Visualisierung räumlicher Mechanismen in Standard 3D-Umgebungen

Julian Stolzlechner

6. November 2014

Definition zentraler Begriffe

Definition

Ein räumlicher *Mechanismus* besteht aus einer Anzahl von Systemen $\Sigma_0, \Sigma_1, \ldots, \Sigma_{n-1}$ deren freie Beweglichkeit durch gewisse Verbindungen (*Gelenke*) eingeschränkt ist.

Ein Mechanismus der aus n Systemen $\Sigma_0, \Sigma_1, \ldots, \Sigma_{n-1}$ besteht, wird auch *n- gliedrige kinematische Kette* genannt. Die Räume Σ_i bezeichnet man dabei als *Glieder* des Mechanismus bzw. der kinematischen Kette.

Insbesondere spricht man von einer Zwanglaufkette, wenn jede Bewegung zweier Glieder gegeneinander nur von einem Parameter t abhängt.

Definition einer Zwanglaufketten

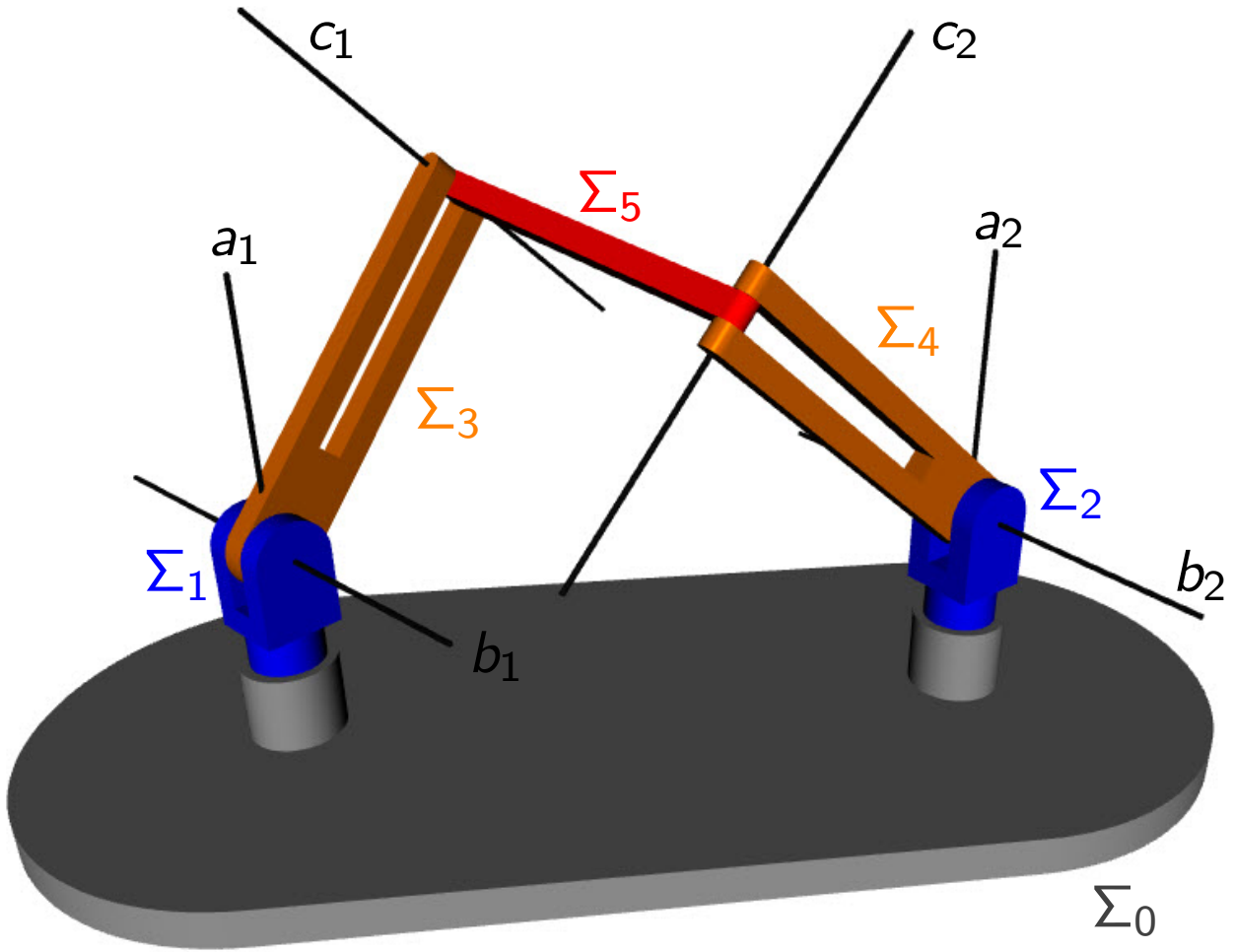
Eine Zwanglaufkette wir eindeutig festgelegt durch:

- die geometrischen Objekte welche die Systeme Σ_i eines Mechanismus repräsentieren $(i = 1, \cdots, n)$
- die vom Parameter t abhängigen Relativbewegungen Σ_i/Σ_0.

$$\Sigma_i/\Sigma_0 : \qquad \mathbf{x_i(t)} = \mathbf{A(t)} \cdot \mathbf{x} + \mathbf{a(t)} \qquad (i = 1, \cdots, n)$$

Ein Beispiel

Der SCHATZ Mechanismus:



VRML/X3D

- VRML: **V**irtual **R**eality **M**odeling **L**anguage
- X3D: e**X**tensible **3D**
- Beschreibungssprachen virtueller 3D-Welten im Internet
- VRML:
  im September 1997 $\rightarrow$ Industriestandard ISO/IEC 14772
- Nachfolgeformat X3D:
  seit Dezember 2004 $\rightarrow$ Industriestandard ISO/IEC 19775
- Interpretation der beschriebenen Szene durch Internetbrowser + Plugin bzw. einen Stand-Alone-Browser
- Erstellung einer VRML/X3D Sezene mit eine gewöhnlichen Editor möglich (Dateiendungen .wrl/.x3d).

VRML/X3D

- VRML: **V**irtual **R**eality **M**odeling **L**anguage
- X3D: e**X**tensible **3D**
- Beschreibungssprachen virtueller 3D-Welten im Internet
- VRML:
  im September 1997 $\rightarrow$ Industriestandard ISO/IEC 14772
- Nachfolgeformat X3D:
  seit Dezember 2004 $\rightarrow$ Industriestandard ISO/IEC 19775
- Interpretation der beschriebenen Szene durch Internetbrowser + Plugin bzw. einen Stand-Alone-Browser
- Erstellung einer VRML/X3D Sezene mit eine gewöhnlichen Editor möglich (Dateiendungen .wrl/.x3d).

VRML/X3D

- VRML: **V**irtual **R**eality **M**odeling **L**anguage
- X3D: e**X**tensible **3D**
- Beschreibungssprachen virtueller 3D-Welten im Internet
- VRML:
  im September 1997 $\rightarrow$ Industriestandard ISO/IEC 14772
- Nachfolgeformat X3D:
  seit Dezember 2004 $\rightarrow$ Industriestandard ISO/IEC 19775
- Interpretation der beschriebenen Szene durch Internetbrowser + Plugin bzw. einen Stand-Alone-Browser
- Erstellung einer VRML/X3D Sezene mit eine gewöhnlichen Editor möglich (Dateiendungen .wrl/.x3d).

VRML/X3D

- VRML: **V**irtual **R**eality **M**odeling **L**anguage
- X3D: e**X**tensible **3D**
- Beschreibungssprachen virtueller 3D-Welten im Internet
- VRML:
  im September 1997 → Industriestandard ISO/IEC 14772
- Nachfolgeformat X3D:
  seit Dezember 2004 → Industriestandard ISO/IEC 19775
- Interpretation der beschriebenen Szene durch Internetbrowser + Plugin bzw. einen Stand-Alone-Browser
- Erstellung einer VRML/X3D Sezene mit eine gewöhnlichen Editor möglich (Dateiendungen .wrl/.x3d).

