. Das Verhalten von Robotern in jedem beliebigen Umfeld kann folglich durch die eben beschriebenen Animationsfilme dargestellt werden.

Damit Simulationsexperimente auch von Laien durchgeführt werden können, sollen Animationen sich auch ohne besondere Vorkenntnisse erstellen zu lassen. 2011 wurde das

„Animationswerkzeug zur Visualisierung von sozialem Handeln“ im Rahmen eines Projektes erstellt. Das Animationswerkzeug ist eine Software zur Erstellung von Bewegungs- und Objektanimationen. Damit lassen sich also Heider-Simmel-Filme erstellen. Das Animationswerkzeug besteht aus drei Komponenten. Es enthält einen Grafikeditor, in dem Objekte und Pfade, auf denen sich die Objekte bewegen, erzeugt werden können, einen Animationseditor, in dem diverse Animationen erstellt werden können, wie z.B. eine Bewegungsanimation oder eine Farbanimation und einen Player, in dem die Animationen abgespielt werden können. Es gibt außerdem ein Szenenmodell, welches das Abspeichern von erstellten Grafiken ermöglicht, sowie das Abspielen der Animation in einem Browser.

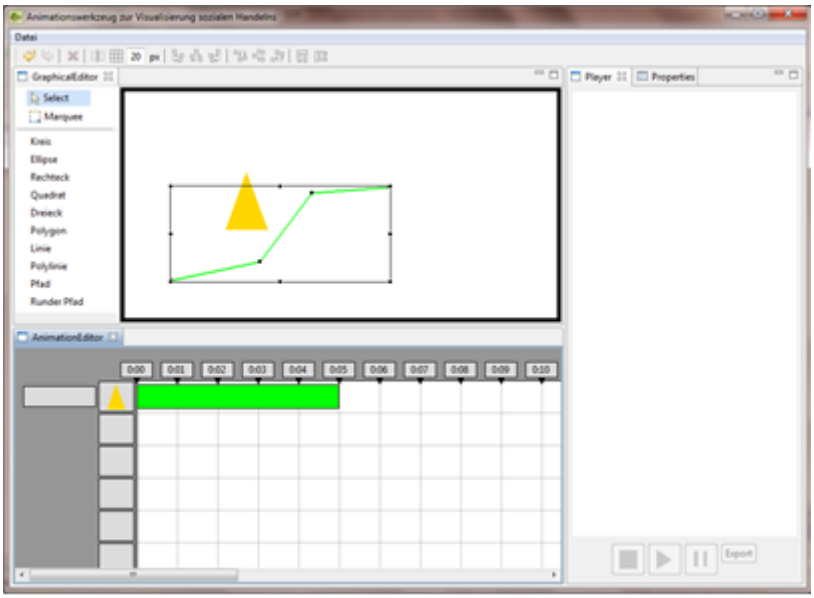

Abbildung 1: Oberfläche des Animationswerkzeugs

Die erste Version dieser Software wurde um Funktionalitäten und hinsichtlich der Benutzerfreundlichkeit erweitert. Weitere Informationen zur Softwarearchitektur und genutzten Tools befinden sich in den Projektberichten zu der Erstversion des Animationswerkzeuges z.B. von Jens Dallmann (2011) [Dallmann, 2011] und Stefan Schäfermeier (2011) [Schäfermeier, 2011]. In diesem Bericht wird genauer auf die Möglichkeit zur Erstellung von Runden Pfaden eingegangen, wobei deren Zweck, der mathematische

Hintergrund, die Realisierung und Evaluation dargestellt werden.

Die Bewegungen der animierten Objekte sollen natürlichen Bewegungsabläufen nachempfunden sein. Wie schon eingangs beschrieben, sollen die erstellten Animationen die reale Welt nach- bzw. abbilden. Die Bewegungen von Objekten in der realen Welt sind, unabhängig davon, ob Roboter oder Mensch, fließend und ohne abrupte Richtungsänderungen. Wir sprechen in diesem Zusammenhang von Runden Pfaden. Um eine Bewegungsanimation zu erstellen, wird ein Objekt einem erstellten Pfad zugeordnet und bewegt sich beim Abspielen entlang dieses Pfades. Ein solcher Pfad kann entweder Ecken haben oder rund sein.

Ein Roboter, der in einem Museum Führungen durchführt, muss die sich ständig verändernde Situation und Position von Besuchern erkennen und seine Bewegungen flexibel daran anpassen. Ein weiteres Beispiel wäre ein Serviceroboter, der sich im Wohnzimmer eines Patienten an die Position der Gegenstände anpassen muss. Bei jeder scharfen Kurve müsste der Roboter abbremsen und neu beschleunigen, was zu einem enormen Zeitverlust führen würde. Außerdem würde ein solches Verhalten zu unerwünschten Irritationen vonseiten der Menschen in der Umgebung des Roboters führen. Der Aspekt der Geschwindigkeitsveränderung ist jedoch nicht im Animationswerkzeug enthalten.

Menschen handeln zielorientiert und vorausdenkend und bewegen sich deshalb auf einem optimalen Weg, der aufgrund von Hindernissen und sonstigen Gegebenheiten nie gerade verläuft, sondern sich durch möglichst langgezogene, gekrümmte Kurven auszeichnet. Sie laufen nicht entlang gerader Linien, bei denen sie gezwungen wären, anzuhalten und die Richtung zu verändern (siehe Abbildung 2). Würden sie so laufen, wäre eine Adaption an plötzliche Veränderungen oder Hindernisse in der Umgebung nicht, oder nur mit zeitlicher Verzögerung, möglich. Menschliche Trajektorien verlaufen demnach meist kurvenartig.

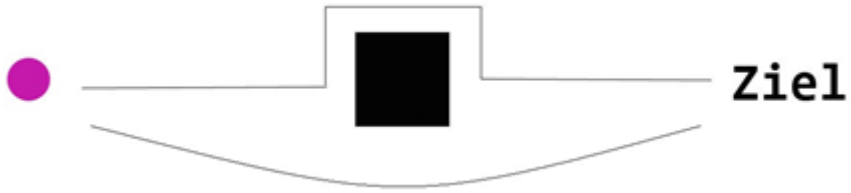

Abbildung 2: Bewegung eines Roboters um ein Hindernis

Die Art der Bewegung eines Objektes wird von Menschen interpretiert, den Objekten dabei Eigenschaften und Emotionen zugeschrieben.
Saerbeck und Bartneck [C. Bartneck, M. Saerbeck, 2010] haben 2010 in einer Studie die Beziehung zwischen Bewegungscharakteristika eines Roboters und dem wahrgenommenen Affekt untersucht. Eine ältere Studie von Gaur und Scassellati
[V. Gaur, B. Scassellati, 2006] von 2006 belegt, dass eine Kombination aus Kurven-Koeffizienten, Geschwindigkeitswechsel und Richtungswechsel ausschlaggebend für das Unterscheiden zwischen animierten und nicht animierten Bewegungen ist. Saerbeck und Bartneck haben diese Ergebnisse aufgegriffen und den Effekt von Beschleunigung und Krümmung, jeweils in drei Stärkestufen, auf den Affekt untersucht. Um die Emotionen zu erforschen, wurden Eigenschaften verschiedener Emotions-Modelle kombiniert und sowohl Valenz, d.h die Wertigkeit im emotionalen Bereich (positiv bis negativ), Erregung, Dominanz, als auch positiver und negativer Affekt gemessen. Ergebnis der Studie ist, dass Beschleunigung einen signifikanten Effekt auf die wahrgenommene Erregung hat und auch die Krümmung einen Einfluss auf Erregung, Valenz und Dominanz besitzt. Vor allem die Valenz lässt sich teilweise auf die Interaktion zwischen Beschleunigung und Kurvenkrümmung zurückführen. Bei der Wahrnehmung eines Roboters spielen Bewegungsmerkmale, insbesondere Geschwindigkeit und Kurvenverläufe, also eine wichtige Rolle. Im nachfolgenden Teil werden die theoretischen Grundlagen zur Erstellung von Kurven erläutert.

2 Mathematischer Hintergrund

2.1 Kurven

Der Begriff der Kurve findet in vielen Bereichen Anwendung und wird abhängig vom jeweiligen Bereich unterschiedlich verstanden.
Mathematisch betrachtet ist eine Kurve in der Analysis und linearen Algebra ein Objekt, welches einer Linie ähnelt, jedoch nicht gerade sein muss. Das bedeutet, eine Linie ist eine besondere Kurve ohne Krümmung. Krümmung bezeichnet die Abweichung einer Kurve von einer Geraden. Die Krümmung steht auch für das Krümmungsmaß, welches für jeden Punkt der Kurve quantitativ angibt, wie stark diese Abweichung ist. Eine Kurve kann als Bild eines Weges definiert werden. Ein Weg ist eine stetige Abbildung von einem Intervall in den betrachteten Raum, also z. B. in die euklidische Ebene $\mathbb{R}^2$. Eine Kurve kann durch eine oder mehrere Gleichungen mithilfe von Koordinaten beschrieben werden, z.B. durch Polynome.
Eine Kurve ist zudem differenzierbar, d.h. es lässt sich zu jedem Punkt eine Tangente an die Kurve anlegen. Eine Funktion ist dann differenzierbar, wenn sie an jeder Stelle ihres Definitionsbereiches differenzierbar ist. Das bedeutet, eine Kurve enthält keine Ecken, kann aber gerade Strecken enthalten.
Bekanntestes Beispiel für eine Kurve ist der Funktionsgraph. Der Funktionsgraph gibt einen Zusammenhang zwischen den x-Werten und den zugeordneten y-Werten einer Funktion f wieder. Zugrunde liegt ein kartesisches Koordinatensystem mit x- und y-Achse. Besondere Funktionsgraphen sind lineare und quadratische Funktionen. Lineare Funktionen haben Geraden als Graph. In kartesischen Koordinaten (x, y) erfüllen die Geraden die Gleichung: $y = f(x) = a * x + b$, wobei x die unabhängige Variable, y die abhängige Variable, b der Ordinatenabschnitt und a die Steigung der Geraden ist. Eine quadratische Funktion (auch ganzrationale Funktion 2. Grades oder Polynom 2. Grades) ist eine Funktion, die als Funktionsterm ein Polynom vom Grad 2 besitzt, also von der Form $f(x) = a * x^2 + b * x + c, a \neq 0$. Im Folgenden werden verschiedene Methoden vorgestellt, mit denen Kurven beschrieben werden können.

