Sterowanie robotów

Wykład 1.

dr inż. Adam Wolniakowski
2018

Wprowadzenie

- prowadzący: dr inż. Adam Wolniakowski
- konsultacje: wtorek, środa, czwartek w godz. 10:15 – 12:00 w s. 608
- wykład: 7 x 2h + kolokwium
- projekt: 7 x 2h + obrona projektu
- ocena z wykładu: kolokwium
- ocena z projektu: średnia ocen z ćwiczeń

Materiały

- Materiały będą dostępne na e-platformie
- Kolejne ćwiczenia i prezentacje z wykładów będą udostępniane drogą mailową

Literatura

- John J. Craig: Wprowadzenie do robotyki.
- M. W. Spong, M. Vidyasagar: Dynamika i sterowanie robotów.
- K. Kozłowski, P. Dutkiewicz, W. Wróblewski: Modelowanie i sterowanie robotów.
- Howie Choset et al.: Principles of Robot Motion.

Dzisiaj

- Narzędzia stosowane w robotyce
- RobWork – wstęp, instalacja
- Przypomnienie sposobów opisu położeń układów współrzędnych
- Tworzenie komórki roboczej w programie RobWork

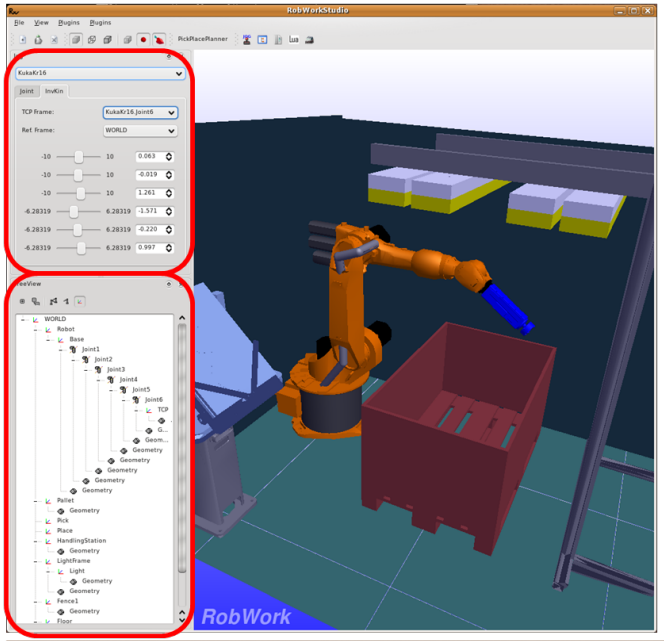
Zagadnienia

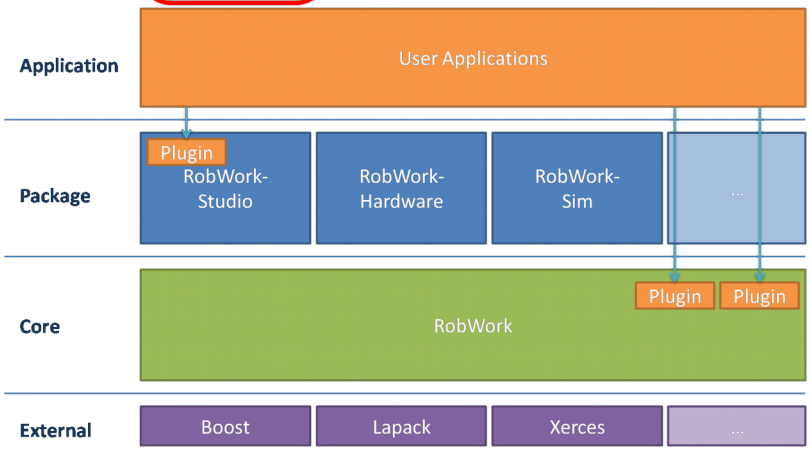
- modelowanie robotów w środowiskach symulacyjnych,
- **kinematyka prosta i odwrotna,**
- interpolacja,
- wykrywanie kolizji,
- planowanie ścieżek,
- generowanie trajektorii,
- **dynamika robotów,**
- układy sterowania.

