Vortex

Die Physik-Engine von CMLabs

14.11.03

Jan-F. Steffen

Was ist Vortex?

- Real-time Physik-Engine
- ermöglicht interaktive 3D-Simulation
- simuliert natürliches Verhalten von Objekten in einer physikalischen Welt
- verhindert Durchdringen graphischer Objekte
- hat selbst keine Grafik-Engine / keinen SzenenGraph

Was ist Vortex?

- enthält Bibliotheken für:
  - robuste Festkörper-Dynamik (*rigid body dynamics*)
  - Kollisions-Erkennung (*collision detection*)
  - Kontakt-Erzeugung (*contact creation*)
  - Auswirkungen von Kollisionen (*collision response*)

Aufbau

- Unterteilung in drei Module:
  - **MCD** (Collision Detection Module)
    - Algorithmen für *collision detection* und *contact creation*
  - **MDT** (Dynamics Tools Module)
    - Spezifikation der physikalischen Interaktion zwischen Objekten, *Constraints* und der Umwelt
  - **MST** (Simulation Tools Module)
    - stellt Verbindung zwischen *MCD* und *MDT* her

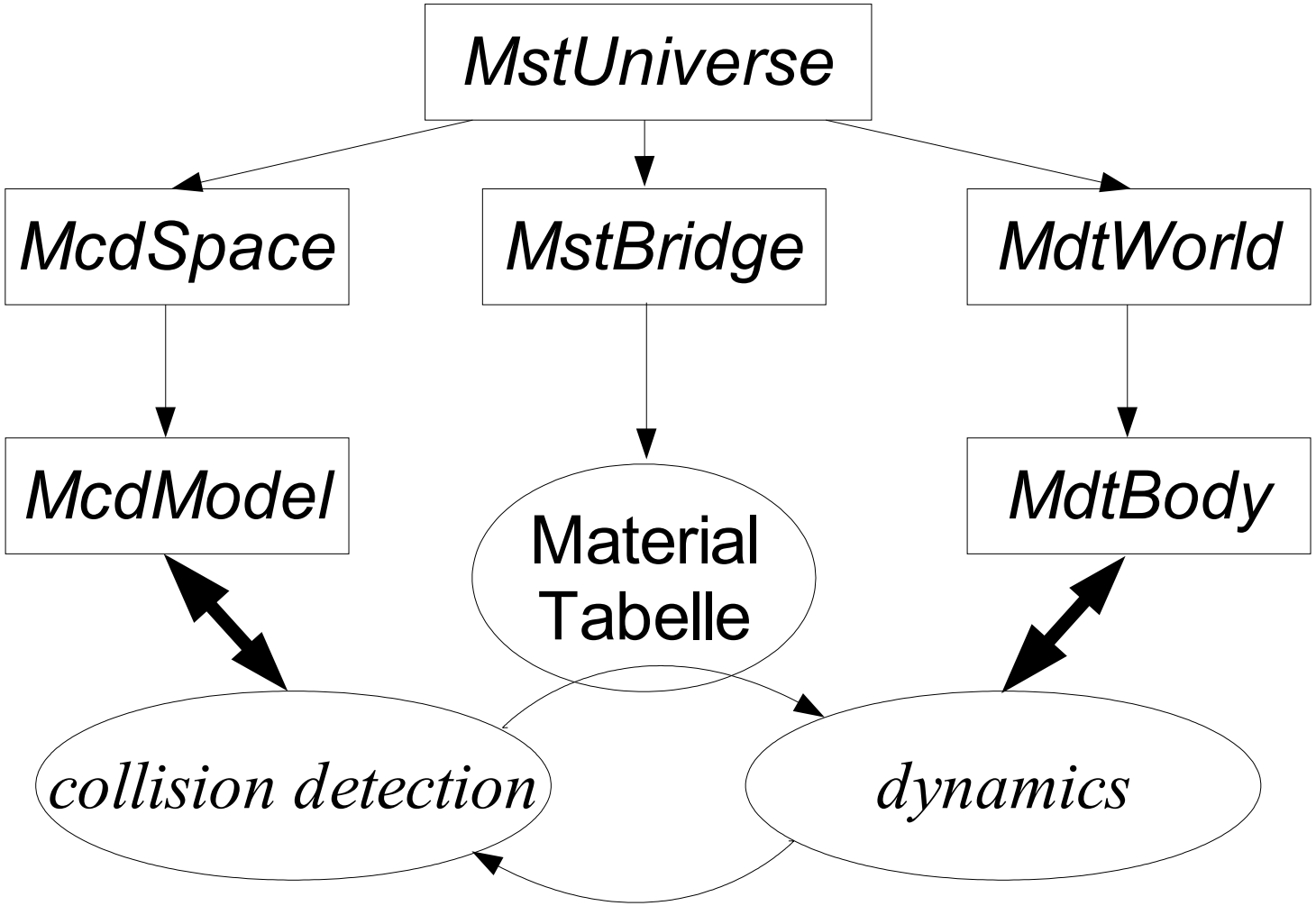
Wichtige Bestandteile

- *McdModel*
  - repräsentiert das zu simulierende Objekt als Körper im Raum (Geometrie, TM, ..) für die *collision detection*
