MEHR AUF WEAVE.DE Weitere Infos gibt's unter www.weave.de/Domebook0313

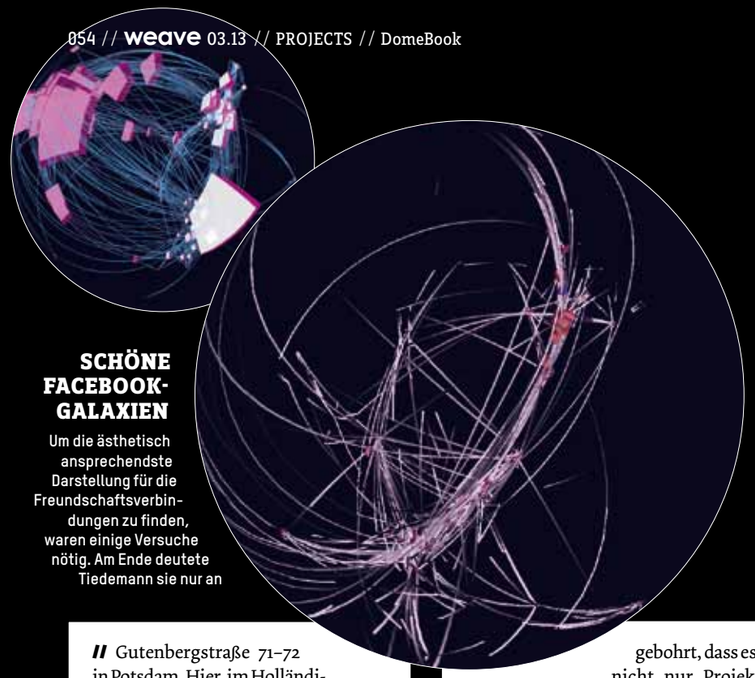
AGENTUR Marc Tiedemann, Berlin (www.marc-tiedemann.de)
HOCHSCHULE Fachhochschule Postdam, Fachbereich Interface Design (http://design.fh-potsdam.de/studiengaenge/interfacedesign.html)
PROJEKT Eine interaktive, immersive 3-D-Datenvisualisierung von Facebook-Inhalten
ZEITRAUM November 2012 bis Februar 2013
TECHNIK Processing (www.processing.org), toxiclibs (http://toxiclibs.org), GLGraphics (http://glgraphics.sourceforge.net)

FACEBOOK-BEZIEHUNGS-DATEN

Der Potsdamer Interface-Design-Student Marc Tiedemann experimentierte mit Facebook-Datensätzen einiger freiwilliger Spender. Das Ergebnis: DomeBook, eine Visualisierung jener sozialen Binnenverbindungen, die das soziale Netzwerk nicht anzeigt. Wie in einem Raumschiff kann man durch das Netzwerk steuern, als wäre man Captain James T. Kirk, und die Cluster unter die Lupe nehmen

URKNALL BEI FACEBOOK

Interface Designer Marc Tiedemann schuf ein interaktives dreidimensionales Sternensystem aus Facebook-Freunden, das die Strukturen des Netzwerks abbildet »



SCHÖNE FACEBOOK-GALAXIEN

Um die ästhetisch ansprechendste Darstellung für die Freundschaftsverbindungen zu finden, waren einige Versuche nötig. Am Ende deutete Tiedemann sie nur an

// Gutenbergstraße 71–72 in Potsdam. Hier, im Holländischen Viertel der Brandenburgischen Hauptstadt steht das Urania-Planetarium, das der Verein Urania »Wilhelm Foerster« Potsdam betreibt. Wo früher ein elektromechanischer Sternenprojektor ratterte, flimmern heute 3-D-Projektionen über die Innenwand der Kuppel. Fünf Beamer werfen nicht nur Bilder von Jupiter, Saturn, Mars und Co an die Halbkugel, sondern neuerdings auch interaktive Echtzeitvisualisierungen. Seit Oktober 2010 teilt sich der Trägerverein eine moderne, sogenannte 360-Grad-Fulldome-Projektionsanlage mit dem Interaction Design Lab der Fachhochschule Potsdam. »Zusammen mit dem Berliner Fraunhofer-Institut für Rechnerarchitektur und Softwaretechnik FIRST hat die Fachhochschule Potsdam das Fulldome-System so aufgebohrt, dass es nicht nur Projektionen abspielt, sondern auch beliebige visuelle Formate darstellt und mit Anwendungen in Echtzeit interagiert«, erklärt Boris Müller, Professor für Interaction Design.