Narzędzia

- **Matlab**, Simulink, Robotics toolbox
- Solid Works, **Blender**, **Meshlab**, 3DStudio, SketchUp, ...
- **GCC**, MinGW, **cmake**
- Python, **LUA**, Java, ...
- różne **edytory/IDE**
- Symulatory: VeroSim, **RobWorkSim**, Gazebo, ...
- Silniki fizyczne: **ODE**, Bullet, RWPE, ...
- ROS (Robot Operating System)!

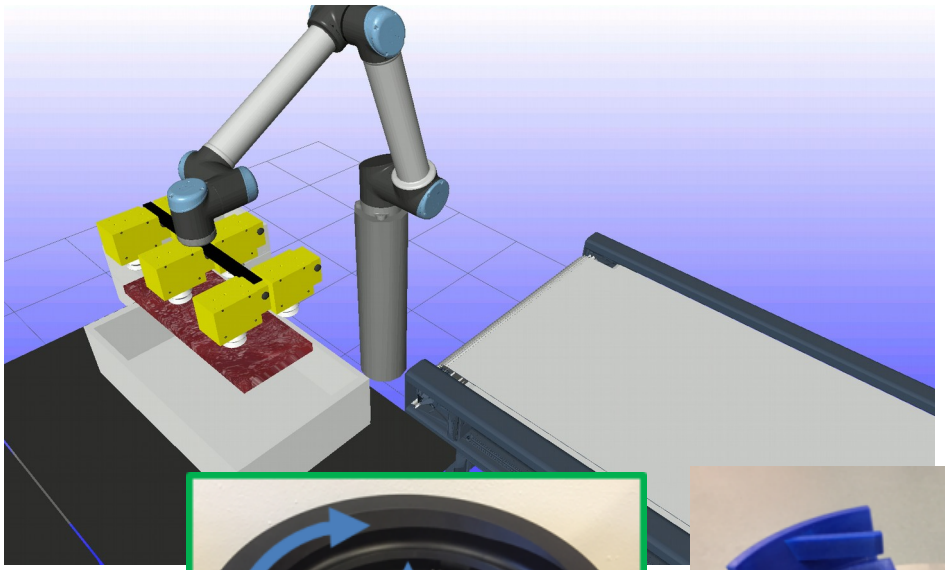
RobWork

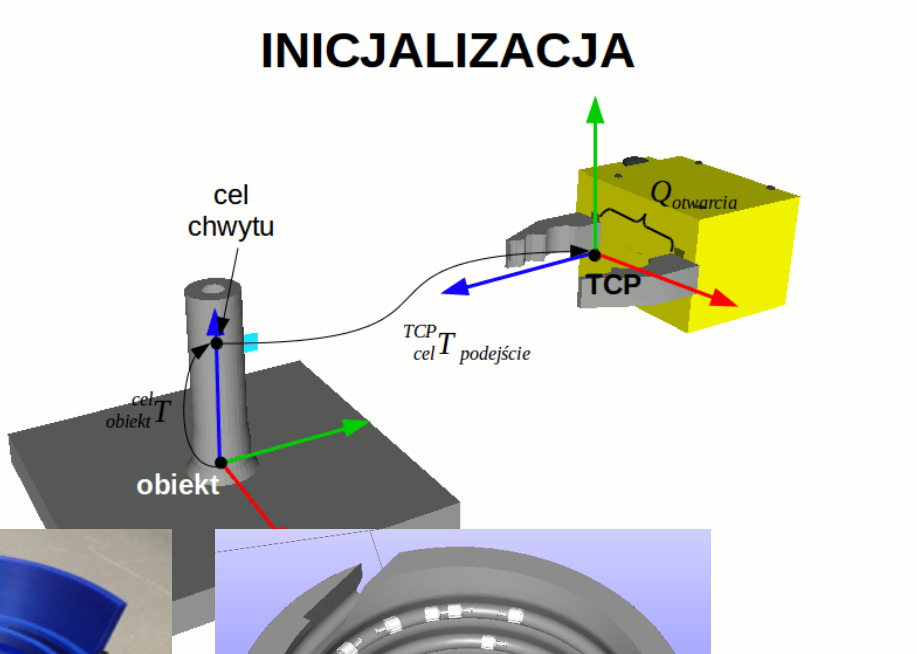