- *MdtBody*
  - repräsentiert die physikalischen Eigenschaften eines Körpers für die Bewegung
- *McdModel* und *MdtBody* zusammen ergeben komplette Repräsentation eines *rigid body*
- *McdSpace*
  - Eigenschaften einer 3D-Region mit *McdModel*s

Wichtige Bestandteile

- *MdtWorld*
  - Liste von *MdtBody*'s und *Constraints* (*Joints* u. Kontakte)
  - definiert Gleichungssystem für Bewegung und Kräfte
- *MstBridge*
  - Material-Tabelle mit Kontakt-Eigenschaften (Reibung etc)
- *MstUniverse*
  - *McdSpace*, *MdtWorld* und *MstBridge*

Zusammenhänge



Constraints in Vortex

- Relationen zwischen zwei Körpern
  - beschränken relative Position und/oder Orientierung
- *Constraint*-Typen in Vortex:
  - *Joint* (Gelenk / Befestigungspunkt)
  - *Contact*

Joints

- Befestigung zweier Körper aneinander bzw. eines Körpers in der Welt
- jeder Körper hat <u>einen</u> Befestigungspunkt
- mindestens ein Freiheitsgrad fest relativ zum 2.Körper
- immer oberes <u>und</u> unteres Limit
- beschränkt z.B. Position, Winkel, MaxSpeed,...
- Satz von Bedingungen, hat <u>keinen</u> eigenen Körper!