2.2 Parabeln

Wenn drei Punkte der Form (x, y) gegeben sind und weder alle x- noch alle y-Koordinaten der drei Punkte identisch sind, lässt sich eine quadratische Funktion ermitteln, die eine Kurve beschreibt, auf der die Punkte liegen. Durch Einsetzen der Punkte in die Gleichung $f(x) = a * x^2 + b * x + c$ ergibt sich ein Gleichungssystem aus drei Gleichungen mit drei Unbekannten, aus denen sich die Koeffizienten ableiten lassen. Mithilfe eines Additionsverfahrens oder des Gauß-Algorithmus lassen sich die Koeffizienten ermitteln. Bezogen auf das Animationswerkzeug, wäre das Zeichnen einer Kurve nach drei gesetzten Punkten möglich. Nachdem die quadratische Funktion bestimmt wurde, muss die Kurve gezeichnet werden. Für alle x-Werte zwischen den Punkten p und $p+1$ und $p+1$ und $p+2$ wird der entsprechende y-Wert berechnet. So ergibt sich eine Kurve zwischen den gesetzten Punkten p, $p+1$ und $p+2$. Wird ein weiterer Punkt $p+3$ gesetzt, wird das Verfahren wiederholt, diesmal mit den Punkten $p+1$, $p+2$ und $p+3$. Das bedeutet, der letzte Kurvenabschnitt wird beim Setzen eines neuen Punktes neu berechnet. Da so Kurven mit Ecken entstehen können, muss das Segment zwischen $p+1$ und $p+2$, d.h. das erste Segment einer Kurve, angepasst werden. Dies könnte durch das Berechnen eines Mittelwertes des alten und neu errechneten Wertes geschehen.
Ein anderes Verfahren wäre ein einfaches Interpolationsverfahren für 2D-Anwendungen, das sogenannte Parabolic Blending: Es seien $n+1$ Punkte P_0, P_1, ..., P_n im $\mathbb{R}^2$ gegeben (n > 2). Man bestimmt für je drei aufeinanderfolgende Punkte das Lagrange-Polynom 2. Grades (Parabel) und bildet zwischen je zwei Punkten eine gewichtete 'Mittel-Kurve' S_r aus den beiden überlappenden Parabeln. Parabeln Q_r :

$Q_r(0) = P_r; Q_r(1) = P_r + 1; Q_r + 1(0) = P_r; Q_r + 1(1) = P_r + 1$

Die gesuchte Kurve S_r berechnet sich als gewichtetes Mittel der Parabeln:

$S_r(t) = (1-t)Q_r(t) + tQ_r + 1(t)(t[0;1].)$

Die Stetigkeit ergibt sich aus der in Sr zusammengesetzten Kurve und der 1. Ableitung. In den Randbereichen behält man die Parabelstücke bei. Die Sr Kurve ist eine Subspline-Funktion 3. Grades [TU Cottbus,].

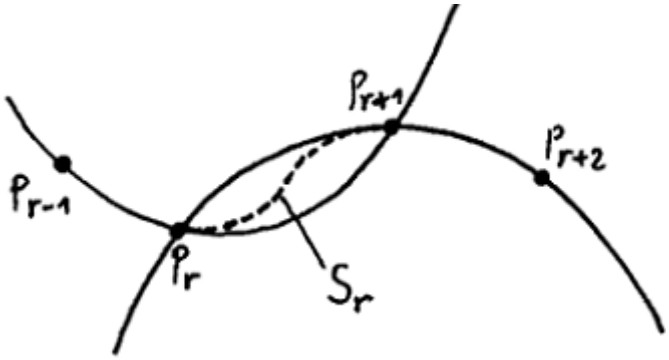

Abbildung 3: Parabel

Mit dieser Methode würden zwar runde Kurven entstehen, jedoch auf Kosten hoher Rechenleistung, da nach jedem gesetzten Punkt eine neue Kurve berechnet und ein bestehendes Segment angeglichen werden muss.

2.3 Polynome

In der elementaren Algebra ist eine Polynomfunktion eine Funktion P der Form

$$P(x) = \sum_{i=0}^{n} a_i x^i = a_n x^n + a_{n-1} x^{n-1} + ... + a_2 x^2 + a_1 x + a_0, n \geq 0,$$

wobei als Definitionsbereich für die Variable x jede beliebige R-Algebra in Frage kommt, wenn R der Wertebereich der Koeffizienten ist. Häufig ist dieser jedoch die Menge der ganzen, der reellen oder der komplexen Zahlen. Die a_i werden Koeffizienten genannt, sie stammen aus einem Ring R, der die ganzen, reellen bzw. komplexen Zahlen umfasst. Die Exponenten sind natürliche Zahlen, der höchste Exponent gibt den Grad eines Polynoms an. Polynome des Grades 2 werden quadratische Funktionen genannt (s.o.), Polynome des Grades 3 werden kubische Polynome genannt. In der Computergrafik werden Kurven sehr häufig mit Hilfe von Polynomen dargestellt.
Mithilfe der Polynom-Interpolation lassen sich aus gesetzten Punkten Kurven zeichnen. Hintergrund der Interpolation (lat. interpolare „auffrischen“, „umgestalten“, „verfälschen“) ist, dass man häufig weiß, dass zwischen gewissen Größen eine funktionale Abhängigkeit besteht, man die Abhängigkeit jedoch nicht quantitativ kennt, d.h. es gibt

keine geschlossene Formel für die Funktion f. Aus gegebenen Stichproben bzw. Messwerten muss man den Funktionswert $y = f(x)$ an einer gegebenen Stelle x ermitteln. Die Messwerte $(x_0, y_0), \ldots, (x_n, y_n)$ werden als Stützstellen bezeichnet. Gesucht ist ein Wert y an einer Stelle x, die selber kein Stützpunkt ist. Dabei wird davon ausgegangen, dass der funktionale Zusammenhang durch eine möglichst „glatte“ Kurve passend wiedergegeben wird. Für diese Funktion f soll dann der Wert $f(x)$ berechnet werden. Die unbekannte Funktion f wird durch p ersetzt, p kann so gestaltet werden, dass $y = p(x) \approx f(x)$.
Als Näherung an p dient ein geeignetes Polynom vom Grad n. Das Polynom ist durch $(n + 1)$ Stützstellen eindeutig definiert. Bezogen auf die Runden Pfade bedeutet dies, dass mindestens drei Punkte gesetzt werden müssen, mithilfe derer ein Polynom gesucht wird, welches die Stützstellen interpoliert.
Newton hat für das Problem der Interpolation einen Lösungsweg gefunden, der als „dividierte Differenzen“ bekannt ist. Das Prinzip der dividierten Differenzen ist eine rekursive Zerlegung des gesuchten Polynoms in zwei Polynome. Durch die sich bei der Zerlegung ergebenden Gleichungen können die Koeffizienten berechnet werden. Nachdem das interpolierende Polynom ausgerechnet wurde, können beliebig viele interpolierte Punkte errechnet werden, die dann eine Kurve erzeugen. Die Kurve würde bei den Runden Pfaden innerhalb der Stützstellen verlaufen, nicht darüber hinaus. Außerhalb und auch an den Rändern beginnt das Polynom jedoch stark zu oszillieren.
Wenn Polynome auf diese Art und Weise genutzt werden würden, um Runde Pfade zu erstellen, wäre nur eine begrenzte Anzahl an zu setzenden Punkten erlaubt. Der Pfad könnte erst gezeichnet werden, wenn alle Punkte gesetzt wurden. Das würde jedoch ein schlechtes Feedback für den User bedeuten, da er erst beim Beenden der Arbeit das Ergebnis sehen würde. Zudem wird das oszillierende Verhalten umso schlimmer, je mehr Stützstellen es gibt. Je höher der Grad eines Polynoms, desto unbrauchbarer wird es, d.h. ein Runder Pfad dürfte nicht mehr als 20-30 Stützpunkte haben. Diese Einschränkung darf dem User jedoch nicht zugemutet werden.

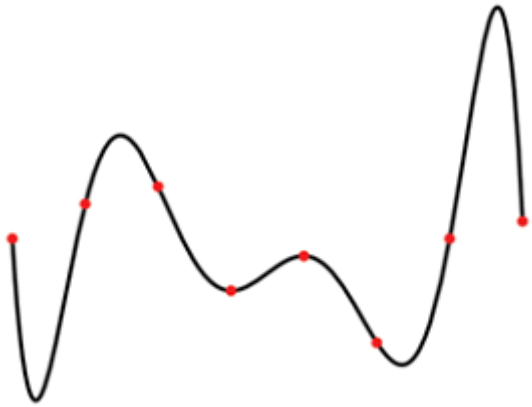

Abbildung 4: Polynom des 7. Grades mit 8 Stützstellen

2.4 Spline-Interpolation

Da die Polynom-Interpolation einige Nachteile aufweist, die mit steigendem Grad zusammenhängen, liegt es nahe, möglichst nur Polynome eines geringeren Grades zu verwenden. Die Idee ist, eine Kurve nicht durch eine einzige Funktion zu beschreiben, sondern durch eine Folge von n „kleinen“ Polynomen des dritten Grades. Hierbei spricht man von kubischen Spline-Funktionen. Die Kurve wird also in mehrere gleich große Teile zerlegt und interpoliert.

Es sei erneut eine Liste an Punkten gegeben und eine Folge von Polynomen $s_0, \ldots, s_{n-1}$ dritten Grades gesucht, die durch die Stützstellen möglichst „glatt“ interpolieren. Es werden n kubische Polynome $s_0(x), \ldots, s_{n-1}(x)$ der Art $s_i(x) = a_i + b_i * (x - x_i) + c_i * (x - x_i)^2 + d_i * (x - x_i)^3$gesucht.