VRML/X3D

- VRML: **V**irtual **R**eality **M**odeling **L**anguage
- X3D: e**X**tensible **3D**
- Beschreibungssprachen virtueller 3D-Welten im Internet
- VRML:
  im September 1997 $\rightarrow$ Industriestandard ISO/IEC 14772
- Nachfolgeformat X3D:
  seit Dezember 2004 $\rightarrow$ Industriestandard ISO/IEC 19775
- Interpretation der beschriebenen Szene durch Internetbrowser + Plugin bzw. einen Stand-Alone-Browser
- Erstellung einer VRML/X3D Sezene mit eine gewöhnlichen Editor möglich (Dateiendungen .wrl/.x3d).

VRML/X3D

- VRML: **V**irtual **R**eality **M**odeling **L**anguage
- X3D: e**X**tensible **3D**
- Beschreibungssprachen virtueller 3D-Welten im Internet
- VRML:
  im September 1997 $\rightarrow$ Industriestandard ISO/IEC 14772
- Nachfolgeformat X3D:
  seit Dezember 2004 $\rightarrow$ Industriestandard ISO/IEC 19775
- Interpretation der beschriebenen Szene durch Internetbrowser + Plugin bzw. einen Stand-Alone-Browser
- Erstellung einer VRML/X3D Sezene mit eine gewöhnlichen Editor möglich (Dateiendungen .wrl/.x3d).

VRML/X3D

- VRML: **V**irtual **R**eality **M**odeling **L**anguage
- X3D: e**X**tensible **3D**
- Beschreibungssprachen virtueller 3D-Welten im Internet
- VRML:
  im September 1997 $\rightarrow$ Industriestandard ISO/IEC 14772
- Nachfolgeformat X3D:
  seit Dezember 2004 $\rightarrow$ Industriestandard ISO/IEC 19775
- Interpretation der beschriebenen Szene durch Internetbrowser + Plugin bzw. einen Stand-Alone-Browser
- Erstellung einer VRML/X3D Sezene mit eine gewöhnlichen Editor möglich (Dateiendungen .wrl/.x3d).

VRML/X3D

- VRML: **V**irtual **R**eality **M**odeling **L**anguage
- X3D: e**X**tensible **3D**
- Beschreibungssprachen virtueller 3D-Welten im Internet
- VRML:
  im September 1997 $\rightarrow$ Industriestandard ISO/IEC 14772
- Nachfolgeformat X3D:
  seit Dezember 2004 $\rightarrow$ Industriestandard ISO/IEC 19775
- Interpretation der beschriebenen Szene durch Internetbrowser + Plugin bzw. einen Stand-Alone-Browser
- Erstellung einer VRML/X3D Sezene mit eine gewöhnlichen Editor möglich (Dateiendungen .wrl/.x3d).

VRML/X3D

Vollständiges VRML-Skript zur Darstellung einer roten Kugel:

```
#VRML V2.0 utf8

Shape {
   appearance Appearance {
      material Material { diffuseColor 1.0 0.0 0.0 }
   }
   geometry Sphere { radius 2.0 }
}
```

VRML/X3D

Vollständiges X3D-Skript zur Darstellung einer roten Kugel:

```
<?xml version='1.0' encoding='UTF-8'?>

<X3D profile='Immersive'>

   <Scene>

      <Shape>
         <Appearance>
            <Material diffuseColor='1 0 0'/>
         </Appearance>
         <Sphere radius='2.0'/>
      </Shape>

   </Scene>

</X3D>
```

Bewegungsanimationen in VRML/X3D

Eine Animation in VRML/X3D besteht im allgemeinen aus folgenden Komponenten:

- **Taktgeber der Animation**
  z.B. ein `TimeSensor`-Knoten
- **ausführenden Knoten**
  z.B. ein `Transform`-Knoten
- **Intepolatoren**
  z.B. `PositionInterpolator`, `OrientationInterpolator`
- **Route-Anweisungen**