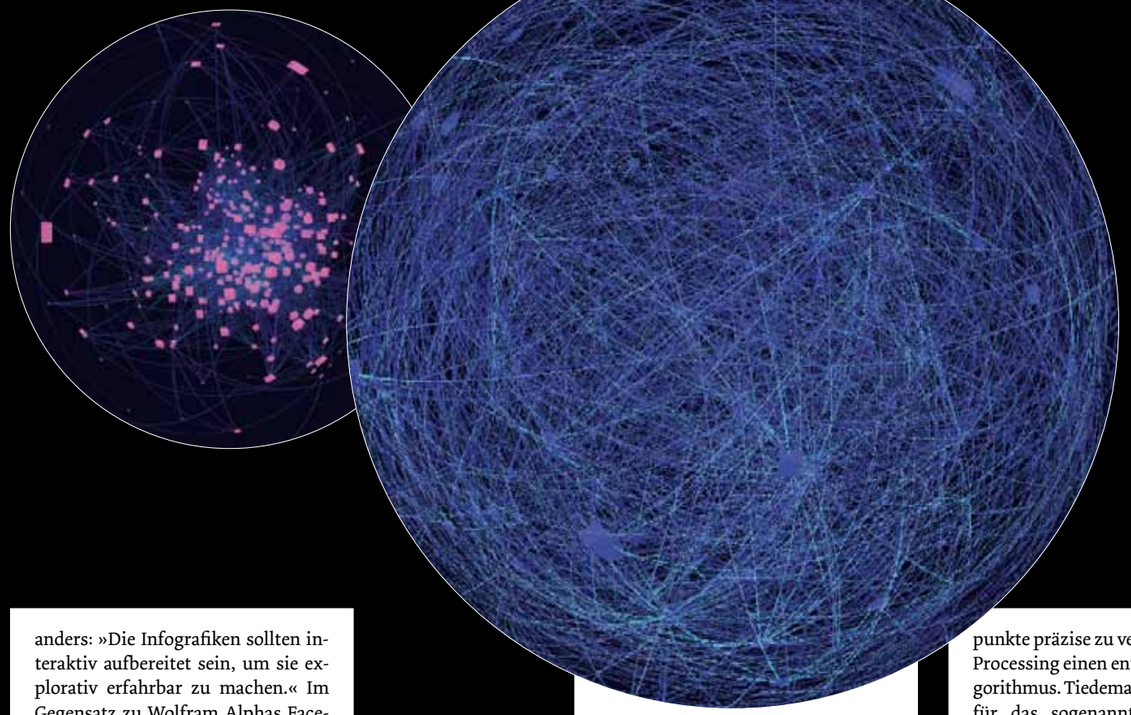
ERFOLGREICH VERKUPPELT

Dieses Modell zeigt die Powerdome-Konstruktion mit Kuppel und Auditorium

Gibt es magnetische Felder bei Facebook?

Seit Start der Anlage im Oktober 2010 entstehen in den Seminaren von Boris Müller und Motion-Graphics-Professor Klaus Dufke immer wieder neue Kuppelanwendungen wie das Projekt »DomeBook«, das Bachelorstudent Marc Tiedemann im Kurs Immersive Datenvisualisierung anfertigte. »Mich inspirierte das Analysetool Personal Analytics for Facebook von Wolfram Alpha«, verrät Tiedemann, der neben dem Designstudium mit Schwerpunkt Informations- und Datenvisualisierung auch für die Berliner Generative-Design-Agentur onformative als Entwickler arbeitet. Daneben betreibt er seit 2009 gemeinsam mit den beiden Musikern Gwydion ap Dafydd und Shai Levy das Unternehmen Konkreet Labs, das einen futuristischen Musik-Controller im App Store veröffentlicht hat.

