Blender2Cinema4D-Fluid-Tutorial

Fluidsimulationen sind eine spanndende Sache. Leider hat Cinema 4D bis dato noch keine Möglichkeit diese Simulationen programmintern ablaufen zu lassen.
Eine kostenlose Möglichkeit Fluids nun auch in Cinema 4D zu nutzen ist der Import von Fluidsimulationen aus Blender (seit Version 2.40 möglich).

Zusammenfassung:

Wir werden zuerst eine einfache Fluid-Szene in Cinema 4D erstellen, diese dann nach Blender importieren, die Fluidsimulation ausführen und zuletzt die Fluid-Meshes wieder nach Cinema 4D importieren und rendern. Das importierte Mesh wird über eine XPression animiert und, bei Bedarf (wenn der Speicherplatz knapp wird) noch mit dem X-Linky-Plugin in externe Referenz-files gebacken.
(Ich werde hier nicht triviale Sachen erklären wie Cinema 4D-Plugin-Installation oder wie man eine Animation in Cinema 4D rendert. Dazu schaut Ihr Euch bitte das Cinema 4D-Handbuch an oder andere Tutorials im Web.)

Bitte arbeitet vorher auch die Blender-Fluidtutorials durch (zumindest das Tutorial 1 http://mediawiki.blender.org/index.php/SoCFluidTutorial1), um ein bisschen mit Blender und der Fluid-Simulation umgehen zu können!

Beschränkungen:

Zur Erstellung dieses Tutorials gibt es noch einige Beschränkungen bzw. noch nicht implementierte Funktionen in der Blender Fluid-Simulation.

Da das Programm aber gratis ist und die Version 2.40 die erste Version ist, in der die Fluid-simualtion überhaupt erst richtig funktioniert, sollte man darüber aber hinwegsehen können und gespannt sein, was in zukünftigen Blenderversionen noch kommt.

Voraussetzungen:

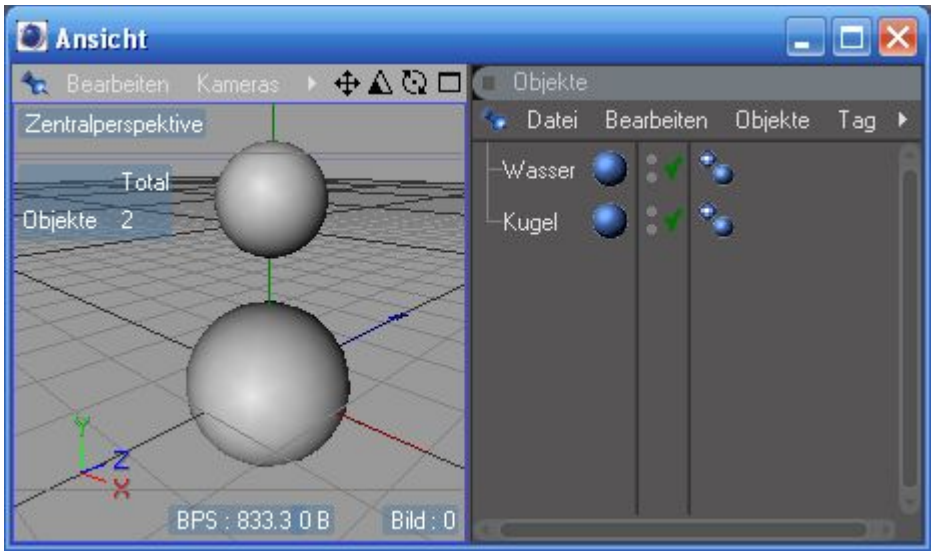
Für eine erfolgreiche Umsetzung des Tutorials sind erforderlich:

- **Cinema 4D** (die Version muss Xpresso unterstützen) empfehlenswert ist **Version 9.5**, da der Meshimport ohne Content-browser zur Qual wird.
- Blender Version 2.40: Download hier: http://www.blender.org
- Ein neues **Blender Ani-Obj-Export-Skript.** Die Erstellung wird im Tutorial erklärt.
- Die **Import-Mesh-Animations-Xpression-Datei** (Vielen Dank an Spekulatius aus dem C4D-Treff)

  Download hier: http://www.jan-wesbuer.de/3d/import-mesh-xpress.c4d
- Optional (für grössere Animationen): Das **X-Linky-Plugin**
  Download hier: http://www.thirdpartyplugins.com

Part 1: Cinema 4D Fluid-Szenen-Erstellung

1. Wir erstellen eine einfache Szene in Cinema 4D mit 2 Kugeln, die obere Kugel wird das Wasser, die andere ein einfaches Hindernis für das herunterfallende Wasser.



