Informationsvisualisierung VO/UE SS08

Benedikt Stehno, *0225175, 066932*
e0225175@student.tuwien.ac.at

Zusammenfassung des Papers

Animated Transitions in Statistical Data Graphics
Jeffrey Heer, George G. Robertson

Das Paper handelt von Animationsübergängen zwischen verschiedenen Visualisierungen von statistischen Daten wie, Piecharts, Barcharts und Scatterplots. Es wird auf die grundlegende Design-Prinzipien und darunterliegenden Theorien, um möglichst effektive und aussagekräftige Animationsübergänge zu erzielen, eingegangen und anhand von zwei kontrollierten Experimenten verifiziert, dass solche „Transitions" die Wahrnehmung der Daten empfindlich verbessern kann. Hierfür haben die Autoren des Papers ihr eigenes Visualisierungs-Framework, namens DynaVis, entwickelt.

Einführung
Zur Analyse und Präsentation zusammenhängender Datensätze werden oftmals verschiedenste Darstellungsformen wie Tortengrafiken oder Balkendiagramme herangezogen, um einzelne Aspekte eines Datensatzes zu visualisieren. Insbesondere bei Präsentationen ist es ein großes Ziel, dem Publikum den Zusammenhang zwischen dem vorhergehenden und nachfolgenden Diagramm möglichst effektiv nahezubringen, da ihnen die Interaktivität fehlt diese Zusammenhänge selbst zu entdecken. Animationen sind ein effektives Mittel um dies zu erreichen, wobei diese sich bei schlechtem Design als problematisch herausstellen können. Sie können aufgrund ihrer intuitiven und narrativen Natur dazu eingesetzt werden, um die Aufmerksamkeit zielgerichtet auf „***points of interest***" zu lenken, wobei insbesondere Bewegung die Aufmerksamkeit des Menschen fesselt. Veränderungen von Position, Größe, Form und Farbe von Objekten können dazu benützt werden, Zusammenhänge, insbesondere „***Ursache und Wirkung***" – Beziehungen, darzustellen.
Auch die Dauer einer Animationsübergangs spielt eine entscheidende Rolle. Dauert der Übergang zu lange, wirkt er langweilig, ein zu kurzer hingegen kann zu fehlerhaften Interpretationen führen. Die optimale Zeitdauer zu finden, hängt vor allem von der Komplexität der Grafik sowie von der Vertrautheit des Publikums mit solchen Visualisierungen.

Übergänge bei statistischen Daten Visualisierungen
Datendiagramme können auf drei Ebenen betrachtet werden: ***syntaktisch***, ***semantisch*** und ***pragmatisch***.
Animationen zwischen den Graphiken können als Zustandsänderungen auf diesen Ebenen formuliert werden Ebenen. Analytische Übergänge sind Änderungen des semantischen Modells der Graphik. Dazu zählen Änderungen des Visual Mappings, des Data Schemas oder der Werte an sich. Bei statischen Übergängen wird die ursprüngliche syntaktische Form einfach durch einen andere ausgetauscht.

Taxononmie von Übergängen
Im Originalpaper wurden Transitions in die nun folgenden Klassen eingeteilt:

View Transformation – eine Transformation des Viewports (Panning & Zooming) wobei dies keine syntaktische Änderung ist, da das Schema der Visualisierung unangetastet bleibt.

Substrate Transformation – dazu zählen Fisheye –Bifokallinsen und Achsenreskalierung.

Filtering - ändert das Schema der Grafik ebenfalls nicht, sondern fokussiert nur die im Moment wichtigen Objekte.

Ordering - betrifft die räumliche, nach einem Attribut sortierte, Anordnung von Elementen.

Timestep - zeigt die Änderung der Daten über einen gewissen Zeitraum.

Visualisation Change - das Visual Mapping des Datensatzes wird geändert. So wird z.B. ein Datensatz zuerst in einem Balkendiagramm dargestellt und später als eine Tortengrafik.

Data Schema Change – das Schema der Visualisierung und damit die Anzahl der Dimensionen ändert sich. Als Beispiel wird zuerst ein univariantes Balkendiagramm angezeigt, wobei eine zusätzliche Spalte des Datensatzes dargestellt werden soll. Diese bivariante Graphik kann nun ein Scatterplot, ein stacked oder grouped Barchart oder eine andere sein. Der Wechsel kann nun ***orthogonal***, wobei eine unabhängige Dimension hinzugefügt oder entfernt wird oder ***nested*** erfolgen, indem die hierarchische Relationen der Daten wiedergespiegelt werden.