Bewegungsanimationen in VRML/X3D

Eine Animation in VRML/X3D besteht im allgemeinen aus folgenden Komponenten:

- **Taktgeber der Animation**
  z.B. ein `TimeSensor`-Knoten
- **ausführenden Knoten**
  z.B. ein `Transform`-Knoten
- **Intepolatoren**
  z.B. `PositionInterpolator`, `OrientationInterpolator`
- **Route-Anweisungen**

Bewegungsanimationen in VRML/X3D

Eine Animation in VRML/X3D besteht im allgemeinen aus folgenden Komponenten:

- **Taktgeber der Animation**
  z.B. ein `TimeSensor`-Knoten
- **ausführenden Knoten**
  z.B. ein `Transform`-Knoten
- **Intepolatoren**
  z.B. `PositionInterpolator`, `OrientationInterpolator`
- **Route-Anweisungen**

Bewegungsanimationen in VRML/X3D

Eine Animation in VRML/X3D besteht im allgemeinen aus folgenden Komponenten:

- **Taktgeber der Animation**
  z.B. ein `TimeSensor`-Knoten
- **ausführenden Knoten**
  z.B. ein `Transform`-Knoten
- **Intepolatoren**
  z.B. `PositionInterpolator`, `OrientationInterpolator`
- **Route-Anweisungen**

Bewegungsanimationen in VRML/X3D

Eine Animation in VRML/X3D besteht im allgemeinen aus folgenden Komponenten:

- **Taktgeber der Animation**
  z.B. ein `TimeSensor`-Knoten
- **ausführenden Knoten**
  z.B. ein `Transform`-Knoten
- **Intepolatoren**
  z.B. `PositionInterpolator`, `OrientationInterpolator`
- **Route-Anweisungen**

Beispiel: animierte Schraubung

ausführender Knoten:

```
DEF Schraubung Transform {
   children [
      Transform {
         translation 10.0 0.0 0.0
         children [
            Shape {
               appearance Appearance {
                  material Material {
                     diffuseColor 1.0 0.0 0.0
                  }
               }
               geometry Sphere { radius 2.0 }
            }
         ]
      }
   ]
}
```

Beispiel: animierte Schraubung

ausführender Knoten:

```
DEF Schraubung Transform {
   children [
      Transform {
         translation 10.0 0.0 0.0
         children [
            Shape {
               appearance Appearance {
                  material Material {
                     diffuseColor 1.0 0.0 0.0
                  }
               }
               geometry Sphere { radius 2.0 }
            }
         ]
      }
   ]
}
```

Beispiel: animierte Schraubung

ausführender Knoten:

```
DEF Schraubung Transform {
   children [
      Transform {
         translation 10.0 0.0 0.0
         children [
            Shape {
               appearance Appearance {
                  material Material {
                     diffuseColor 1.0 0.0 0.0
                  }
               }
               geometry Sphere { radius 2.0 }
            }
         ]
      }
   ]
}
```

Beispiel: animierte Schraubung

Taktgeber:

```
DEF Uhr TimeSensor {
   cycleInterval 5.0
   loop TRUE
}
```

Beispiel: animierte Schraubung

Taktgeber:

```
DEF Uhr TimeSensor {
   cycleInterval 5.0
   loop TRUE
}
```

Beispiel: animierte Schraubung

Interpolatoren:

```
DEF Schiebanteil PositionInterpolator {
   key [ 0.0, 0.25, 0.5, 0.75, 1.0 ]
   keyValue [ 0.0 0.0 0.0,
              0.0 0.0 5.0,
              0.0 0.0 10.0,
              0.0 0.0 5.0,
              0.0 0.0 0.0 ]
}

DEF Drehanteil OrientationInterpolator {
   key [ 0.0, 0.25, 0.5, 0.75, 1.0 ]
   keyValue [ 0.0 0.0 1.0 0.0,
              0.0 0.0 1.0 3.14,
              0.0 0.0 1.0 6.28,
              0.0 0.0 1.0 3.14,
              0.0 0.0 1.0 0.0 ]
}
```