   Es müssen keine Materialien vergeben werden, da diese durch den Export/Import sowieso verloren gehen bzw. wird hinterher ja nur noch das Fluid-Mesh in diese Szene hinzugeladen. Es stört also auch nicht, wenn schon Materialien vergeben wurden. Diese Szene wird zur Sicherheit als C4D-File abgespeichert, denn wir brauchen sie später ja noch wieder.

Part 2: Export aus Cinema 4D

1. Wir stellen in den Programmvoreinstellungen den Wavefront-Import- und Wavefront-Export-Faktor auf 100.
2. Nun wird die Szene als Wavefront-file (*.obj) exportiert.
   (am Besten einen neuen Ordner dafür anlegen, denn später kommen evtl. noch hunderte Files dazu.)

Part 3: Vorbereitungen für Blender

1. Für den späteren Export brauchen wir eine Anpassung am Wavefront-Obj-Export:

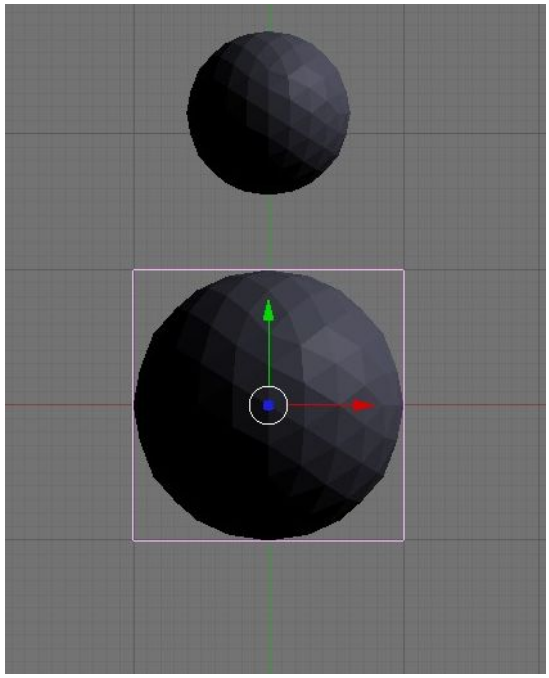
   Den auszutauschenden Code-Schnipsel dazu findet Ihr hier:
   http://www.elysiun.com/forum/viewtopic.php?t=56824
2. Das zu bearbeitende Skript findet Ihr in Eurem Blender-Installations-Ordner:
   (Wahrscheinlich lautet er C:Progamme\Blender Foundation\Blender\.blender\scripts)
   Dort findet Ihr eine Datei mit Namen: obj_export.py

   Erstellt eine Kopie dieser Datei im gleichen Ordner und nennt sie in obj_fluid_export.py um.
3. Nun öffnet die Datei obj_fluid_export.py mit einem Texteditor Eurer Wahl
   (Sehr zu empfehlen ist z.B. dieser Gratis-editor: http://notepad-plus.sourceforge.net , es tut aber auch das normale Windows Notepad)
4. Jetzt tauscht am Anfang des py-Files die Zeile:
   Name: 'Wavefront (.obj)...'
   gegen
   Name: 'Wavefront-Animation (.obj)...'

5. Dann noch ein paar Zeilen darunter:
   Tooltip: 'Save a Wavefront OBJ File'
   gegen
   Tooltip: 'Save a Wavefront OBJ Animation Files'
6. Und zuletzt die Zeile (ganz am Ende der Datei)
   Window.FileSelector(save_obj, 'Export Wavefront OBJ', newFName('obj'))
   gegen den Code-Schnipsel aus dem Forum.
7. Nun die Datei abspeichern
8. Spätestens jetzt solltet Ihr Euch mal das Blender Tutorial 1 aneignen um noch zügig weiter zu kommen. (Link zum Blendertutorial findet Ihr am Anfang dieses Tutorials)