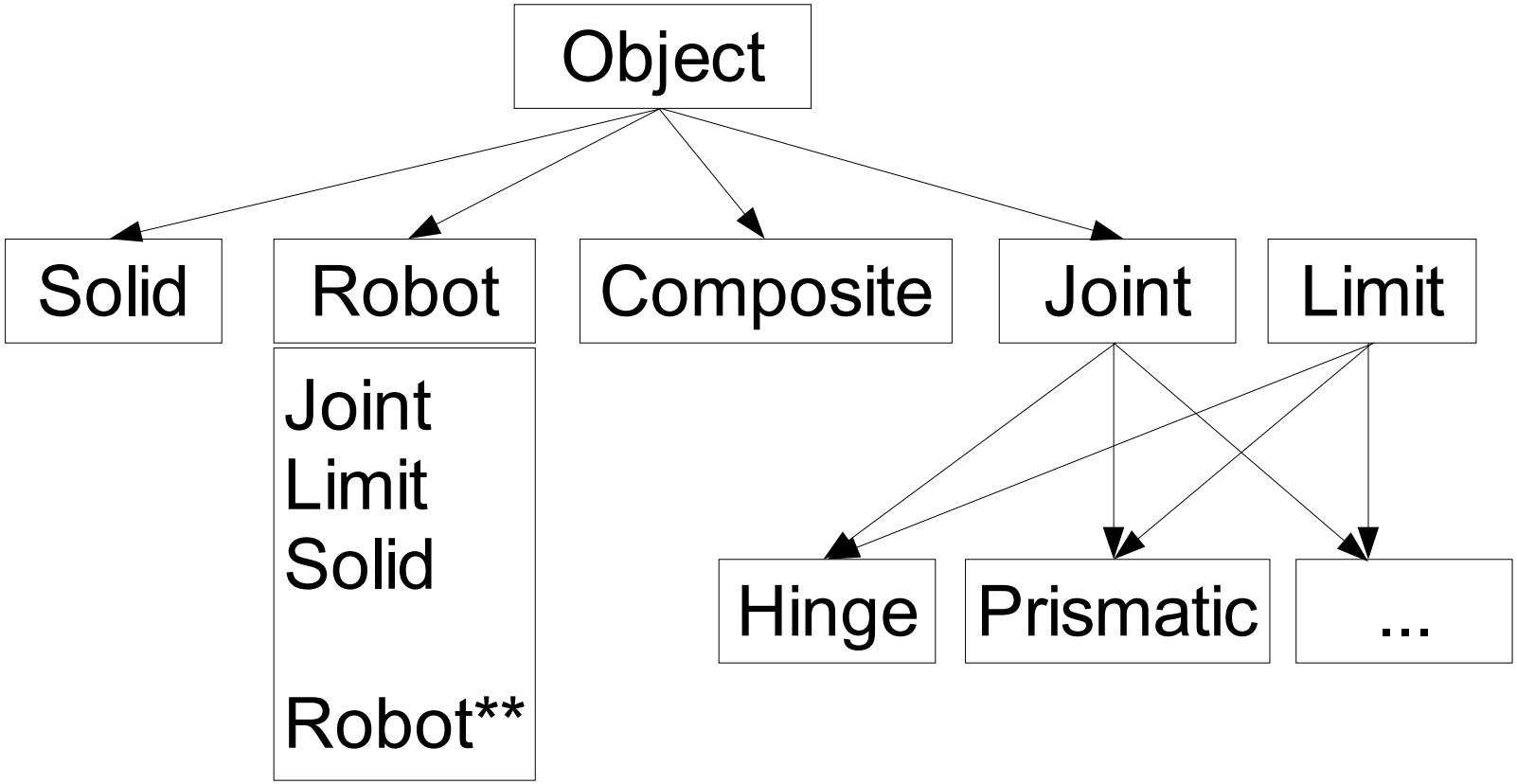
Contacts

- beschränken Bewegung an mindestens einem Punkt
- "Kontakt" heißt: *Abstand innerhalb einer Toleranz*
  - Körper können separiert oder durchdrungen sein!
- Kontakt-Eigenschaften (Auswirkung auf Dynamik) abhängig von Material-Tabelle