Beispiel: animierte Schraubung

Interpolatoren:

```
DEF Schiebanteil PositionInterpolator {
   key [ 0.0, 0.25, 0.5, 0.75, 1.0 ]
   keyValue [ 0.0 0.0 0.0,
              0.0 0.0 5.0,
              0.0 0.0 10.0,
              0.0 0.0 5.0,
              0.0 0.0 0.0 ]
}

DEF Drehanteil OrientationInterpolator {
   key [ 0.0, 0.25, 0.5, 0.75, 1.0 ]
   keyValue [ 0.0 0.0 1.0 0.0,
              0.0 0.0 1.0 3.14,
              0.0 0.0 1.0 6.28,
              0.0 0.0 1.0 3.14,
              0.0 0.0 1.0 0.0 ]
}
```

Beispiel: animierte Schraubung

Interpolatoren:

```
DEF Schiebanteil PositionInterpolator {
   key [ 0.0, 0.25, 0.5, 0.75, 1.0 ]
   keyValue [ 0.0 0.0 0.0,
              0.0 0.0 5.0,
              0.0 0.0 10.0,
              0.0 0.0 5.0,
              0.0 0.0 0.0 ]
}

DEF Drehanteil OrientationInterpolator {
   key [ 0.0, 0.25, 0.5, 0.75, 1.0 ]
   keyValue [ 0.0 0.0 1.0 0.0,
              0.0 0.0 1.0 3.14,
              0.0 0.0 1.0 6.28,
              0.0 0.0 1.0 3.14,
              0.0 0.0 1.0 0.0 ]
}
```

Beispiel: animierte Schraubung

Routen:

```
ROUTE Uhr.fraction_changed TO Schiebanteil.set_fraction
ROUTE Uhr.fraction_changed TO Drehanteil.set_fraction

ROUTE Schiebanteil.value_changed TO Schraubung.set_translation
ROUTE Drehanteil.value_changed TO Schraubung.set_rotation
```

Beispiel: animierte Schraubung

Routen:

```
ROUTE Uhr.fraction_changed TO Schiebanteil.set_fraction
ROUTE Uhr.fraction_changed TO Drehanteil.set_fraction

ROUTE Schiebanteil.value_changed TO Schraubung.set_translation
ROUTE Drehanteil.value_changed TO Schraubung.set_rotation
```

BEISPIEL

Interaktion in VRML/X3D

Einflussnahme des Benutzers auf die virtuelle Welt durch:

- Bewegung des unsichtbaren Beobachters durch diverse Button im Browser
- Mausoperationen in Zusammenspiel mit Sensoren

Interaktion in VRML/X3D

Einflussnahme des Benutzers auf die virtuelle Welt durch:

- Bewegung des unsichtbaren Beobachters durch diverse Button im Browser
- Mausoperationen in Zusammenspiel mit Sensoren

Interaktion in VRML/X3D

Einflussnahme des Benutzers auf die virtuelle Welt durch:

- Bewegung des unsichtbaren Beobachters durch diverse Button im Browser
- Mausoperationen in Zusammenspiel mit Sensoren

Interaktion in VRML/X3D

Einflussnahme des Benutzers auf die virtuelle Welt durch:

- Bewegung des unsichtbaren Beobachters durch diverse Button im Browser
- Mausoperationen in Zusammenspiel mit Sensoren

Interaktion in VRML/X3D

Berührungssensoren: Der TouchSensor-Knoten

```
TouchSensor {
    enabled TRUE
    touchTime
}
```

Interaktion in VRML/X3D

Bewegungssensoren: Der PlaneSensor-Knoten:

```
PlaneSensor {
    enabled TRUE
    minPosition 0 0
    maxPosition 1 1
    translation_changed
}
```

Interaktion in VRML/X3D

Annäherungssensoren: Der ProximitySensor-Knoten:

```
ProximitySensor {
    enabled TRUE
    center 0 0 0
    size 1 1 1
    position_changed
    orientation_changed
}
```

BEISPIELE

Ausgangspunkt/Ziel der Arbeit

Ausgangspunkt:
C++ Bibliothek zur Erstellung statischer 2D bzw. 3D Szenen:

- Headerdateien: `geometry.h`, `graphics.h`
- Ausgabeformat: 2D $\rightarrow$ PostScript, 3D $\rightarrow$ VRML, POV-Ray

Ziel:
Erweiterung der Bibliothek um die Datei `motion.h` zur Erzeugung dynamischer 3D Szenen im VRML- bzw. X3D-Format.