Mit dem Analysetool des US-amerikanischen Suchmaschinenanbieters Wolfram Alpha (www.wolframalpha.com/facebook), das auf der Basis des persönlichen Facebook-Profils Diagramme und Schaubilder erstellt, können Nutzer ihre Netzwerk-Aktivitäten visualisieren. »Ich war extrem angetan davon, wie sich Cluster bilden und soziale Strukturen deutlich werden«, schwärmt Tiedemann. So etwas wollte er auch umsetzen – aber anders: »Die Infografiken sollten interaktiv aufbereitet sein, um sie explorativ erfahrbar zu machen.« Im Gegensatz zu Wolfram Alphas Facebook-Report, der erst seit Anfang 2013 dreidimensional ist, sollte Tiedemanns Visualisierung von Anfang an den gesamten Raum einnehmen. Tiedemann schwebte die Darstellung eines sozialen Netzwerks als Datenskulptur vor, die die Strukturen sichtbar und (be)greifbar macht.



Daten saugen, Leinwandwölbungen austricksen

Zunächst hieß es, Daten zu finden, auf denen die Anwendung basieren könnte. Facebook verfügt zwar über die Graph API (http://is.gd/LhoH96), über die man Anfragen starten kann – doch muss, wer mit seiner Software dort andocken möchte, sich wie bei jeder Programmierschnittstelle mit der Autorisierung und dem Sicherheitsschlüssel herumschlagen.

Einfacher geht es mit Facebooks Web-App »Give Me My Data« (http://givememydata.com), die ohne viel Federlesen eine Liste gemeinsamer Freunde, sogenannter Mutuals, ausspuckt – und zwar in verschiedenen Dateiformaten, sodass die Daten sich komfortabel weiterverarbeiten lassen. Sie zeigen zwar nur eine Momentaufnahme vom Zeitpunkt des Auslesens, liefern aber relativ schnell einen Datensatz, um einen ersten Prototyp zu formen. Da »Give Me My Data« nur den eigenen Account ausliest, startete Tiedemann über seine Homepage einen Aufruf und bekam so rund zehn Datensätze für sein Projekt. Mit ihnen fütterte er die grafische Entwicklungsumgebung Processing, um die gewünschte Visualisierung zu zeichnen.

Für das Innere einer Kugel muss man die Projektion vorverzerren, um den 3-D-Content perspektivisch korrekt darzustellen. »Für den immersiven Charakter der Visualisierung ist die perspektivisch korrekte Darstellung sehr wichtig«, weiß Tiedemann. Doch womit? Immerhin brütete er nicht als Erster über diesem Problem. Der Babelsberger Installations- und Performance-Künstler, Designer und Programmierer Christopher Warnow (http://christopherwarnow.com) hatte in seiner Studienzeit an der Fachhochschule Potsdam das FullDome-Template (http://is.gd/gUz98D) für das Urania-Planetarium geschrieben, das diese Berechnung ohne weitere Angaben durchführt. Es berechnet ein Bild aus der Perspektive von sechs virtuellen Kameras – ein Prozess, der selbst hochperformante Rechner in die Knie zwingt. Tiedemanns Lösung lautet Vertex Buffer Objects. Mit der Grafikbibliothek GLGraphics des amerikanischen Developers Andres Colubri lässt sich dieses OpenGL-Feature auf einfache Art und Weise nutzen, um Berechnungen vom Rechner auf die Grafikkarte auszulagern, sodass die CPU entlastet wird.