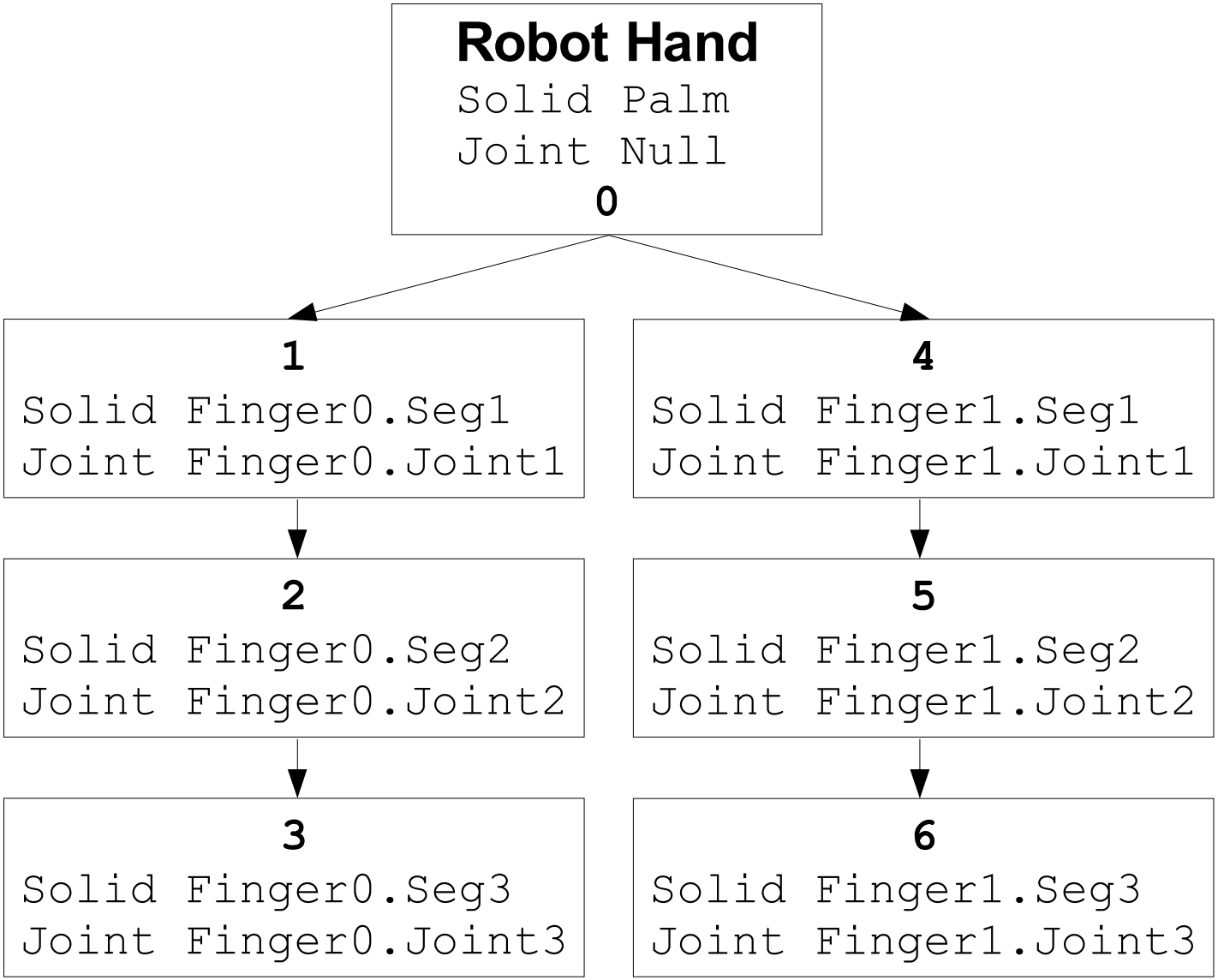
C++ Wrapper

- *Universe*-Klasse:
  - verwaltet Szenen
    - *List<Scene>*
  - globale Parameter: Materialien, Gravitation etc
- *Scene*-Klasse:
  - verwaltet Vortex-Körper, *Joints*, Grafik-Objekte
    - *Hashtable<Object>*
    - *Hashtable<Joint>*
    - *Hashtable<Graphics>*

C++ Wrapper


Object
Solid
Robot
Composite
Joint
Limit
Joint
Limit
Solid
Robot**
Hinge
Prismatic
...

C++ Wrapper


Robot Hand
Solid Palm
Joint Null
0
1
Solid Finger0.Seg1
Joint Finger0.Joint1
4
Solid Finger1.Seg1
Joint Finger1.Joint1
2
Solid Finger0.Seg2
Joint Finger0.Joint2
5
Solid Finger1.Seg2
Joint Finger1.Joint2
3
Solid Finger0.Seg3
Joint Finger0.Joint3
6
Solid Finger1.Seg3
Joint Finger1.Joint3

Vortex in Neo

- benötigte Neo-Units:
  - **Render** (Unit-Container *vtk*)
    - Vtk-Render-Fenster mit Maus-/Keyboard-Interaktion
  - **universe** (Unit-Container *vortex*)
    - Vortex-Universum
  - **scene** (Unit-Container *vortex*)
    - Vortex-Szene
  - **vtkViewer** (Unit-Container *vortex*)
    - lädt Vortex-Universum in den Vtk-Renderer

Einfaches Beispiel

VortexNeo-Bsp1.NST.gz

- Units: Render, universe, scene, vtkViewer
- Solids: box, cylinder

Vtk-Render-Fenster

- *Linke Maustaste* gedrückt
  - Drehen um "Bild-x-y"-Achsen
- *Ctrl* + *linke Maustaste* gedrückt
  - Drehen um "Bild-z"-Achse
- *Rechte Maustaste* gedrückt
  - Zoom (Bewegung in "Bild-z"-Richtung)
- *Mittlere Maustaste* gedrückt
  - Bewegung in "Bild-x-y"-Richtung
- *Shift* + *linke Maustaste* auf Objekt
  - Kraft auf Objekt ausüben