Ausgangspunkt/Ziel der Arbeit

Ausgangspunkt:
C++ Bibliothek zur Erstellung statischer 2D bzw. 3D Szenen:

- Headerdateien: `geometry.h`, `graphics.h`
- Ausgabeformat: 2D $\rightarrow$ PostScript, 3D $\rightarrow$ VRML, POV-Ray

Ziel:
Erweiterung der Bibliothek um die Datei `motion.h` zur Erzeugung dynamischer 3D Szenen im VRML- bzw. X3D-Format.

Grundidee

Programmierung einer eigenen Klasse namens 'mechanism' mit dessen Hilfe Zwanglaufketten samt der Geometrie ihrer Systeme erstellt und im VRML- bzw. X3D-Format abgespeichert werden können.

Funktionsweise:

über Methoden der Klasse

$\longrightarrow$

relevante Daten gesammelt

$\longrightarrow$

in lokale Variablen der Klasse gespeichert

$\longrightarrow$

VRML- bzw. X3D-Export

Grundidee

Programmierung einer eigenen Klasse namens 'mechanism' mit dessen Hilfe Zwanglaufketten samt der Geometrie ihrer Systeme erstellt und im VRML- bzw. X3D-Format abgespeichert werden können.

Funktionsweise:

über Methoden der Klasse

$\longrightarrow$

relevante Daten gesammelt

$\longrightarrow$

in lokale Variablen der Klasse gespeichert

$\longrightarrow$

VRML- bzw. X3D-Export

Vorgangsweise bei der Erstellung eines Mechanismsus:

1. Erstellung einer Instanz der Klasse `mechanism`

```
meinMechanismus mechanism(anz, t0, t1, nop, a_vec, e_vec, bcol, dur)
```

`anz` ... Anzahle der Systeme
`t0, t1` ... Parameterintervall
`nop` ... Anzahl der KeyFrame/Schlüsselwerte
`a_vec` ... Augpunkt
`e_vec` ... Hauptsehstrahl
`bcol` ... Hintergrundfarbe
`dur` ... dauer der Animation

Vorgangsweise bei der Erstellung eines Mechanismsus:

1. Erstellung einer Instanz der Klasse `mechanism`

```
meinMechanismus mechanism(anz, t0, t1, nop, a_vec, e_vec, bcol, dur)
```

`anz` ... Anzahle der Systeme
`t0, t1` ... Parameterintervall
`nop` ... Anzahl der KeyFrame/Schlüsselwerte
`a_vec` ... Augpunkt
`e_vec` ... Hauptsehstrahl
`bcol` ... Hintergrundfarbe
`dur` ... dauer der Animation

Vorgangsweise bei der Erstellung eines Mechanismsus:

2. Einfügen von Ojekten

```
meinMechanismus.ins(Objekt, i)
```

Als Speicher dienen Listen von Objektlisten `L`. Das eingefügte Objekt wird in der Liste `L[i]` eingefügt.

Vorgangsweise bei der Erstellung eines Mechanismsus:

2. Einfügen von Ojekten

```
meinMechanismus.ins(Objekt, i)
```

Als Speicher dienen Listen von Objektlisten `L`. Das eingefügte Objekt wird in der Liste `L[i]` eingefügt.

Vorgangsweise bei der Erstellung eines Mechanismsus:

3. Einfügen der Relativbewegungen

```
meinMechanismus.ins(motion, i)
```

Schlüsselwerte für die keyFrame-Animation werden berechnet und in lokale Variablen `MPI` und `MOI` abgespeichert.