Das bedeutet, es müssen insgesamt $4n$ unbekannte Koeffizienten (a_i, b_i, c_i, d_i) bestimmt werden und somit $4n$ Gleichungen gelöst werden. Damit die Interpolation „glatt“ ist, müssen benachbarte Polynome an ihrem Berührungspunkt sowohl im Wert als auch in der ersten und zweiten Ableitung übereinstimmen um zu garantieren, dass hier die gleiche Steigung vorherrscht. Für eine genaue Beschreibung der Rechnung sei auf S. 198-205 von ’Programmieren lernen – Eine grundlegende Einführung mit Java’ von Pepper [Pepper, 2007] verwiesen.

Der Aufwand zum Lösen des Gleichungssystems ist linear abhängig von n, da sich für jeden weiteren Abschnitt ein weiteres Polynom ergibt. Splines sind einfach in der Handhabung, mit geringem Rechenaufwand zu berechnen und eine gute Approximation der gegebenen Punkte. Ein weiterer Vorteil bei der Nutzung von natürlichen Splines ist, dass die Oszillationen durch die abschnittsweise Definition von Funktionen niedrigeren Grades verhindert werden. Es gibt ein weit verbreitetes Spektrum von Spline-Arten bzw. –Gruppen. Zu den bekanntesten Vertretern gehören:

- Kubische Splines (s.o.)
- Parametrische Splines
- B-Splines
- Bézier Splines/Kurven
- Hermite Splines

2.5 Parametrische Splines

Bei der Nutzung von natürlichen Splines muss beachtet werden, dass jedem x-Wert nur höchstens ein y-Wert zugeordnet werden darf. Dies gilt bei allgemeinen zweidimensionalen Kurven jedoch nicht mehr und es werden stattdessen Parametrische Splines verwendet. Ein Parametrischer Spline muss benutzt werden, wenn die Monotonie der Stützstellen nicht gegeben ist. Monotonie in diesem Zusammenhang bedeutet, dass die Stützpunkte entweder monoton wachsen, fallen oder konstant bleiben. Bei nicht monotonen Stützstellen können Schleifen wie im nachfolgenden Beispiel entstehen. Demnach kann bspw. nach dem Punkt $(6|1)$ der Punkt $(2,2|1,2)$ folgen. Dies ist nun kein Funktionsgraph mehr.

Da es im Animationswerkzeug möglich sein soll, Pfade zu erstellen, die nicht monoton sind, ist dies eine geeignete Variante, Runde Pfade zu realisieren. Aus der gegebenen Punktemenge $(x_1, y_1), \ldots, (x_n, y_n)$ wird ein Parametrischer Spline konstruiert, indem

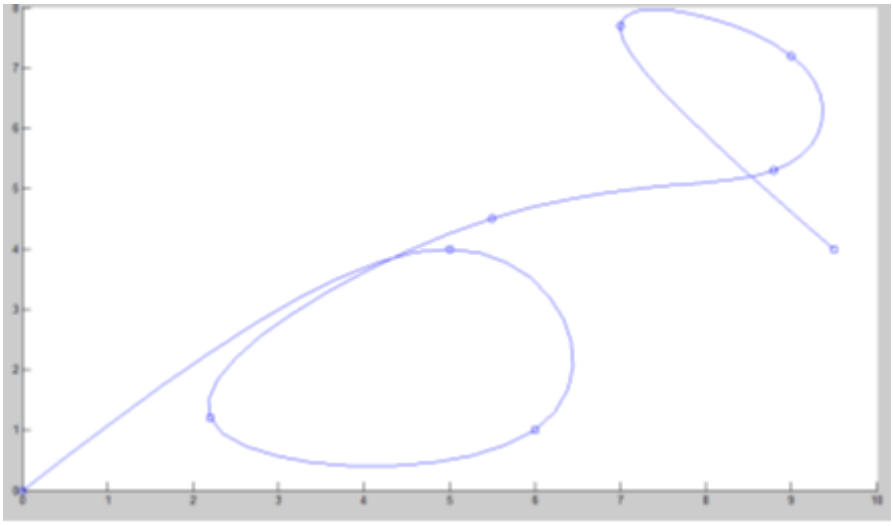

Abbildung 5: Parametrische Spline

die Koordinaten als Funktionen einer Variablen t angesehen werden, also $x(t)$ und $y(t)$. Somit sind die Stützstellen $x(t_1), \ldots, x(t_n)$ und $y(t_1), \ldots y(t_n)$. Bei den Parametrischen Splines ist die Wahl des Kurvenparameters t ausschlaggebend für den Verlauf der Kurve. Monotones Wachstum ist die Voraussetzung an den Parameter t. $t_0 < t_1 < \ldots t_n$. Die Wahl des Parameters t kann durch verschiedene Möglichkeiten bestimmt werden. Eine Möglichkeit wäre die Bogenlänge der einzelnen Teilabschnitte zu benutzen oder t in gleichmäßigen Abständen zu wählen (äquidistante Parametrisierung). Eine andere Möglichkeit wäre, den Parameter über die Annäherung der Kurve durch die jeweilige Länge der Sehnen zwischen den Punkten zu definieren (chordale Parametrisierung).

2.6 B-Splines

B-Splines werden ebenfalls genutzt um zweidimensionale Kurven zu zeichnen. Auch hier setzen sich die Kurven stückweise aus Polynomen zusammen. Der Kurvenverlauf wird durch Kontrollpunkte, sogenannte De-Boor-Punkte, bestimmt. Die Kurve liegt innerhalb der Kontrollpunkte (konvexe Hülle).
B-Splines wären zwar eine Möglichkeit, Runde Pfade zu erstellen, jedoch soll die Kurve im Hinblick auf die Benutzerfreundlichkeit durch die gesetzten Punkte verlaufen und

sich ihnen nicht nur annähern. Es ließe sich zwar erzwingen, dass die Kurve durch einen bestimmten Punkt verläuft, indem man den Punkt dreimal hintereinander in die Stützstellen aufnimmt, dies wäre jedoch bei jedem Punkt nötig und deshalb sehr umständlich. Da die Kurve im Normalfall nicht mehr durch die Stützstellen verläuft, können B-Splines nicht verwendet werden, um interpolierende Werte von Funktionen zu bestimmen.
Bei der B-Spline-Interpolation werden zur Berechnung eines interpolierenden Punktes $Q(t)$ die einzelnen Kontrollpunkte mit unterschiedlichen Gewichten zur Hilfe genommen. Eine Gewichtsfunktion gibt die Stärke des Einflusses an, je weiter weg vom Punkt, desto kleiner ist der Einfluss. Für die Bestimmung der Gewichtsfunktion gibt es mehrere Varianten, auf die hier jedoch nicht näher eingegangen werden soll.

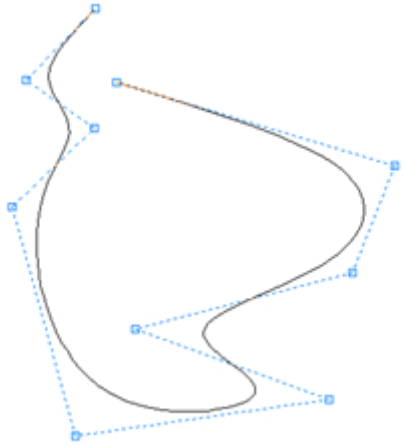

Abbildung 6: B-Spline

2.7 Bézier-Splines

Die Bézier-Kurve ähnelt der B-Spline, auch hier dienen die Stützstellen nur noch als Kontrollpunkte, sodass die Kurve nicht mehr durch alle Punkte verläuft. Man unterscheidet quadratische Bézierkurven von kubischen Bézierkurven. Quadratische werden durch drei Punkte bestimmt, während Kubische von vier Punkten bestimmt werden.

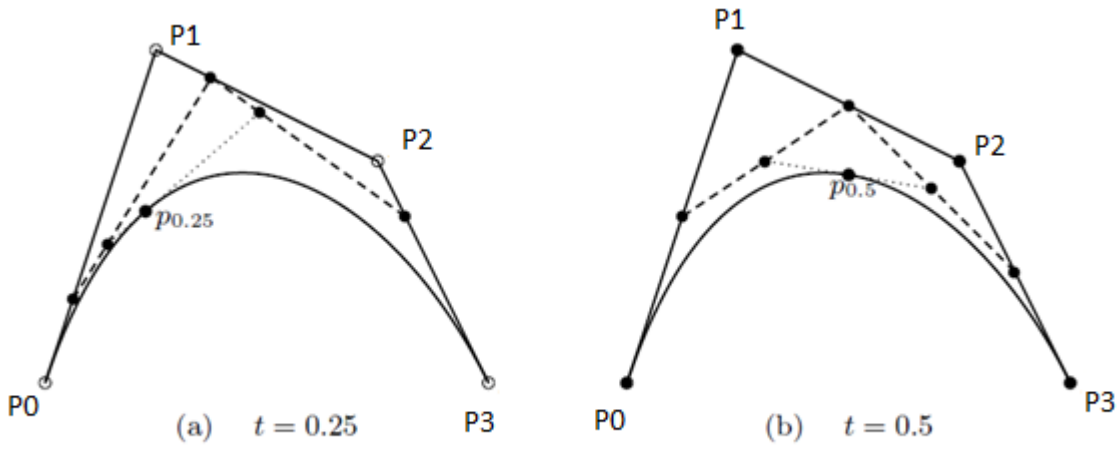

Abbildung 7: Bézierkurve

P_0 und P_3 bilden die Endpunkte, P_1 und P_2 regeln nur den Kurvenverlauf. Die Tangente an den Endpunkten entspricht den Kontrolllinien $\overline{P_0, P_1}$ bzw. $\overline{P_2, P_3}$. Jeder einzelne Punkt auf der Kurve wird durch einen Parameterwert $t \in [0, 1]$ bestimmt. Auf allen drei Kontrolllinien $\overline{P_0, P_1}$; $\overline{P_1, P_2}$ und $\overline{P_2, P_3}$ wird der Punkt bestimmt, der die Linie im Verhältnis t (bei (a) 0,25) : $(1 - t)$ teilt, wodurch zwei neue Linien entstehen. Diese beiden Linien werden erneut im gleichen Verhältnis geteilt, wodurch noch eine weitere Linie entsteht, die ebenfalls geteilt wird. Dies liefert den Punkt $p_{0,25}$ auf der Kurve, der zu dem Wert $t = 0,25$ gehört. Eine Bézierkurve n-ten Grades zu gegebenen $n + 1$ Kontroll- oder Bézierpunkten $(P_i)_{i=0^n}$, die das so genannte Kontrollpolygon bilden, ist für $t \in [0, 1]$ definiert als $C(t) = \sum_{i=0}^{n} B_{i,n}(t)P_i$, wobei $B_{i,n}(t) = \binom{n}{i} t^i (1 - t)^{n-i}$ das i-te Bernsteinpolynom n-ten Grades ist.