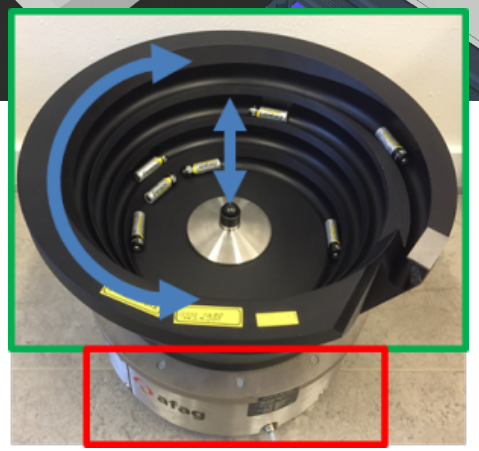


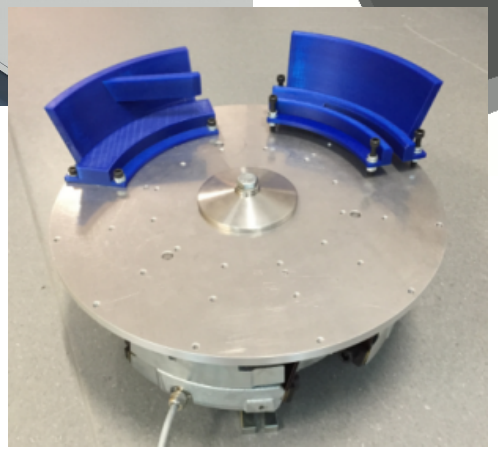
- www.robwork.dk
- Linux/Windows (?)
- modelowanie kinematyczne i dynamiczne manipulatorów szeregowych i równoległych
- wizualizacja
- wykrywanie kolizji
- kinematyka prosta i odwrotna
- planowanie i optymalizacja trajektorii
- modelowanie kontrolerów i sensorów
- RobWorkStudio GUI + pluginy
- interfejs dla języków skryptowych: Lua, Python, Java (Matlab)