Ein Freundschafts-Algorithmus für Facebook

Doch sollte das Netzwerk nicht nur hübsch aussehen, sondern auch Facebook-Realitäten mathematisch verlässlich abbilden. Um die Netzwerkpunkte präzise zu verteilen, brauchte Processing einen entsprechenden Algorithmus. Tiedemann entschied sich für das sogenannte kräftebasierte Zeichnen (Force-Direted Graph Drawing) als Berechnungsmethode, bei dem er für die Anordnung der einzelnen Punkte und Verbindungen an- und abstoßende Kräfte definiert, auf deren Basis sich die einzelnen Punkte selbstständig im Raum organisieren.

Marc Tiedemanns pseudo-physikalisches System basiert auf zwei einfachen Regeln, nach denen a) alle Nodes voneinander abstoßen und b) Freunde sich gegenseitig anziehen. Dabei steht jeder Node für einen Facebook-Freund. Haben zwei Facebook-User gemeinsame Freunde (Mutuals), bekommen die beiden eine sogenannte Spring-Verbindung, die sie wie eine gestauchte Sprungfeder näher zusammenführt. In diesem selbstorganisierenden System entstehen nach und nach Freundesgruppen-Cluster.

Wie eine halbtransparente, farblich hinterlegte Ikone schweben die Profilfotos sämtlicher Freunde im Raum – Magenta für Frauen, Cyan für Männer. Zwischen ihnen sieht man die Verbindungslinien. Neben dem Freundschaftsstatus kann die App weitere Daten auslesen, beispielsweise sogenannte Affiliations, die Auskunft über Arbeitgeber oder besuchte Bildungseinrichtungen geben. Tiedemann sammelte die Orte separat, berechnete mithilfe der vorgegebenen Freundespostitionen einen Mittelwert und platzierte dort den Namen der Institution. Wie zu erwarten ist eine starke Korrelation von Freundesgruppen und besuchten Institutionen zu erkennen.

Fehlt nur noch das Lenkrad

Doch wie steuert man durch dieses Netzwerk? Tiedemann probierte es zuerst mit der 3-D-Maus SpaceNavigator des Münchner 3-D-Eingabegeräte-Herstellers 3Dconnexion (http://is.gd/RIf2zn), doch erwies sich die Navigation aufgrund der vielen Steuerungsoptionen – hoch/runter, vor/zurück, links/rechts, Rotation in alle Richtungen – als zu verwirrend. Mit Microsofts Bewegungssteuerung Kinect ließ sich nur ein kleiner Teil der Kuppel abdecken, sodass der Nutzer sich in einem begrenzten Bereich hätte aufhalten müssen. Um die Idee einer immersiven Erfahrung einzulösen, hätte man aufwendig drei Kinect-Einheiten zugleich nutzen müssen.

Die Fernbedienung Wii Mote schien zunächst alle Probleme zu lösen, doch in einer 360-Grad-Projektion sind die Orientierungspunkte – vorne und hinten, links und rechts – relativ. Was fehlt, ist eine absolute Orientierung. »Das brachte mich auf die Idee, den Kompass des iPhones zu nutzen und ihn mit gyroskopischen Daten zu verbinden«, verrät Tiedemann. Die App »gyrOSC« der Software-Schmiede BitShape in Milwaukee liest die Kompass- und Gyroskopdaten des iPhones aus und überträgt sie im Open-Sound-Control(OSC)-Format an den Rechner. Wer dann ein iPhone zur Hand nimmt, kann wie mit einem Laserpointer auf einen Bereich zielen und in diese Richtung fliegen. Tiedemann freut sich: »Diese Konstruktion ist für alle interaktiven Systeme spannend. Selbst klassische Sternensimulationen kann man mit diesem System völlig neu erfahrbar machen.« Na dann: Faust voran wie Superman … *as*

»

STRUKTUREN ERGRÜNDEN

Zusatzinformationen wie Ausbildungsstätten und Arbeitgeber werfen Licht auf die sozialen Strukturen eines Netzwerks wie Facebook

» FIXE TRICKS MIT TOXICLIBS

Um die schwebende 3-D-Cluster-Struktur der Facebook-Daten ästhetisch ansprechend und mathematisch korrekt zu animieren, ließ Marc Tiedemann seinen Processing-Sketch auf Basis eines pseudo-physikalischen Algorithmus zeichen