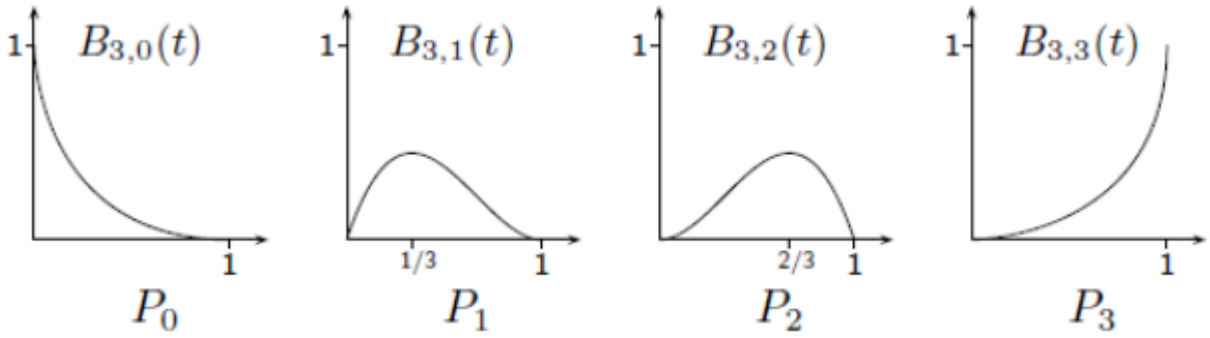

Abbildung 8: Bernsteinpolynome

Aus der Kombination beider genannter Formeln ergibt sich:

$$\begin{aligned} C(t) &= \sum_{i=0}^{3} \binom{3}{i} t^i (1-t)^{3-i} P_i \\ &= (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3t^2(1-t)P_2 + t^3 P_3 \\ &= (-P_0 + 3P_1 - 3P_2 + P_3)t^3 + (3P_0 - 6P_1 + 3P_2)t^2 \\ &\quad + (-3P_0 + 3P_1)t + P_0, t \in [0,1] \end{aligned}$$

Durch Einsetzen der Punkte und der Wahl eines Wertes für t lässt sich die Gleichung lösen. Im Vergleich zur B-Spline zeigt sich bei der Verwendung von Bézierkurven ein größerer Rechenaufwand. Zudem wird das Kontrollpunkt-Polygon bei der B-Spline-Kurve besser approximiert als bei der Bézierkurve.

Zusammenfassend lässt sich sagen, dass eine Vielzahl an Möglichkeiten besteht, Runde Pfade darzustellen. Es ist abzuwägen, ob die Kurve besonders „glatt“ verlaufen soll, ob sie durch alle gesetzten Punkte verlaufen oder sich diesen nur annähern soll und wie hoch der Rechenaufwand ist. Bezogen auf die zu entwickelnde Funktion der Runden Pfade für das Animationswerkzeug muss jedoch berücksichtigt werden, dass die Art der Darstellung durch SVG möglich sein muss. SVG bietet neben echten Kurven wie bei Kreisen und Ellipsen, Linienzügen und Polygonen auch elliptische Bögen und Bézierkurven an. Verschiedene Kurvenstücke erster, zweiter oder dritter Ordnung sind miteinander kombinierbar und mischbar.

3 Alpha-Version

Die erste Version und damit die Alpha-Version des Animationswerkzeuges enthält bereits die Möglichkeit, Runde Pfade zu erzeugen.
Die Erstellung der Kurven basiert auf der Nutzung von kubischen Bézierkurven. Wie schon beschrieben, sind bei der Verwendung von Bézierkurven Kontrollpunkte nötig, die in dieser Version von dem User selber gesetzt werden. Das bedeutet, erst nach vier selbst gesetzten Punkten wird die erste Kurve gezeichnet. Ein Kurvenabschnitt i besteht immer aus vier Punkten, und zwar einem Startpunkt, zwei Kontrollpunkten und einem Endpunkt. Die Kurve verläuft nun jedoch nicht durch Punkt $p + 1$ und Punkt $p + 2$, sondern nur durch Punkt p und Punkt $p + 3$. Der zuletzt selbst gesetzte Punkt ist der erste Punkt des nächsten Kurvenabschnitts $i + 1$. Damit keine Sprünge entstehen und die Kurven fließend ineinander übergehen, wird der erste Kontrollpunkt des $i +$ 1-ten Abschnitts automatisch als Spiegelpunkt des zweiten Kontrollpunktes des i-ten Abschnittes gesetzt. So ist die Steigung in dem Punkt, in dem die Abschnitte ineinander übergehen, identisch. Die Kurvenberechnung beeinflusst also die Richtung und Form des nächsten Abschnittes.

Die relevanten Klassen für die Runden Pfade im Animationswerkzeug sind RoundedPath und PathFigure. Die Speicherung der einzelnen Punkte und die Berechnung werden voneinander getrennt um das Programm leichter erweitern zu können.

Wie Abbildung 9 zeigt, ist RoundedPath eine Unterklasse der Klasse Path, hier werden alle Startpunkte, Endpunkte und Kontrollpunkte gespeichert. In der Methode *addPoint (Point p)* wird ein neuer Punkt der PointList hinzugefügt, gegebenenfalls wird hier auch der berechnete Kontrollpunkt der PointList hinzugefügt.
In der PathFigure findet die Rundung der Kurvenabschnitte statt. Unvollständige Abschnitte werden wie normale Pfade behandelt. Für jeden Kurvenabschnitt, der vorher bestimmt wurde, werden die Punkte in die Gleichung der kubischen Bézierkurve eingesetzt und die einzelnen Punkte der Kurve gezeichnet. Für eine genauere Beschreibung der Berechnung sei auf den Projektbericht von Christian Wilms [Wilms, 2011] verwiesen.

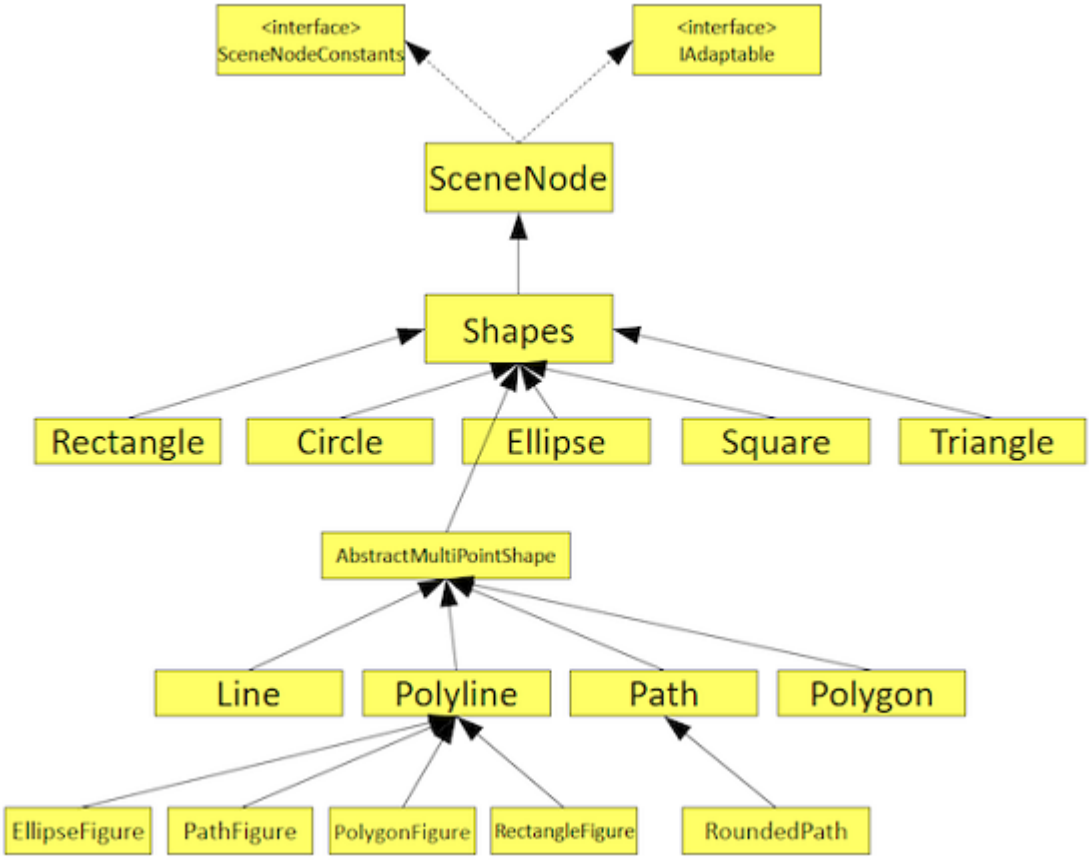

Abbildung 9: Modellhierarchie

4 Studie – Eine Evaluation der Alpha-Version

4.1 Vorgehen

Um die Funktionalität der Runden Pfade in der Alpha-Version zu testen und mögliche Tendenzen für die Weiterentwicklung dieses Features festzustellen, wurde eine Pilotstudie mit 6 Personen durchgeführt. Die Versuchspersonen (VP) sollten insgesamt vier Werkzeuge zur Erstellung von Runden Pfaden testen und bewerten.

Neben der Pfadkonstruktionskomponente des Animationswerkzeuges wurden drei Programme genutzt: GIMP, Inkscape und Scribus. Alle drei Programme bieten die Möglichkeit, mit Bézierkurven Kurven zu erstellen.
Aufgabe der Versuchspersonen war es, mit jedem der vier Werkzeuge eine vorgegebene s-förmige Kurve nachzuzeichnen und anschließend zu bewerten, wie sie mit der Erstellung zurechtkamen, wie intuitiv die Bedienung war, ob diese den Erwartungen entsprach und welche Erwartungen sie hatten. Zudem sollten die VP bewerten, wie sehr die von ihnen selbst erstellte Kurve der vorgegebenen Kurve entspricht. Außerdem sollten sie dem Werkzeug eine Schulnote geben. Die Reihenfolge der Werkzeuge erfolgte zufällig. Bei allen Versuchspersonen handelte es sich um Personen, die keinerlei Erfahrungen mit Grafikprogrammen und der Erstellung von Pfaden hatten. Vor dem Zeichnen der Kurve hatten die VP jedoch ausreichend Zeit, sich mit dem jeweiligen Werkzeug und der Funktionsweise vertraut zu machen.