Vtk-Render-Fenster

- *'h'*: Tastenbelegung in Konsole zeigen (*help*)
- *'g'*: Zusatz-Grafiken zeigen
- *'s'*: SimulationStep
- *'r'*: auf ganze Szene zoomen (*reset*)
- *'f'*: Kräfte für angewähltes Objekt zeigen
- *'c'*: Kontakte für angewähltes Objekt + Reibungskegel
- *'m'*: Massen-Schwerpunkt (+Trägheits-Ellipsoid)

Weitere Beispiele

1) VortexNeo-Bsp-Unit.NST.gz
2) VortexNeo-Bsp-Prog.NST.gz

- Einbeziehung der *robot*- und *scene*-Unit
  - über *robot_ref*- bzw *scene*-Unit (1.)
  - über *prog*-Unit (2.)
- *universe:step* und *Render:Update*
- Input-Window korrekt konfigurieren

Szenen-Beschreibung

- einfache Objekte direkt erzeugbar
  - Box, Sphere, Cylinder
- komplexere Szenen mittels XML-Dateien (*.me*)
  - Welt-Parameter
  - Geometrien
  - Solids
  - ...

Die XML-Datei (Grundgerüst)

```
<ME>
    <WORLD>
```

- globale Welt-(Universum-)Parameter
- *Tag* wird nur bei *top-level*-.me-Dateien ausgewertet
- mehrere *top-level*-.me-Dateien überschreiben alte Werte

```
    </WORLD>

    <IMPORT>
```

- hierarchischer Aufbau komplexer Szenen durch Importieren anderer *.me*-Dateien

```
    </IMPORT>

    <GEOMETRIES>
```

- Geometrie-Primitive zur Mehrfach-Verwendung durch *solids*
- bilden Basis für die *collision detection*

```
    </GEOMETRIES>
```

Die XML-Datei (Grundgerüst)

<SOLID>

- Definition des eigentlichen Vortex-Körpers
- besteht aus einem Geometrie-Primitiv oder aus mehreren <*PART*>s

</SOLID>

<SOLID>...

<HINGE> .. </HINGE> <!-- *Joint* -->

<PRISMATIC> .. </PRISMATIC> <!-- *Joint* -->

<SPRING> .. </SPRING> <!-- *Joint* -->

<LIMIT>

- Grenzwerte für Joints bzw. Joint-Motoren

</LIMIT>

<GRAPHICPRIMS>

- Grafik-Primitive zur Mehrfach-Verwendung durch <*GRAPHICS*>

</GRAPHICPRIMS>

Die XML-Datei (Grundgerüst)

```
    <GRAPHICS>
```

- Objekte ausschließlich mit Grafik, ohne Vortex-Körper
- gehen nicht in *collision detection* mit ein

```
    </GRAPHICS>
</ME>

<ROBOT>
```

- Aufbau einer *Robot*-Baumstruktur
- beginnend mit dem angegebenen *solid* werden alle folgenden durch *joints* verbundene *solids* zum *Robot* hinzugefügt

```
</ROBOT>

<NOCOLLISION>
```

- setzt die *collision detection* für <u>ein</u> *solid*-Paar außer Kraft
- nötig wegen Abstands-Toleranz bei Kontakten

```
</NOCOLLISION>
```

XML-Beispiel 1

```
<ME>
  <GEOMETRIES>
    <BOX>
        <ID>seg0box1</ID>
        <DIMS>1.0, 4.9, 1.7</DIMS>
    </BOX>

    <CYLINDER>
        <ID>seg0cyl1</ID>
        <RADIUS>0.5</RADIUS>
        <HEIGHT>0.3</HEIGHT>
    </CYLINDER>

    <MESH>
        <ID> fingerkuppe </ID>
        <FILE> fingerkuppe.obj </FILE>
        <SCALE> 2.5 </SCALE>
    </MESH>
  </GEOMETRIES>

  <SOLID>
    <ID>Seg0</ID>
    <PART>
        <GEOMETRY_ID>seg0box1</GEOMETRY_ID>
        <POS>0, -3.64, -0.655</POS>
        <EULER>-15°, 0, 0</EULER>
    </PART>
    <PART>
        <GEOMETRY_ID>fingerkuppe</GEOMETRY_ID>
        <POS> -0.25, -1.31, 0 </POS>
        <EULER>0, 1.57, 0</EULER>
    </PART>
  </SOLID>
```