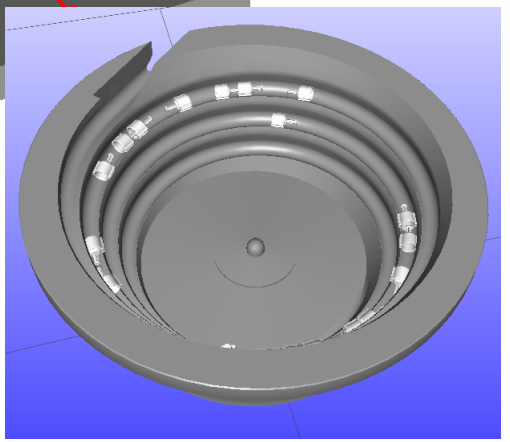
Przykłady zastosowań



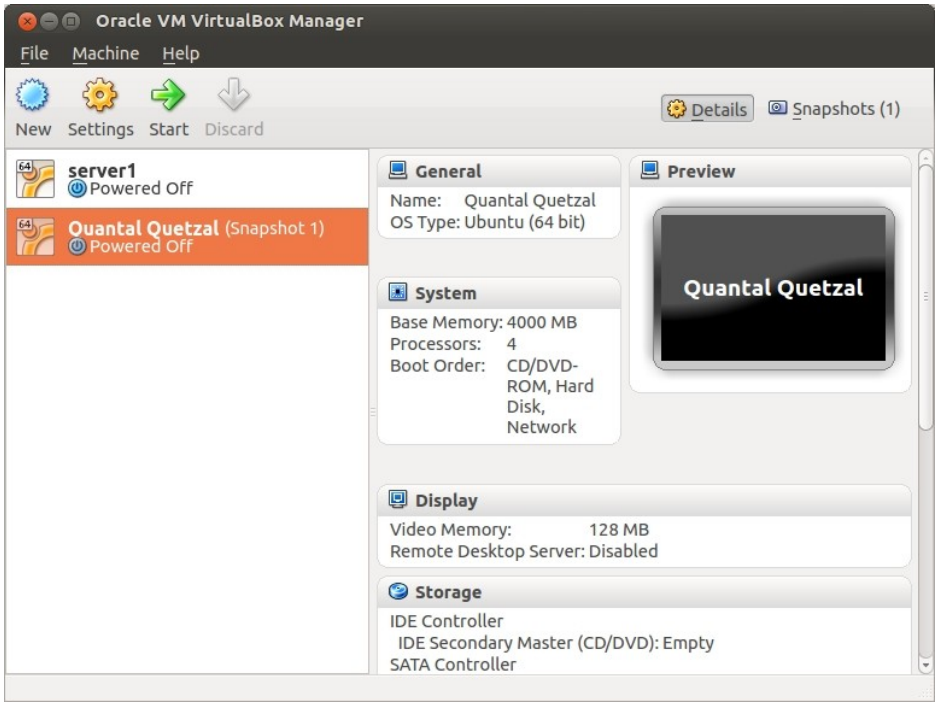








Maszyna wirtualna



- VirtualBox 5.2.6
- 4 GB pamięci RAM
- 4 procesory
- min. 15-20 GB dysku
- Linux 64-bitowy
- -> zainstalować Ubuntu 16.04

Instalacja RobWorka (1)

- Instalacja jest opisana na: http://www.robwork.dk/apidoc/nightly/rw/page_rw_installation_ubuntu.html
- Kolejne polecenia wykonujemy w terminalu:

```
sudo apt-get install subversion git mercurial
```

```
sudo apt-get install gcc g++ cmake
```

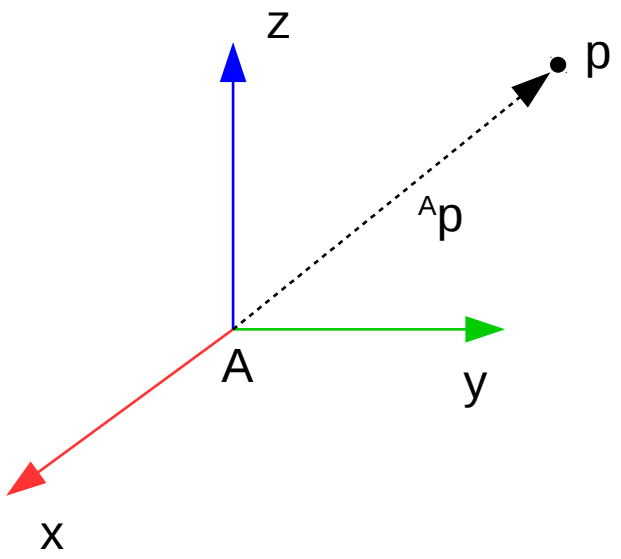
```
sudo apt-get install libboost-dev libboost-date-time-dev libboost-filesystem-dev libboost-program-options-dev libboost-regex-dev libboost-serialization-dev libboost-system-dev libboost-test-dev libboost-thread-dev
```

Instalacja RobWorka (2)

```
sudo apt-get install libxerces-c3.1 libxerces-c-dev
sudo apt-get install swig liblua5.2-dev python-dev default-jdk
sudo apt-get install libgtest-dev
sudo apt-get install qtbase5-dev
sudo apt-get install libode-dev libode4
```