Design Überlegungen für DynaVis
Die Autoren Papers halten sich die folgenden Guidelines, um die Animationen effektiv und sinnvoll für ihr Visualisierungs-Framework DynaVis zu gestalten:

Congruence
Maintain valid data graphics during transitions
Use consistent semantic-syntactic mappings
Respect semantic correspondence
Avoid ambiguity

Apprehension
Group similar transitions
Maximize predictability
Use simple transitions
Use staging for complex transitions
Make transitions as long as needed, but no longer

Implementation von DynaVis
In diesem Kapitel beschreiben die Autoren, wie sie, durch die (vorher schon zusammengefasste) „*Taxonomie der Transitions*" und die „*Design Guidelines*" geleitet, die einzelnen Animationen entworfen haben. Hierbei versuchten sie, möglichst alle Designratschläge einzuhalten, wobei dies nicht immer möglich war.
Erwähnenswert ist, dass alle Animationen in DynaVis slow-in/slow-out getimed werden. Beim ***Filtering*** wird Alpha-Blending eingesetzt, um die neuen Datensätze ein und auszublenden. Um Überlappungen bei ***Sorting*** entgegenzutreten, werden auf die Objekte verschiedene Delays angewendet, was das Visual-Tracking der einzelnen erleichtert. Weiteres werden beim Axis-Rescaling (***Substrate Transformation***) die alten Labels der Achsen langsam und weich ausgeblendet und die neuen später eingeblendet. Temporale Wertänderungen (**Timestep)** werden in DynaVis direkt animiert, wie z.B. die Änderung der Höhe eines Balkens, wobei bei stacked Barcharts, ebenfalls eine Translation hinzukommt. Die Größenänderung und Neupositionierung wird aber in einzelne Animationsabschnitte aufgeteilt. Um Okklusionen zu umschiffen, wurde aber mitunter auch zu sehr unintuitiven Animationen gegriffen, wie der „***multi-ring configuration for donut charts***". Bei der Änderung der Visualisierung (**Visualisation Change**) werden die Elemente sowohl in der Form, als auch ihre Position linear interpoliert, wobei es möglicherweise zu Überdeckungen kommen kann. Animationen sind bei **Data Schema Changes** nur dann sinnvoll einsetzbar, wenn beide Graphiken sich eine Datendimension teilen. Anderenfalls können sie zu Missverständnissen führen, da ein nicht vorhandener Zusammenhang zwischen den Graphiken durch die Animationen angedeutet wird. In diesem Fall wird empfohlen, eine normale Überblendung einzusetzen und auf Animationen zu verzichten.

DynaVis wurde in C# und in Verbindung mit dem Direct3D Framework entwickelt. Alle Animationen werden über einen Transition-Manager abgewickelt, der wiederum einzelne Animationsübergänge erstellt, so genannte „Transitioners", welche z.B. für die lineare Interpolation der Positionen zuständig sind. All

diese Animationen wurden „hand-coded“ in einer simplen Transitions-Language, die aus diesen einfachen Operatoren besteht.

Ergebnisse
Um zu testen, ob Animationen hilfreich sein können, führten die beiden Autoren zwei Experimente durch: Bei dem Ersten ging es um „***Object Tracking***“ (wo sich ein Element in der einen Grafik befindet und dann in der zweiten) und „***Estimating Changing Values***“ (es soll erschätzt werden, wie sehr sich ein Element prozentuell geändert hat).
Es stellte sich heraus, das Animationen eindeutig besser abschnitten als statische Überblendung der Grafik

Implementierungsideen/details
Grundsätzlich sollten einige aus dem Paper vorgestellten Transitions implementiert werden, wobei sich wahrscheinlich zeitlich nicht alle ausgehen werden. Vor allem die Animationen zwischen Barchart, grouped Barchart und Scatterplot haben Priorität, wobei auch das Hinzufügen („alpha-blending“) und Entfernen einzelner Elemente als Animation angedacht ist.
Das Programm wird in Java 1.5 entwickelt, da Java über eine mächtige Graphics2D-API verfügt. Diese ermöglicht es auch, beliebige Figuren zu zeichnen und zu füllen (GeneralPath), die aus Linien, quadratischen und Bezier Kurven bestehen können, sowie nachträgliche bzw. von Objekt unabhängige Transformationen durchzuführen. Es wird nur ein Grundelement geben, das über Getter- und Setter-Methoden die Form (Kreis, Balken oder Tortenstück) als auch Farbe und Position ändern kann. Die einzelnen Objekte sollen dann von einem Transition-Manager Objekt verwaltet werden, der auch die ganze Animation kontrolliert. Zusätzlich soll es noch eine Timerklasse geben, die von Transition-Manager verwendet wird. Es ist angedacht, dass die Daten aus einer Textdatei eingelesen werden.