4.2 Ergebnisse

4.2.1 Allgemeine Ergebnisse

Das Animationswerkzeug entsprach nicht den Erwartungen der Versuchspersonen. Obwohl vier von sechs VP mit dem Endergebnis ihrer Kurve zufrieden waren, gab es Probleme bei der Erstellung.
Alle VP waren irritiert durch den Verlauf der Kurve. Da der Pfad nicht durch alle gesetzten Punkte verläuft und die Kontrollpunkte nicht als solche zu erkennen sind, erfordert es enormes Mitdenken vonseiten des Users. Eine der VP gab an, „mit zu überlegen, wie die nächste Kurve erstellt wird“. Da viele der VP das Prinzip der Kurvenerstellung nicht verstanden haben, reagierten sie unzufrieden auf die gezeichneten Kurven, die nicht dem entsprachen, was sie erwartet hatten.
Das Werkzeug wurde ebenfalls nicht als intuitiv zu bedienen empfunden. Einige der VP haben versucht, die Punkte „anzufassen“ und nachträglich zu verändern oder durch „Ziehen“ den Kurvenverlauf zu ändern. Um die vorgegebene Kurve nachzuzeichnen, haben zwei VP sehr viele Punkte nebeneinander gesetzt. So entsteht zwar ein sehr unübersicht-

liches Bild, doch entspricht die Kurve der Vorgegebenen eher.
Die übrigen getesteten Werkzeuge funktionieren prinzipiell alle ähnlich. Durch Klicken und Ziehen entstehen mithilfe von Hilfsachsen Kurven, die sich auch nachträglich durch Anfasser verändern lassen. Für die VP war es sehr schwer, sich an die Bedienung zu gewöhnen und zu antizipieren, wie sich ihre Aktionen auf die Kurve auswirken. Sie haben sich ebenfalls schnell mit der erstellten Kurve zufrieden gegeben, auch wenn die Kurve nicht exakt der vorgegebenen Kurve entsprach. Dennoch wurden zwei der anderen Werkzeuge (Inkscape und GIMP) besser bewertet als die Pfaderstellungskomponente des Animationswerkzeuges.

Durch die Pilotstudie haben sich hinsichtlich der Erstellung Runder Pfade diverse Probleme bemerkbar gemacht, die sich sowohl auf funktionale, als auch auf psychologische Aspekte beziehen.

4.2.2 Funktionale Aspekte

Die gezeichnete Kurve entspricht meist nicht dem intendierten Pfad, das heißt, entweder muss sich der Nutzer mit dem Ergebnis zufrieden geben oder die Kurve noch einmal erstellen. Dies führt zu einem Mehraufwand. Nachteilig ist ebenfalls, dass es im Nachhinein nicht möglich ist, die Punkte und die Stärke der Krümmung zu verändern. Zudem wird durch den automatisch gesetzten Kontrollpunkt der Kurvenverlauf so stark beeinflusst, dass es dem Nutzer nicht immer möglich ist, die gewünschte Kurve zu erstellen. Ein weiterer Nachteil ist, dass die Kurve erst nach vier gesetzten Punkten gezeichnet wird und der User erst ab diesem Punkt ein Feedback erhält.

4.2.3 Psychologische Aspekte

Da die Kurve nicht durch alle gesetzten Punkte verläuft, erscheint das Werkzeug unberechenbar, was dazu führt, dass der Nutzer sich dem Programm nicht gewachsen fühlt. Der Nutzer darf keine Hemmungen vor der Benutzung haben, diese verhindern effektives und effizientes Arbeiten oder schrecken ihn soweit ab, dass er von der Nutzung der Runden Pfade absieht. Er verliert das Gefühl der Kontrolle. Dies ist jedoch ein bedeutender

Faktor hinsichtlich der Benutzerfreundlichkeit.
Ein Gefühl der Kontrolle gehört zu den 8 goldenen Regeln, die Shneiderman für ein gutes Interface Design aufgestellt hat [C. Plaisant, B. Shneiderman, 2010].
Zudem geben die automatisch gesetzten Kontrollpunkte sehr stark die Richtung vor und der User muss sich an die neue Situation anpassen, er reagiert eher anstatt zu agieren. Neben dem Kontrollverlust kommt es zu einem weiteren Problem. Der User ist gezwungen, zu antizipieren, wie die Kurve aussieht. Eine weitere Regel Shneidermans ist es, das Kurzzeitgedächtnis zu entlasten. Der User soll sich nicht um Einzelheiten der Realisierung seines Vorhabens kümmern müssen. Die Erstellung einer Kurve soll ihm keine Probleme bereiten oder mehr Aufmerksamkeit als nötig erfordern, da diese für andere Zwecke gebraucht wird. Da nachträglich keine Kurve mehr verändert werden kann, wird von dem User generell sehr viel Nachdenken und Planen erfordert.
Ein weiterer Aspekt, der berücksichtigt werden muss, ist, dass das Animationswerkzeug auch für Laien geeignet sein muss. Das bedeutet, es soll möglich sein, das Werkzeug intuitiv zu bedienen ohne sich lange einzuarbeiten. Es darf ebenfalls keine Voraussetzung sein, die mathematischen Aspekte, die im Hintergrund ablaufen, zu kennen.
Dem User sollten nötige Informationen mitgeteilt werden, doch darf er nicht mit Einzelheiten der internen Realisierung konfrontiert werden. Dass bei der Kurvenerstellung Bézierkurven verwendet werden und wie genau dies geschieht, darf den User in seiner Arbeit nicht betreffen.

5 Beta-Version

In einer von unserer Projektgruppe erstellten neuen Version wurden einige der genannten Kritikpunkte behoben.
Es werden weiterhin Bézierkurven genutzt um die Kurven zu zeichnen, da dies eine geeignete Methode ist, Krümmungen zu erzeugen. Des Weiteren bietet SVG bereits Bézierkurven an und die Alpha-Version verwendet sie ebenfalls, sodass der Fokus nicht auf einer kompletten Neuentwicklung lag, sondern auf einer Weiterentwicklung der bereits vorhandenen Implementierung.

Die maßgebliche Änderung gegenüber der Alpha-Version ist die automatische Erzeugung beider Kontrollpunkte. Setzte der User in der ursprünglichen Version alle Punkte inklusive Kontrollpunkte selber, setzt er in der Beta-Version nur noch die Punkte, durch die der Pfad verlaufen soll. Die für die Bézierkurve benötigten Kontrollpunkte werden folgendermaßen automatisch berechnet: In einem Winkel von 20° wird mit einem Abstand, der sich aus der Kurvenlänge geteilt durch 5 ergibt, ein neuer Punkt gesetzt. Diese Werte haben sich durch Austesten und Beurteilen der Kurvenkrümmung bei verschiedenen Parametern ergeben. Kurven, die mittels der so berechneten Kontrollpunkte erzeugt wurden, wurden von verschiedenen Testern als angemessen gekrümmt empfunden.
Eine Bézierkurve verläuft nun zwischen zwei selbst gesetzten Punkten, sodass der Pfad durch alle Punkte verläuft, die der User gesetzt hat. Ob die Kontrollpunkte zwischen zwei gesetzten Punkten links oder rechts liegen, d.h. ob sich dadurch eine links- oder rechtsgekrümmte Kurve ergibt, hängt von drei Faktoren ab:

1. Die Veränderung der x- und y-Koordinaten
   Sowohl für die x, als auch für die y-Koordinate wird geprüft, ob zwischen zwei Punkten p und $p+1$ ein Anstieg oder ein Sinken stattgefunden hat. Dies ergibt vier mögliche Kombinationen des Richtungswechsels. Gleiches wird für die Punkte $p+1$ und $p+2$ geprüft, wodurch sich 16 Kombination ergeben.

2. Die Krümmung des vorherigen Kurvensegments
   Für jeden Kurvenabschnitt wird in einer Variablen gespeichert, ob die Kurve links- bzw. rechtsgekrümmt ist. Zusammen mit den möglichen Kombinationen, die sich aus 1. ergeben, sind insgesamt 32 Kombinationen zu berücksichtigen.

3. Die Steigung des vorherigen Kurvensegments
   Je nach der Stärke der Steigung ergibt sich die Krümmung des aktuellen Kurvensegments. Dies ergibt 64 mögliche Kombinationen, die bei einem neuen Punkt für das Setzen der Kontrollpunkte in Frage kommen.

Nachfolgendes Beispiel erklärt das Zusammenspiel dieser drei Komponenten der Berechnung.