Part 4: Import nach Blender

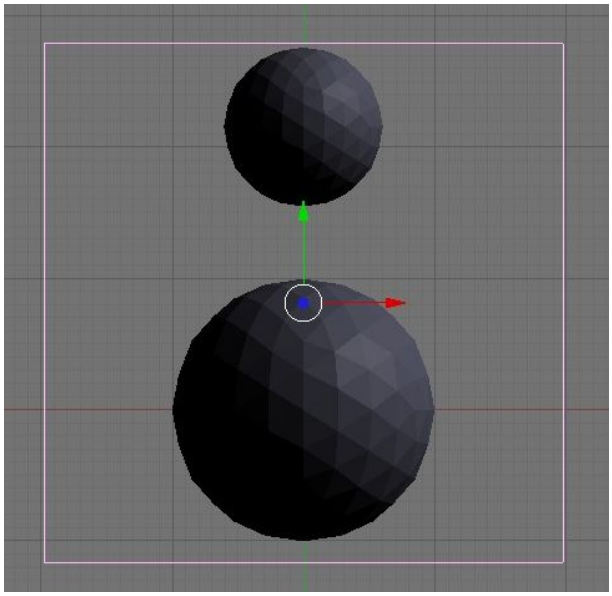
1. Blender starten.
2. Stellt die Bildanzahl der Animation auf 250 Bilder.
3. Das aus Cinema 4D exportierte Wavefront Obj-file in Blender importieren über: Datei/import/Wavefront-Obj
4. Wir stellen den Standard Blenderwürfel auf Wire-Darstellung.
5. Auf den ersten Blick sieht der Import gut aus, jedoch ist das Mesh um 90 Grad verdreht. Das fällt einem zwar nicht sofort auf, aber wir schauen gerade von oben auf die Szene.
6. Bitte lasst die importierten Meshes unskaliert, unverschoben und unrotiert, also so wie es ist.
7. Es sollte jetzt ungefähr so aussehen:



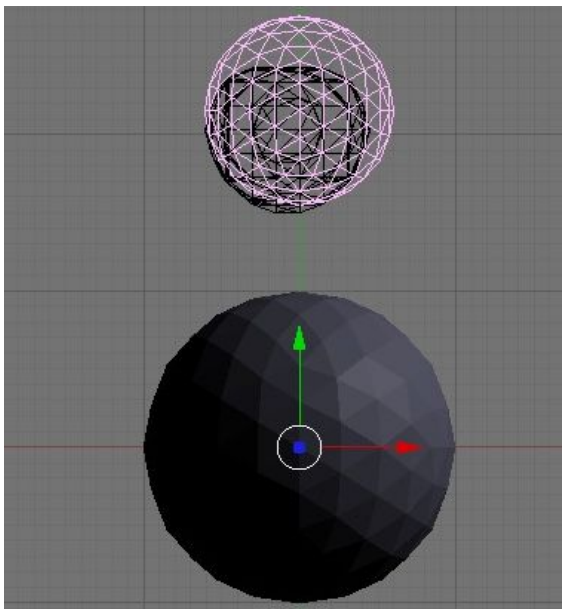
Part 5: Fluidsimulation in Blender

1. Skaliert den Würfel nun mindestens so gross und verschiebt ihn, so dass alles was mit Wasser interagieren soll in den Würfel passt. In diesem Bereich wird dann die Simulation ablaufen.

2. Wählt den Bereich aber andererseits nicht zu gross, weil Ihr sonst eine höhere Auflösung bei der Simulation benötigt, oder aber das Fluid-mesh wird eine sehr geringe Auflösung haben. Die Szene sollte jetzt z.B. so aussehen:

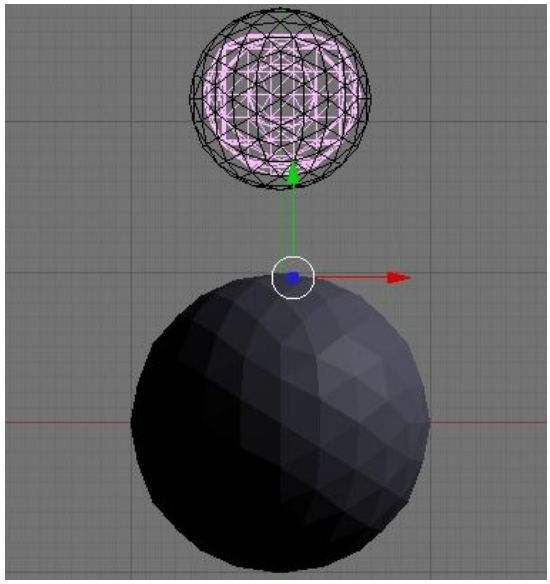


3. Nun müssen wir noch den Objekten Ihre Aufgaben zuweisen.
    - Stellt im Fuid-tab für die obere Kugel „Fluid" ein
    - für die untere Kugel „Obstacle" (Hindernis)
    - Für den Würfel „Domain" (die Simualtions-umgebung)
4. Nun stellen wir noch für die Domain im Advanced-Feld die Gravity um (wegen der 90° Verdrehung) und zwar auf:
   X:0.00 Y:-9.81 Z:0.00
5. Jetzt könnt Ihr die Simulation mit „bake" starten.
6. Vielleicht ist Euch aufgefallen, dass die Flüssigkeit nicht wirklich am Startpunkt der Kugel losfliesst. Erkennen könnt Ihr das gut, indem Ihr die Fluid-Kugel auf Wire stellt:



Da das nicht optimal ist (Bug?) werden wir nun noch das Fluid-mesh an seine korrekte Ausgangsposition verschieben. Dies solltet Ihr sowohl in der Oben- als auch der Seiten-Ansicht machen.

7. Wenn Ihr jetzt mit „alt+a" noch mal die Animation kontrolliert, werdet Ihr erkennen können, dass es so viel besser aussieht:



8. Zu guter Letzt löschen wir noch alle Mesh-Objekte ausser dem Fluid-Mesh (da sie sonst später auch mehrfach nach Cinema 4D importiert würden und dies ja nicht erwünscht ist. Ausserdem haben wir ja noch die Original-C4D-Szene, gelle?);)

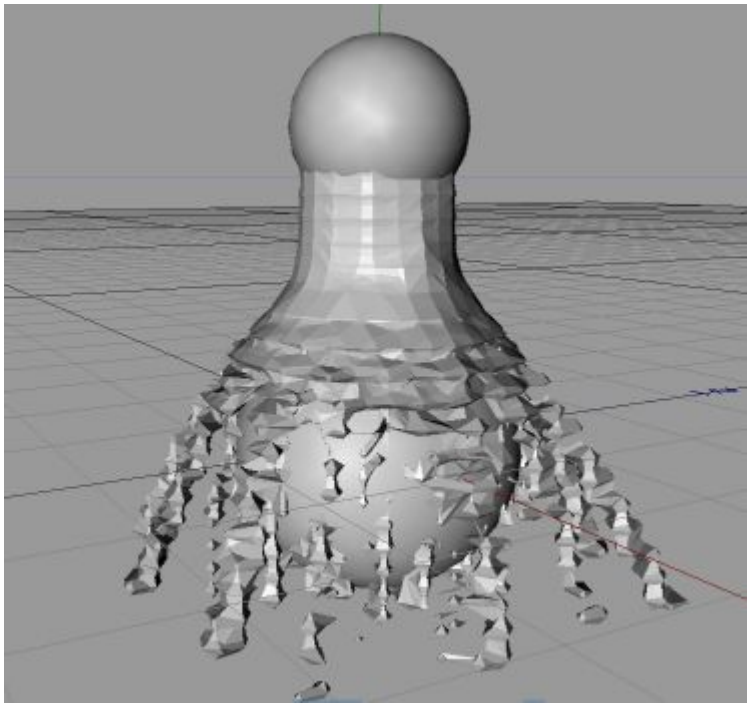
Part 6: Ani-Export aus Blender

1. Wir verabschieden uns von Blender, indem wir mit unserem neuen Export-skript über die selbsterstellte Exportfunktion „Save a Wavefront OBJ Animation Files" die Animation exportieren.
2. Die exportierten Mesh-Dateien liegen nun im Ordner C:Temp\ vor.
   (Das ist jetzt so etwas Ähnliches wie eine Einzelbildsequenz für 3D-Dateien.)

Part 7: Ani-Import in Cinema 4D

1. Nachdem Ihr Euch genug mit Blender auseinandergesetzt habt (und das war bestimmt eine komische aber auch interessante Erfahrung, wollen wir uns doch wieder unserem geliebten Cinema 4D zuwenden.
2. Nun wird die Original C4D-Datei in Cinema 4D geöffnet.
3. Setzt bitte in den Dokument-Voreinstellungen die Maximum-Bildanzahl auf die, die Ihr in Blender hattet (hier im Beispiel 250).
4. Wir öffnen den Content-Browser und gehen zum Ordner C:Temp.
5. Ihr stellt die Ansicht auf „Liste" und „sortiert nach Art" um
6. Wir importieren die Fluid-Dateien, indem wir alle Obj-Dateien auswählen und dann per Rechtsklick hinzuladen (ohne die mtl-dateien).
   Es kann sein, dass der Vorgang ein bisschen dauert...