-1-

```
  <SOLID>
    <ID>Seg1</ID>
    <GEOMETRY_ID>seg0box2</GEOMETRY_ID>
    <POS>0, -0.75, -0.05</POS>
    <EULER>0, 0, 0</EULER>
  </SOLID>

  <NOCOLLISION>
    <SOLIDS>Seg0, Seg1</SOLIDS>
  </NOCOLLISION>

  <HINGE>
    <ID>Joint1</ID>
    <SOLID_IDS>Seg0, Seg1</SOLID_IDS>
    <RELTO>Seg0</RELTO>
    <AXIS>0, 0, 1</AXIS>
    <LIMIT>
        <STOPS>-25°, 25°</STOPS>
        <MAXFORCE>100</MAXFORCE>
        <MAXSPEED>0.5</MAXSPEED>
    </LIMIT>
  </HINGE>

  <GRAPHICPRIMS>
    <FRAME>
        <ID>frame1</ID>
    </FRAME>
  </GRAPHICPRIMS>

  <GRAPHICS>
    <PRIMID>frame1</PRIMID>
    <RELTO>Seg0</RELTO>
  </GRAPHICS>
</ME>
```

-2-

XML-Beispiel 2

```
<ME>
  <WORLD>
    <GRAVITY>0, 0, 0</GRAVITY>
  </WORLD>

  <IMPORT>
    <ID>Palm</ID>
    <NAME>TUM_palm.me</NAME>
  </IMPORT>

  <IMPORT>
    <ID>Finger0</ID>
    <NAME>TUM_finger.me</NAME>
    <POS>0.73, -5.55, -0.8</POS>
    <EULER>105°, 45°, 0</EULER>
  </IMPORT>

  <IMPORT>
    <ID>Finger1</ID>
    <NAME>TUM_finger.me</NAME>
    <POS>-5.3, -2.2, 1.3</POS>
    <EULER>165°, 0, 90°</EULER>
  </IMPORT>

  <IMPORT>
    <ID>Finger2</ID>
    <NAME>TUM_finger.me</NAME>
    <POS>-5.3, 2.2, 1.3</POS>
    <EULER>165°, 0, 90°</EULER>
  </IMPORT>
```



```
  <NOCOLLISION>
    <SOLIDS>Palm.Palm, Finger0.Seg0</SOLIDS>
  </NOCOLLISION>

  <NOCOLLISION>
    <SOLIDS>Palm.Palm, Finger1.Seg0</SOLIDS>
  </NOCOLLISION>

  <NOCOLLISION>
    <SOLIDS>Palm.Palm, Finger2.Seg0</SOLIDS>
  </NOCOLLISION>

</ME>

<ROBOT>
    <ID>Hand</ID>
    <SOLID>Palm.Palm</SOLID>

    <ROBOT>
        <JOINT>Finger0.Joint1</JOINT>
        <SOLID>Finger0.Seg1</SOLID>
    </ROBOT>
    <ROBOT>
        <JOINT>Finger1.Joint1</JOINT>
        <SOLID>Finger1.Seg1</SOLID>
    </ROBOT>
    <ROBOT>
        <JOINT>Finger2.Joint1</JOINT>
        <SOLID>Finger2.Seg1</SOLID>
    </ROBOT>
</ROBOT>
```

Quelle / Dokumentation

http://www.techfak.uni-bielefeld.de/ags/ni/projects/simulation_and_visual/neo/contrib/vortex/vortex.html

enthält folgende Links:

- offizielle Vortex Homepage
- offizielle Vortex Dokumentation
- Vortex Tutorial
- *.me*-Datei-Beschreibung