Marc Tiedemann, Interface Designer und Bachelor-Student im Kurs Immersive Datenvisualisierung an der Fachhochschule Potsdam
››› www.marc-tiedemann.de

// Mit Processing und der angeschlossenen Open-Source-Library-Sammlung für Java und Processing toxiclibs (www.toxiclibs.org) kann man leicht ästhetisch ansprechende 3-D-Netzwerk-Cluster zeichnen, wie wir sie auch hier brauchen. Toxiclibs enthät die Physik-Engine VerletPhysics – ein idealer Startpunkt für jegliche Art von Partikelsystemen, der alles bietet, was man zum kräftebasierten Zeichnen (Force-Directed Graph Drawing) braucht. Im Folgenden zeige ich Ihnen, wie Sie ein solches System aufsetzen.

PHYSIK-ENGINE INTEGRIEREN

1 Als Erstes initialisieren wir die Physik-Engine innerhalb der setup()-Funktion:

```
import toxi.physics.VerletPhysics;
VerletPhysics physics;
setup(){
...
physics = new VerletPhysics();
...
}
```

NODE-KLASSE ANLEGEN

2 Wir erstellen die Klasse FriendNode und erweitern sie um die VerletParticle-Funktionen ❶. Damit haben wir unsere eigene Node-Klasse, der wir nach belieben Eigenschaften und Funktionen mitgeben können, um sie als VerletParticle in unser Physiksystem zu integrieren. So könnten wir jedem Node einen Namen und ein Geschlecht mitgeben ❷ und ihn um eine draw-Funktion erweitern ❸.

```
import toxi.physics.VerletParticle;
class FriendNode extends VerletParticle { ❶
private String myName = ""; ❷
private int gender = 0;
FriendNode(float x, float y,float z) { ❶
super(x,y,z);
}
public void whoAmI(String myName, int
gender){ ❷
this.myName = myName;
this.gender = gender;
}
public void draw(){ ❸
pushMatrix();
translate(x,y,z);
switch (gender){
case 1: fill(255,0,205);
break;
case 2: fill(0,205,255);
break;
default: fill(255);
break;
}
box(10);
popMatrix();
}
}
```

KRÄFTE DEFINIEREN

3 Jetzt kommt der spannende Teil, das kräftebasierte Zeichnen des Netzwerks: Zunächst initialisieren wir für jedes Element in unserem Netzwerk (in diesem Fall für jede Person aus der Freundesliste) einen Node an einer beliebigen Ausgangsposition und geben ihm entsprechende Werte wie den Namen und das Geschlecht mit:

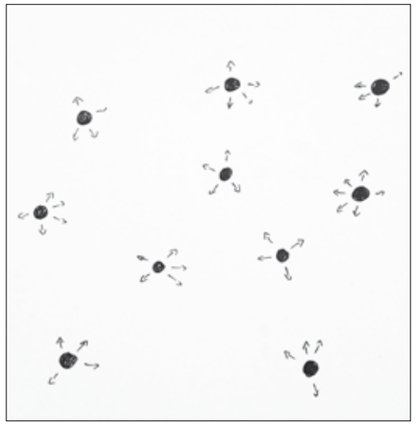
```
int initialSpread = 500;
int nForceRadius = 600;
float nForceStrenght = -0.5f;
float nJiggle = 0.008f;
friends = new FriendNode[friendData.length];
for(int i =0;i<friendData.length;i++){
float initialX = random(0,initialSpread);
...
friends[i] = new FriendNode(initialX,initialY,initialZ);
friends[i].whoAmI(friendsdata.name,
friendsData.gender);
```