7. Das Ganze sollte im Editor nun ungefähr so aussehen:

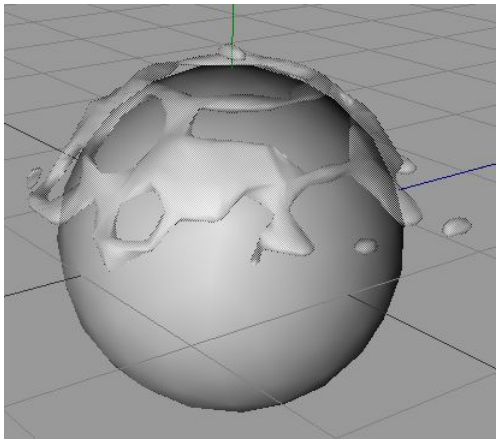


Das ist eine durchaus interessante Sache, aber nicht das, was wir wollen: alle Frame-Meshes werden gleichzeitig angezeigt.

8. Wir bereiten die Meshes weiter vor, indem wir alle Normalen- und Material-Tags der importierten Meshes löschen. Danach führen wir noch die Funktion „Unbenutze Materialien löschen" im Materialmanager aus.
9. Jetzt noch alle Phong-Tags markieren und „Glätten bis" auf 180° stellen.

Part 8: Meshanimierung über Xpresso

1. Nun laden wir die Datei: import-mesh-xpress.c4d hinzu.
2. Wir ziehen alle importierten Meshes in das Blender-Fluid-Null-Objekt. Jetzt solltet Ihr direkt die Animation sehen.
3. Die Original-Wasser-Kugel aus Cinema 4D könnt Ihr natürlich jetzt löschen.
4. Jetzt evtl. das Blender-Fluid-Null-Objekt noch in ein Hypernurbsobjekt mit Hypernurbs-Unterteilung 1 oder 2 schieben und Ihr fügt noch ein paar passende Materialien hinzu und dann sieht es schon bei mir im Editor beim Frame 54 z.B. so aus:



Part 9 (optional): Backen der Animation mit X-Linky-Plugin

Bei grösseren Animationen oder höher aufgelösten Fluid-Meshes wird schnell der Speicher in Cinema 4D knapp, da das Mesh ja vervielfacht vorliegt. Ein 20.000 Polygon-Fluid wird so schnell, bei einer 250 Frame Animation, zu 20.000 x 250 = 5.000.000 Polygonen, was auch den speicherstärksten Rechner ins Schwitzen bringt oder aber unmöglich ist zu rendern.

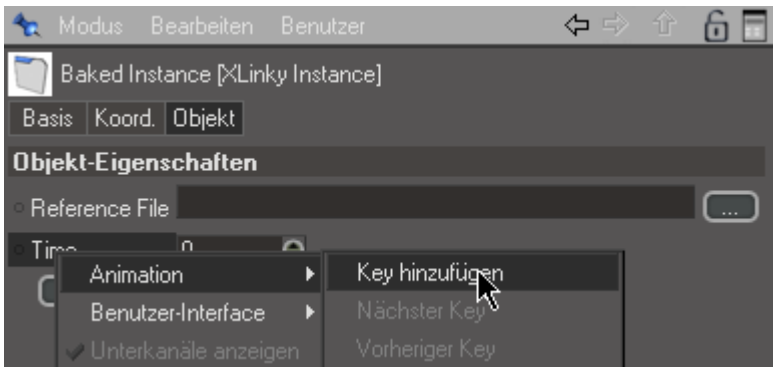
Zum Glück gibt es für diesen Zweck das **X-Linky-Plugin**! Diese Pugin backt jedes Frame der Mesh-Animation wiederum in ein externes File. Dieses wird erst zum dem Frame der Animation in Cinema 4D geladen, bei dem es in Cinema 4D im Editor oder beim Rendern gebraucht wird. So muss zwar immer ein Frame-Mesh nachgeladen werden, aber im Speicher sind bei unserem Rechenbeispiel immer nur 20.000 Polygone.