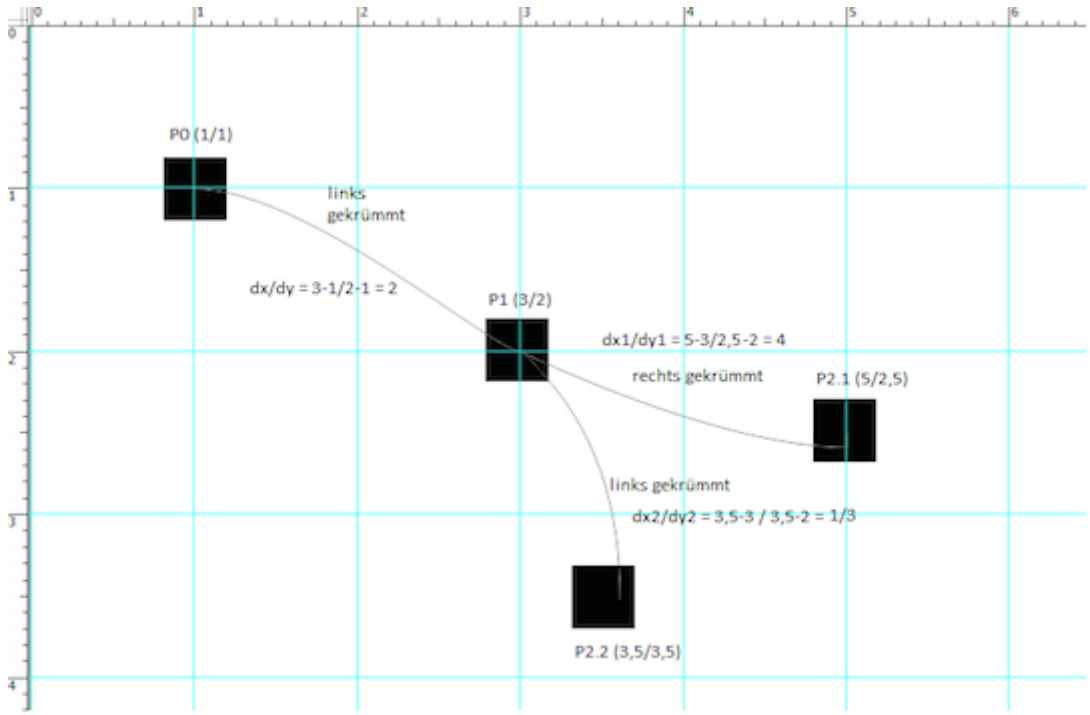

Abbildung 10: Krümmungsberechnung der Kurve

Zwischen P_0 und P_1 findet bei x- und y-Koordinate ein Anstieg statt, auch zwischen P_1 und $P_{2.1}$ und zwischen P_1 und $P_{2.2}$ ist dies der Fall. Der Pfad zwischen P_0 und P_1 ist linksgekrümmt. Da die Steigung zwischen P_1 und $P_{2.1}$ größer ist als die Steigung zwischen P_0 und P_1, ändert sich die Krümmung um einen fließenden Übergang zwischen den beiden Segmenten zu schaffen. Die Kurve zwischen P_1 und $P_{2.2}$ bleibt linksgekrümmt, da die Steigung sich nicht vergrößert hat.

In der Klasse *RoundedPath* findet weiterhin die Berechnung der Punkte statt. In der Methode *chooseControlPoint* werden die oben beschriebenen Kriterien abgeprüft und mit Fallunterscheidung untersucht, welche Krümmung nötig ist.
In der *addPoint*-Methode werden zu Beginn des Pfades, wenn der zweite Punkt gesetzt wird, die beiden Kontrollpunkte auf den ersten und letzten Punkt gesetzt, wodurch eine

Linie entsteht. Dies ist sinnvoll, da noch nicht klar ist, in welche Richtung der Pfad sich bewegt und wie die Krümmung aussehen soll. Erst beim Setzen des dritten Punktes wird der Pfad gerundet. Nachträglich wird auch das erste Kurvensegment gekrümmt. Sobald die Größe der PointList sechs übersteigt, finden keine nachträglichen Änderungen mehr statt, gerundet wird jedes neue Kurvensegment.
Als weitere Änderung berechnet sich der erste Kontrollpunkt nicht mehr als Spiegelpunkt des vorherigen Kontrollpunktes. Im Zuge der Entwicklung der Beta-Version wurde dem Pfad ebenfalls eine richtungsanzeigende Pfeilspitze hinzugefügt um die Orientierung zu erleichtern.

Klarer Vorteil der neuen Version ist das verbesserte Feedback für den User, da der Pfad tatsächlich durch die gesetzten Punkte verläuft. Der Pfad entspricht eher den Erwartungen des Users.
Nachteilig ist jedoch, dass es nicht immer glatte Kurvenübergänge gibt, da die Steigung in dem Punkt, der zwei Kurvensegmente verbindet, nicht gleich ist. Ecken und Kanten verhindern also, das eine natürliche, fließende Kurve entsteht. Um dies zu vermeiden, muss dafür gesorgt werden, dass die Übergänge und der vorherige Kurvenabschnitt im Nachhinein angepasst werden.

6 Gamma-Version

Im Nachhinein hat unsere Kleingruppe eine weitere Version erstellt. In dieser Gamma-Version werden die Kontrollpunkte so berechnet, dass die Kurven noch etwas runder verlaufen. Auch hier wird die Kurve gezeichnet, sobald drei Punkte gesetzt wurden. Die Hilfspunkte werden anschließend mit Hilfe von drei Geraden ermittelt. Die erste Gerade geht durch die Punkte p und $p+2$, die zweite Geraden parallel dazu durch den Punkt $p+1$ und orthogonal zu den ersten beiden Geraden eine Dritte durch den Punkt $p+2$. Die Hilfspunkte, zum einen zu Punkt $p+2$, werden definiert durch die Hälfte der Strecke zwischen dem Schnittpunkt der beiden zuvor gezeichneten Geraden und dem Punkt $p+1$. Der Hilfspunkt vor Punkt $p+2$ ergibt sich, wenn man zu Punkt $p+1$ den

durch ROUNDNESS dividierten Vektor zwischen Punkt $p+2$ und Punkt p addiert. Der Hilfspunkt nach Punkt $p+2$ ergibt sich, indem man den durch ROUNDNESS dividierten Vektor subtrahiert.

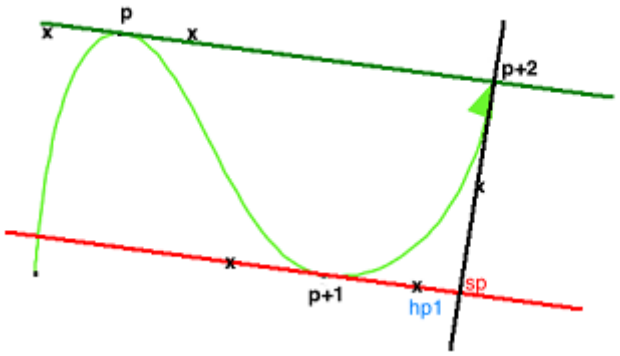

Abbildung 11: Kurvenberechnung Gamma-Version

7 Studie – Eine Evaluation der Beta- und Gamma-Version

7.1 Vorgehen

In einer Nachstudie haben wir getestet, wie die zwei neuen Versionen bei Versuchspersonen ankommen. Dafür haben wir im Animationswerkzeug eine Umgebung erstellt, in der ein Verkehrsabschnitt dargestellt wird, in dem sich vier Autos, bestehend aus Rechtecken und Kreisen, bewegen bzw. parken. Wir haben einen Verkehrsabschnitt gewählt, da Autos keine Ecken fahren, sondern immer auf runden Trajektorien verlaufen. Außerdem soll der Verkehrsabschnitt den Versuchspersonen helfen, sich in die Situation hineinzuversetzen, da jeder diese Situation kennen sollte, und somit also ein realer Bezug besteht. In dieser Umgebung wurde den VP die Aufgabe gestellt, eines der Autos aus einer Parklücke und durch den Verkehr zu manövrieren, um sich anschließend hinter einem weiteren Auto in den Verkehr einzufädeln. Dies sollte durch einen Runden Pfad realisiert werden.

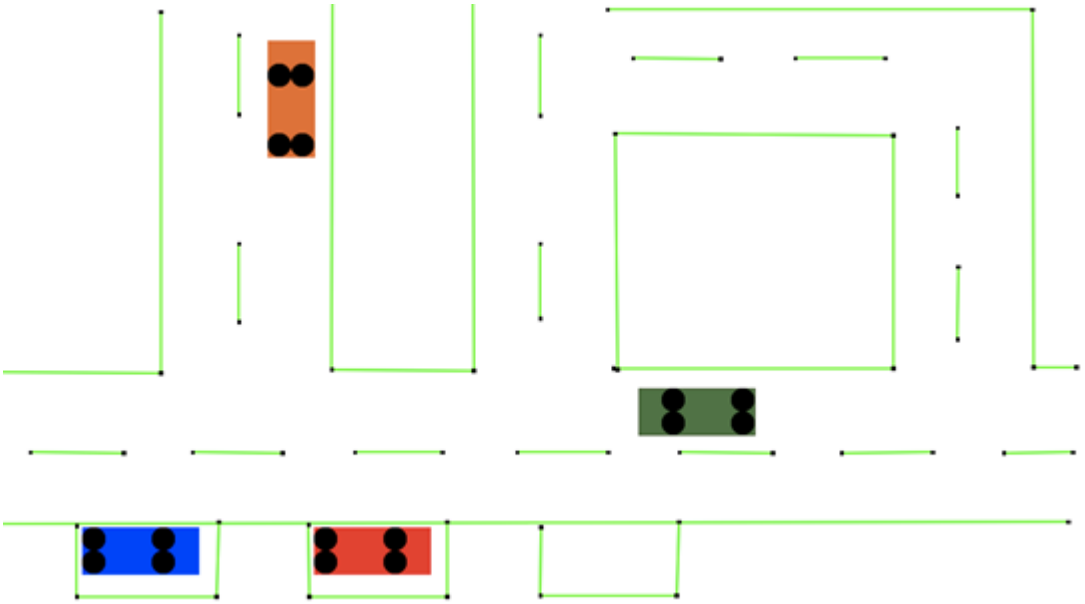

Abbildung 12: Verkehrssimulation

7.2 Ergebnisse

7.2.1 Allgemeine Ergebnisse

Alle VP kamen mit der Aufgabe, den Pfad zu erstellen, gut zurecht, waren jedoch bei beiden Versionen vom Aussehen des endgültigen Pfades nicht vollends zufrieden. Bei der Beta-Version entstehen viele Knicke wohingegen die Gamma-Version als schlingernd beschrieben wurde. Dennoch fanden zwei von drei VP die Beta-Version intuitiv, wohingegen die Gamma-Version sehr unterschiedlich bewertet wurde, was auf die manchmal starken Veränderungen des Pfadverlaufes im Nachhinein zurückzuführen ist.
Die Handhabung des Werkzeuges fiel den VP deutlich leichter als in der Pilotstudie mit dem alten Werkzeug. Ob die Beta-Version oder die Gamma-Version die Intuitivere ist, konnte jedoch nicht eindeutig ermittelt werden, denn beide wurden von den VP gut angenommen und die Aufgabe erfolgreich bewältigt.
Auffällig war zudem, wie die VP mit den Kurven der Straßen umgegangen sind. Anstatt durch den Gegenverkehr zu geraten oder eine Kurve zu schneiden, haben sie darauf ge-

achtet, eine Ideallinie zu treffen. Dabei haben die VP die Punkte des Pfades genau in die Kurven gesetzt, und auch sonst eher mehr Punkte gesetzt als weniger, um einen noch genaueren Verlauf der Kurve möglich zu machen oder die Kurve in der Beta-Version noch etwas runder.