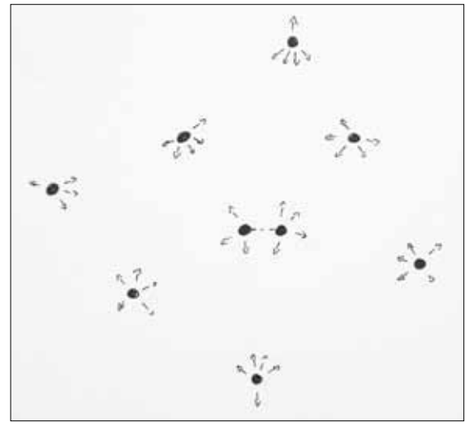
Danach müssen wir dafür sorgen, dass sich die Nodes prinzipiell voneinander abstoßen. Zu diesem Zweck nutzen wir das sogenannte AttractionBehavior. Dieses wirkt von einem beliebigen Punkt aus anziehend oder abstoßend auf sämtliche Nodes im System. Also initialisieren wir für jeden Node einen AttractionBehavior und geben anstelle einer festen Position den Node mit. Auf diese Weise bleibt dieser fest an die Node-Position gekoppelt. Dann legen wir noch die Reichweite der Kraft in Pixeln und die Stärke der Kraft in numerischen Werten von null an aufwärts fest, wobei positive Werte anziehen und negative abstoßen. Als Letztes definieren wir noch die Stärke des Jitters. Je höher dieser ist, desto mehr »zittert« die Kraft. So gibt man dem System eine leichte Grundbewegung, um es realistisch und nicht unnatürlich starr wirken zu lassen.

```
...
AttractionBehavior negativeForce = new AttractionBehavior(
friends[i], 600, -0.5f, 0.008f);
...
```

Sind alle Parameter gesetzt, übergeben wir das Behavior der Physik-Engine.

```
physics.addBehavior(negativeForce);
```



Geschafft, jetzt sollten sich sämtliche Nodes gleichmäßig im Raum verteilen.

ANZIEHUNGSKRÄFTE DEFINIEREN

4 Weiter geht es mit der eigentlichen Netzwerkbildung: Wir wollen, dass Freunde, die sich kennen, auch im Raum nahe beieinander stehen. Also brauchen wir eine Kraft, die diese aneinanderzieht. Hierfür nutzen wir einen VerletSpring ❶. Man kann ihn sich wie eine Sprungfeder vorstellen, die zwischen zwei Punkten gespannt wird. In diesem Beispiel kennen sich Freund eins und Freund sechs, also übergeben wir diese dem VerletSpring ❸ und legen eine Distanz fest, auf die sie »gezogen« werden sollen ❹, sowie eine Kraft, mit der dies geschieht ❺. Das Ganze übergeben wir dann der Engine.

```
FriendNode friendA = friends[6];
FriendNode friendB = friends[1]; ❷
VerletSpring ❶ connection = new Verlet
Spring( ❷ friendA,friendB, ❹ 100, ❺ 0.0008f);
physics.addSpring(connection);
```

Bevor wir unser Partikelsystem zeichnen, müssen wir noch die Physik-Engine updaten.

```
void draw(){
physics.update();
for(int i=0;i<friends;i++)friends.draw();
}
```

Um eine wirklich schöne Netzwerkvisualisierung beziehungsweise die gewünschte Clusterbildung darzustellen, muss man entsprechend viele Verbindungen an das System übergeben. Beim DomeBook waren es rund 200 Personen mit insgesamt 2500 Verbindungen, die ich mithilfe der Facebook-App »Give Me My Data« im JSON-Format exportiert und in Processing eingelesen habe. Dies ist wirklich nur ein Beispiel, Sie können mit den Werten beliebig experimentieren und erhalten andere sicherlich ebenso spannende Muster. *Marc Tiedemann (as)*

KRÄFTEVERHÄLTNISSE

Die schematische Darstellung zeigt das kräftebasierte Netzwerk in verschiedenen Aggregatzuständen von links nach rechts: Mit der gleichen abstoßenden Kraft auf allen Punkten, dann mit einer einzigen Spring-Verbindung und schließlich als formierte Cluster auf Basis gemeinsamer Freundschaften