(Auch hier entsteht dadurch schnell eine grosse Menge an Speicherbedarf, doch meistens ist der Festplattenspeicher ja in ausreichender Menge verfügbar)

1. Wir bereiten nun das Mesh vor, indem wir das Material-Tag von dem Blender-Fluid-Null-Objekt löschen (auch die einzelnen Frame-Meshes dürfen **KEIN** Material-Tag besitzen. Ansonsten ist später **keine** Material-zuweisung bei dem Xlinky-Instance Objekt möglich.)
2. Wir markieren das Blender-Fluid-Null-Objekt und führen das Plugin „XLinky-> Bake Mesh Animation to XREFS" aus.

   Es wird nach einem Speicherpfad für die XREFS gefragt. Sucht Euch einen Pfad mit möglichst viel Speicherplatz aus.
   Der Back-Vorgang beginnt und nimmt einige Zeit in Anspruch.
3. Wir löschen das Blender-Fluid-Null-Objekt und führen das Plugin „XLinky-> Xlinky-Instance" aus. Ein neues Objekt erscheint im Objektmanager.
4. Wir geben der XLinky-Instance im Attribute-Manager unter Reference-File den Pfad zum ersten gebacken Frame-Mesh-File an. (*.bmo)
5. Nun fehlt nur noch die Animation der XLinky-Instanz:

   Wir gehen in der Zeitleiste auf den Frame 0 machen im Attribute-Manager einen Rechtsklick auf „Time" und fügen einen Key hinzu:



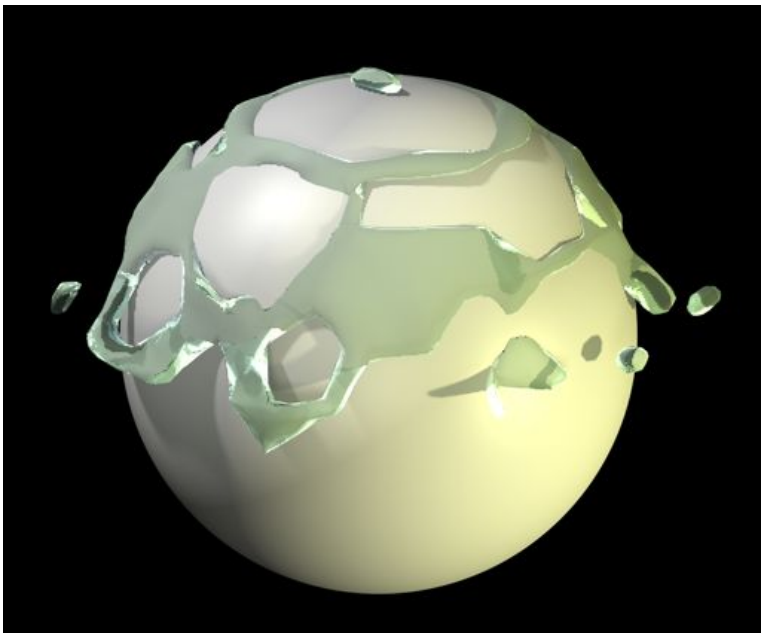
   und danach gehen wir in der Zeitleiste auf das Frame 250 und machen danach einen Rechtsklick auf „Time" und fügen auch hier noch einen Key hinzu.
6. Nun sollte das Fluid-mesh auch über XLinky wunderbar animiert sein und nimmt wenig RAM-Speicher in Anspruch.

   (Natürlich kann man evtl. nicht immer die ganze Fluidsimulation importieren. Hierzu muss man dann einen Teil der Animation importieren, dann backen, noch einen Teil importieren, wieder backen usw... So dass man im Endeffekt die komplette Animation gebacken hat und dann doch über das Xlinky-Instanz-Objekt referenzieren kann)

Part 10: Rendern in Cinema 4D

Der einfachste Part! Den Cinema 4D Render-Button drücken, die Animation rendern lassen und sich freuen. ;)

Dies war jetzt natülich nur ein ganz simples Beispiel mit den Default-Einstellungen bei der Fluid-Simulation. Ihr solltet ruhig mal mit den Simulations-Parametern rumspielen. Vor allem sorgt eine höhere Auflösung bei der Simulation für bessere Resultate.



Viel Spass beim Ausprobieren wünscht

Jan Wesbuer

Hinweis:

Ich verbessere bzw. aktualisiere das Tutorial gerne.
Für Feedback bzw. Verbesserungsvorschläge also bitte an me@jan-wesbuer.de mailen.