`MPI[i]` $\cdots$ Liste mit Schiebvektoren der Relativbewegung Σ_i/Σ_0
`MOI[i]` $\cdots$ Liste viedimensionalen Vektoren die zur Drehmatrix der Relativbewegung Σ_i/Σ_0 gehören

Vorgangsweise bei der Erstellung eines Mechanismsus:

3. Einfügen der Relativbewegungen

```
meinMechanismus.ins(motion, i)
```

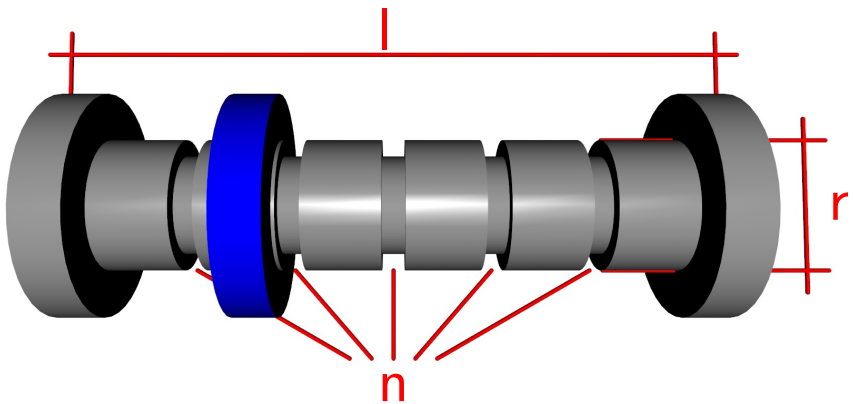
Schlüsselwerte für die keyFrame-Animation werden berechnet und in lokale Variablen `MPI` und `MOI` abgespeichert.

`MPI[i]` $\cdots$ Liste mit Schiebvektoren der Relativbewegung Σ_i/Σ_0
`MOI[i]` $\cdots$ Liste viedimensionalen Vektoren die zur Drehmatrix der Relativbewegung Σ_i/Σ_0 gehören

Vorgangsweise bei der Erstellung eines Mechanismsus:

4. Einfügen von Bahnkurven und Schiebereglern

```
meinMechanismus.insSlider(vec, alpha, l, r, col1, col2, n)
```




```
meinMechanismus.insOrbitCurve(vec, i)
```

Vorgangsweise bei der Erstellung eines Mechanismsus:

4. Export als VRML bzw. X3D Datei

```
meinMechanismus.exportVRML("..Dateipfad/Name.wrl")

meinMechanismus.exportX3D("..Dateipfad/Name.x3d")
```

Auf Basis der in den lokalen Variablen gespeicherten Daten wird eine Datein in VRML- bzw. X3D- Syntax exportiert:

```
ofstream f;
f.open(filname);
f << "#VRML V2.0 utf8" << endl;

...
```

Vorgangsweise bei der Erstellung eines Mechanismsus:

4. Export als VRML bzw. X3D Datei

```
meinMechanismus.exportVRML("..Dateipfad/Name.wrl")

meinMechanismus.exportX3D("..Dateipfad/Name.x3d")
```

Auf Basis der in den lokalen Variablen gespeicherten Daten wird eine Datein in VRML- bzw. X3D- Syntax exportiert:

```
ofstream f;
f.open(filname);
f << "#VRML V2.0 utf8" << endl;

...
```

Vorgangsweise bei der Erstellung eines Mechanismsus:

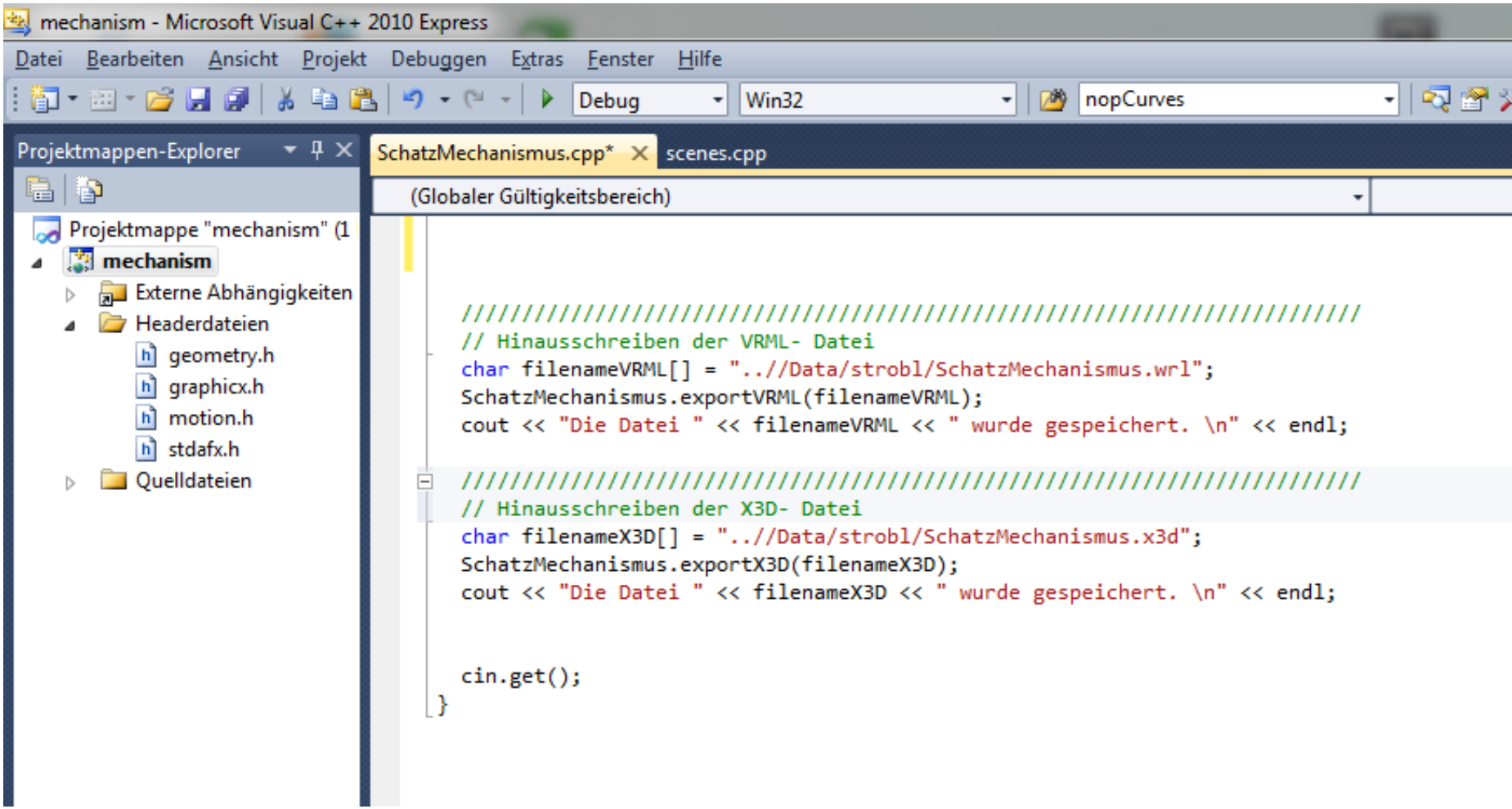
4. Export als VRML bzw. X3D Datei

```
meinMechanismus.exportVRML("..Dateipfad/Name.wrl")

meinMechanismus.exportX3D("..Dateipfad/Name.x3d")
```

Auf Basis der in den lokalen Variablen gespeicherten Daten wird eine Datein in VRML- bzw. X3D- Syntax exportiert:

```
ofstream f;
f.open(filname);
f << "#VRML V2.0 utf8" << endl;

...
```

Vorgangsweise bei der Erstellung eines Mechanismsus:



BEISPIELE