Die Ergebnisse der Aufgabe zeigen demnach jedoch, dass mit der Gamma-Version oft Schlenker im Pfad entstehen, die erst im späteren Verlauf zu sehen sind. Die Erwartungen wurden aber in fast allen Fällen vom Werkzeug erfüllt, was mit dem ursprünglichen Werkzeug nicht der Fall war. Somit sind beide Versionen eine deutliche Verbesserung.

7.2.2 Funktionale Aspekte

Die gezeichneten Kurven entsprechen in beiden Versionen um einiges mehr dem intendierten Pfad, sie weichen manchmal noch etwas von dem gewünschten Pfad ab, jedoch geht der Pfad durch alle gezeichneten Punkte. Außerdem wird die Kurve jetzt schon nach drei gesetzten Punkten gezeichnet. Leider ist es jedoch weiterhin nicht möglich die Punkte im Nachhinein noch zu verändern oder zu verschieben. Hier besteht also noch Verbesserungsbedarf.
Einzig das weiterhin nicht mögliche Verändern der Kurve im Nachhinein erfordert ein wenig Planung und Nachdenken des Benutzers, ist aber zu verkraften, da die Kurve sich nicht unberechenbar verhält.

7.2.3 Psychologische Aspekte

Da die Kurve wie beschrieben durch alle gesetzten Punkte verläuft, hat der Nutzer keine Hemmungen mehr und kann besser mit dem Werkzeug arbeiten, da er voraus sehen kann, wo der Pfad verlaufen wird. Der Nutzer bekommt also das Gefühl, eine gewisse Kontrolle über den zu erstellenden Pfad zu haben. Der sofort sichtbare Verlauf der Kurve ist ebenfalls positiv, denn der Nutzer kann nun sofort sehen, ob die Kurve wie gewünscht verläuft oder direkt Änderungen vornehmen. Es besteht also die Möglichkeit zu agieren. Dadurch wird die Nutzung des Werkzeuges intuitiv, es muss nicht darüber nachgedacht werden, wo der Pfad vielleicht verlaufen wird oder was man tun muss bzw. wo man die

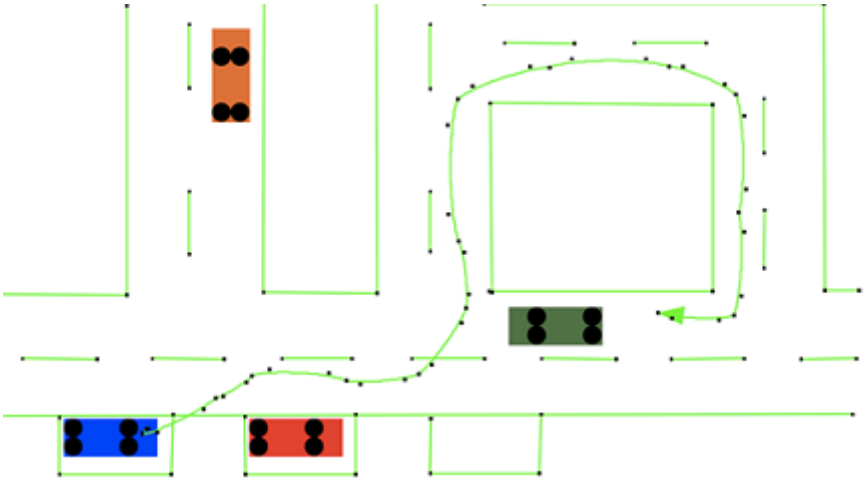

Abbildung 13: Beta-Version: Der Pfad verläuft wie die Versuchsperson es erwartet hat. Gut zu erkennen sind die Kontrollpunkte rechts bzw. links zwischen den gesetzten Punkten. Der Pfad ist zwar nicht komplett rund, jedoch gibt es keine starken Ausreißer in den Gegenverkehr oder Ähnliches.

Punkte hinsetzen muss, damit der gewünschte Verlauf auch entsteht. Alle VP haben diesen Aspekt positiv bewertet, da es nicht lange gedauert hat, sich mit dem Werkzeug vertraut zu machen.

Dennoch gibt es immer noch Aspekte, die den Nutzer verwirren können. Bei der Gamma-Version wird der Kurvenverlauf, je nachdem wie neue Punkte gesetzt werden, eventuell neu berechnet, wodurch manchmal doch noch nicht vorhersehbare Schlenker entstehen. Da diese aber nur klein sind, empfinden die VP es nicht als übermäßig verstörend, nur etwas eigentümlich.

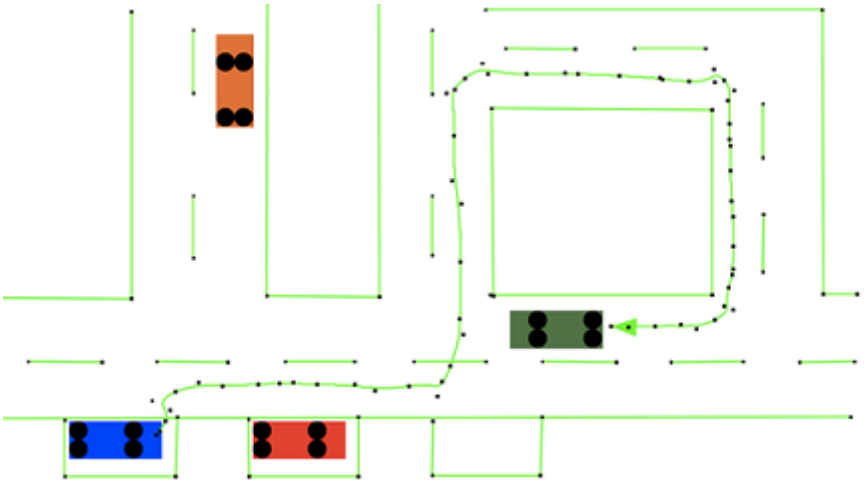

Abbildung 14: Gamma-Version: Auch hier hat die Versuchsperson den gewünschten Pfad erzielt. Der Pfad ist oft runder als in der Beta-Version, jedoch gibt es ab und zu Ausreißer in den Gegenverkehr oder Ähnliches. Auch hier sind die Kontrollpunkte gut zu erkennen.

8 Vergleich der drei Versionen (Neele Stoeckler)

Durch die Studien mit Versuchspersonen konnte festgestellt werden, dass sich die Bedienbarkeit und damit die Benutzerfreundlichkeit der Runden Pfade stark verbessert hat. Wie Abbildung 15 zeigt, ist es möglich, ganz neue Kurvenverläufe individuell zu erstellen.

Allerdings ließen sich nicht alle Defizite der ursprünglichen Version beheben. Während im direkten Anwendungsfall, bspw. bei einer Verkehrsszene, die Unterschiede zwischen der Beta- und Gamma-Version nur minimal auffallen, zeigen sich bei einer s-förmigen Kurve deutliche Defizite der Beta-Version. Bei starken Steigungsänderungen entstehen eckige Kurven, während bei der Gamma-Version ein fließender Übergang die Kurvenabschnitte miteinander verbindet.

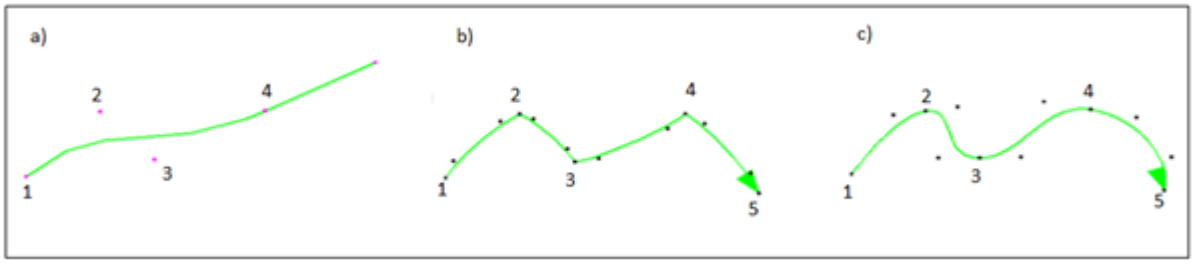

Abbildung 15: a) Alpha-Version, b) Beta-Version, c) Gamma-Version

Wie Abbildung 16 zeigt, lassen sich jedoch auch mit der Beta-Version Runde Pfade erstellen. Je nach der Verteilung der Punkte machen sich also Unterschiede bemerkbar. Die Schlenker und Ecken, die noch zu beheben sind, können sich negativ auf die Animationen auswirken. Beim Abspielen einer Animation irritieren die ruckartigen Bewegungen, besonders von Objekten mit vielen Ecken, den User.
Die Animationen sollen die Wirklichkeit so gut es geht, repräsentieren. Dies ist jedoch momentan noch nicht möglich, da Bewegungen z.B. von Robotern nicht wirklichkeitsgetreu wiedergegeben werden können.

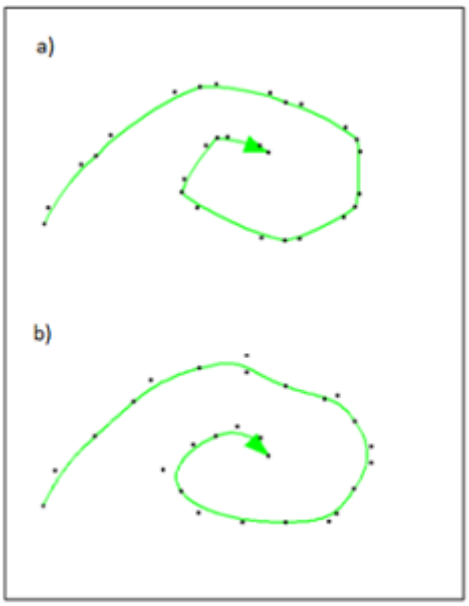

Abbildung 16: a) Beta-Version, b) Gamma-Version

Wie in der Nachstudie gezeigt werden konnte, kann es wünschenswert sein, Runde Pfade mit normalen Pfaden zu kombinieren. Soll ein Roboter sich auf einer geraden Linie fortbewegen wie das Auto im Experiment auf der Straße, empfindet der User es als störend, wenn Kurven gezeichnet werden. Besonders bei der Gamma-Version fallen die Kurven auf geraden Streckenabschnitten auf, wenn nur wenige Punkte gesetzt werden.

Wünschenswert wäre eine Möglichkeit, Pfade mit Ecken mit Runden Pfaden zu kombinieren. So könnte zu jedem Objekt in einem beliebigen Anwendungsbeispiel ein geeigneter Pfad erstellt werden. Denkbar wäre, den Pfad im Nachhinein glätten zu können. Das Verändern der Punkte sollte ebenfalls möglich sein um die Stärke der Krümmung flexibel an die Bedürfnisse des Users anzupassen.
Folgendes Beispielszenario mit einem Serviceroboter, dessen Wirkung mithilfe des Animationswerkzeuges untersucht werden könnte, zeigt die Erforderlichkeit, Pfade mit Ecken mit Runden Pfaden zu kombinieren. Ein Roboter, der sich in eine Warteschlange einreihen kann und soziale Regeln befolgt, wurde 2000 von Nakauchi und Simmons
[R. Simmons, Y. Nakauchi, 2000] entwickelt. Er interagiert nicht direkt mit den ihn umgebenden Menschen, muss sich aber dennoch an die Konventionen halten. Bei der Modellierung wurde berücksichtigt, wie Menschen Warteschlangen bilden und welchen persönlichen Raum es einzuhalten gilt.
Ein Roboter, der in einer Schlange steht, die gerade verläuft, sollte in der Lage sein, sich geradeaus zu bewegen ohne links- oder rechtsgekrümmte Wege zu gehen. Er sollte aber auch fähig sein, Hindernissen auszuweichen oder der Schlange zu folgen, wenn diese sich krümmt.
In einer solchen Umgebung, bspw. in einer Cafeteria ist ein soziales Verhalten des Roboters besonders wichtig. Abrupte Richtungswechsel und unvorhergesehene Schlenker würden Menschen in der Umgebung verunsichern, können aber auch erwünscht sein, bspw. wenn der Roboter aus der Schlange austritt um andere Personen durchzulassen. Bei der Übertragung der Situation ins Animationswerkzeug gelten selbstverständlich die gleichen Anforderungen. Gerade bei solch einer Simulation einer Situation muss die Repräsentation so nah an der realen Welt sein wie möglich, damit die Testpersonen so viel

wie möglich entlastet werden und sich die modellierte Situation besser vorstellen können. Situationen wie die oben beschriebene Alltagssituation zeigen die Wichtigkeit und Komplexität der Runden Pfade.

9 Fazit

Unser Ziel, die Runden Pfade benutzerfreundlicher zu gestalten, wurde insofern erfüllt, als dass der Pfad nun durch alle gesetzten Punkte verläuft und der Nutzer dadurch einen höheren Grad an Kontrolle über das Werkzeug gewinnt. Durch die richtungsanzeigende Pfeilspitze wird der Nutzer ebenfalls entlastet, da so immer klar ist, aus welcher Richtung der Pfad gerade kommt.
Es ist aber auch offensichtlich, dass es noch weitere Möglichkeiten zur Verbesserung des Werkzeuges gibt. Der Vergleich zu anderen Grafikprogrammen zeigt, dass beispielsweise die Möglichkeit einer nachträglichen Veränderung der gesetzten Punkte oder der Kontrollpunkte häufig genutzt wird um die Kurve entsprechend anzupassen und dies auch sehr positiv von den VP gesehen wird. Durch Gedrückthalten und Bewegen der Maustaste können hier Bézierkurven erstellt werden. Bereits bei unserer Pilotstudie, die mit Menschen ohne Vorerfahrung im Bereich der Grafikprogramme durchgeführt wurde, zeichnete sich jedoch ab, dass diese Möglichkeit den Laien überfordert. Bei der weiteren Gestaltung der Werkzeuge des Animationswerkzeuges sollte sich deshalb stärker an dem Nutzer und seinen Präferenzen orientiert werden.
Weitere Möglichkeiten zur besseren Bedienbarkeit könnten bspw. sein, den User die Stärke der Krümmung selber bestimmen zu lassen. Dies ließe sich insbesondere bei der Gamma-Version leicht realisieren. Über einen Dialog könnte der User einen Wert für die Stärke der Krümmung angeben und verschiedene Kurvenarten ausprobieren. Dieses Explorieren würde dem User diverse Möglichkeiten aufzeigen und wahrscheinlich auch für ein besseres Arbeitsergebnis sorgen. Durch das Explorieren steigert sich das Verständnis des Programms und erhöht sich das Gefühl der Kontrolle.
Wie im Vergleich zwischen den drei Versionen zur Erstellung eines Runden Pfades schon beschrieben, sollte es eine Möglichkeit geben, Runde und normale Pfade miteinander zu

kombinieren um besonders realistische Bewegungsabläufe zu simulieren. Wie eingangs beschrieben, herrscht reges Interesse an der Forschung im Bereich der Robotik und insbesondere im Bereich der Mensch-Roboter-Interaktion. Die Weiterentwicklung des Animationswerkzeuges ist also enorm wichtig um zukünftig noch mehr Tests und Studien mit dieser Art der Repräsentation durchzuführen.

Literatur

[Wik, a] http://en.wikipedia.org/wiki/Curve. Zuletzt abgerufen am: 12.3.2012.

[Wik, b] http://de.wikipedia.org/wiki/Kurve_%28Mathematik%29. Zuletzt abgerufen am: 12.3.2012.

[Wik, c] http://de.wikipedia.org/wiki/Parabel_%28Mathematik%29. Zuletzt abgerufen am: 12.3.2012.

[Wik, d] http://de.wikipedia.org/wiki/Polynom. Zuletzt abgerufen am: 12.3.2012.

[Wik, e] http://de.wikipedia.org/wiki/Spline. Zuletzt abgerufen am: 12.3.2012.

[Wik, f] http://de.wikipedia.org/wiki/Spline-Interpolation. Zuletzt abgerufen am: 12.3.2012.

[C. Bartneck, M. Saerbeck, 2010] C. Bartneck, M. Saerbeck (2010). Perception of affect elicited by robot motion. *Proceedings of the 5th ACM/IEEE International Conference on Human-Robot Interaction*, pages 53–60.

[C. Plaisant, B. Shneiderman, 2010] C. Plaisant, B. Shneiderman (2010). *Designing the User Interface - Strategies for effective Human-Computer-Interaction.* Amsterdam: Addison-Wesley, 5th edition.

[Dallmann, 2011] Dallmann, J. (2011). Evaluation der Softwarearchitektur. *Projektbericht Universität Hamburg.*

[Dr. O. Hoffmann,] Dr. O. Hoffmann. Skalierbare Vektorgraphik: Kurven und Pfade. http://hoffmann.bplaced.net/hilfe.php?me=svg&in=svgkurven. Zuletzt abgerufen am: 12.3.2012.

[F.C. Gee, W.N. Browne, K. Kawamura, 2005] F.C. Gee, W.N. Browne, K. Kawamura (2005). Uncanny valley revisited. *IEEE International Workshop on Robots and Human Interactive Communication*, pages 151–157.

[Irmen,] Irmen, K. Nicht Parametrische und Parametrische Kubische Splines. https://www.matse.rz.rwthaachen.de/dienste/public/show_document.php?id=7177. Zuletzt abgerufen am: 12.3.2012.

[M. Bril, M.Bender, 2005] M. Bril, M.Bender (2005). *Computergrafik - Ein anwendungsorientiertes Lehrbuch*. München: Carl Hanser Verlag, 2nd edition.

[M. Simmel, F. Heider, 1944] M. Simmel, F. Heider (1944). An experimental study of apparent behavior. *The American Journal of Psychology*, 57(2):243–259.

[Mori, 1970] Mori, M. (1970). The uncanny valley. http://www.androidscience.com/theuncannyvalley/proceedings2005/uncannyvalley.html. Zuletzt abgerufen am: 12.3.2012, Übersetzung von Karl F. MacDorman und Takshi Minato.

[Pepper, 2007] Pepper, P. (2007). *Programmieren lernen – Eine grundlegende Einführung mit Java*. Berlin Heidelberg: Springer Verlag.

[R. Simmons, Y. Nakauchi, 2000] R. Simmons, Y. Nakauchi (2000). A social robot that stands in line. *Intelligent Robots and Systems (IROS 2000) Proceedings 2000 IEEE/RSJ International Conference on*, 1:357–364.

[S. Greenberg, E. Sharlin, P. Saulnier, 2011] S. Greenberg, E. Sharlin, P. Saulnier (2011). Exploring minimal nonverbal interruption in hri. *20th IEEE International Symposium on Robot and Human Interactive Communication*, pages 79–86.

[Schäfermeier, 2011] Schäfermeier, S. (2011). Animationseditor: Anforderungen und Lösungen. *Projektbericht Universität Hamburg*.

[Staron, 2011] Staron, T. (2011). Der Grafipheditor und GEF: Die Umsetzung, das Modell und der Eigenschafteneditor. *Projektbericht Universität Hamburg*.

[TU Cottbus,] TU Cottbus. Parametrische Kurven und Flächen. http://www-gs.informatik.tu-cottbus.de/cg2_v09a.pdf. Zuletzt abgerufen am: 12.3.2012.

[V. Gaur, B. Scassellati, 2006] V. Gaur, B. Scassellati (2006). Which motion features induce the perception of animacy? http://cs-www.cs.yale.edu/homes/scaz/papers/Viksit-ICDL-06.pdf. Zuletzt abgerufen am: 12.3.2012.

[Vornberger, 2010] Vornberger, O. (2010). Computergrafik. www-lehre.informatik.uni-osnabrueck.de/misc/druck/computergrafik_final.pdf. Zuletzt abgerufen am: 12.3.2012.

[Wilms, 2011] Wilms, C. (2011). Ein Multi-Selection-Tool für GEF. *Projektbericht Universität Hamburg.*