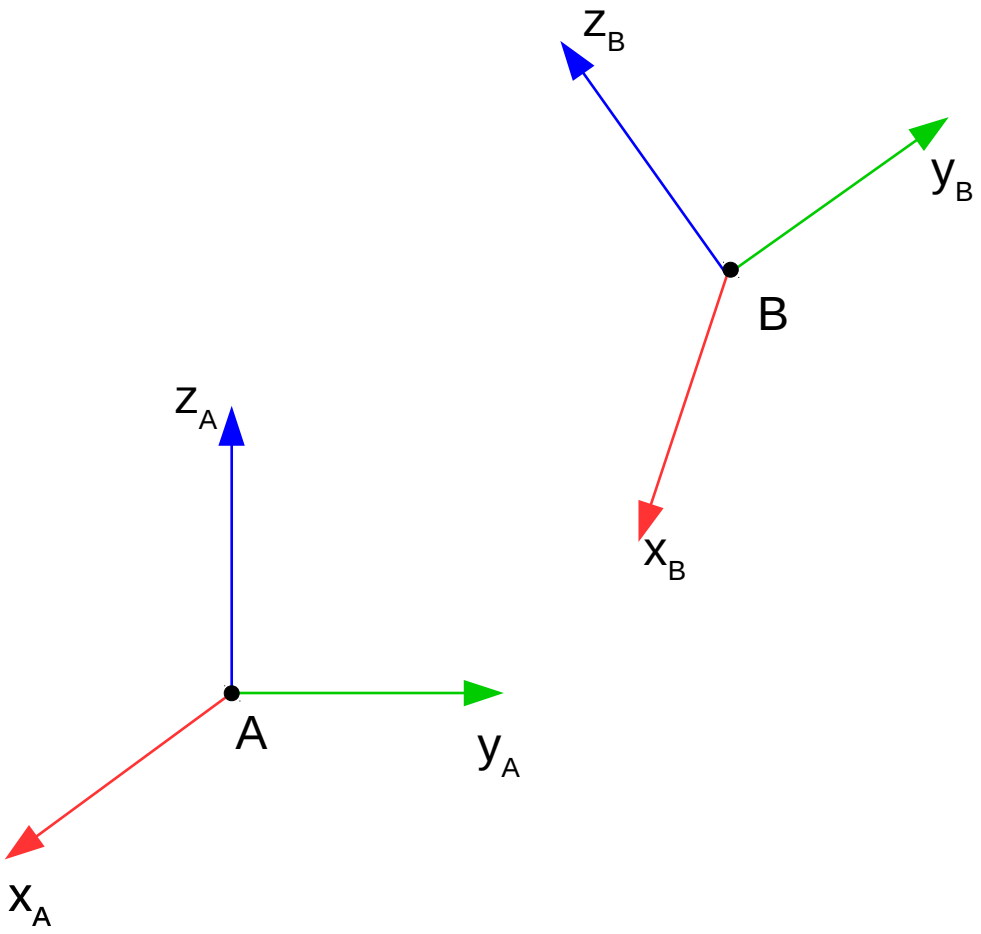
```
cd ~
mkdir robwork
cd robwork
svn co --username Guest --password '' https://svnsrv.sdu.dk/svn/RobWork/trunk/ .
cd RobWork/build
cmake -DCMAKE_BUILD_TYPE=Release ..
make -j4
cd ../../RobWorkStudio/build
cmake -DCMAKE_BUILD_TYPE=Release ..
make -j4
cd ../../RobWorkSim/build
cmake -DCMAKE_BUILD_TYPE=Release ..
make -j4
```

Opis położenia



- RGB = XYZ!
- ${}^{A}p$: położenie punktu p w układzie współrzędnych A
- ${}^{A}p = [\ {}^{A}p_x\ {}^{A}p_y\ {}^{A}p_z\ ]^T$

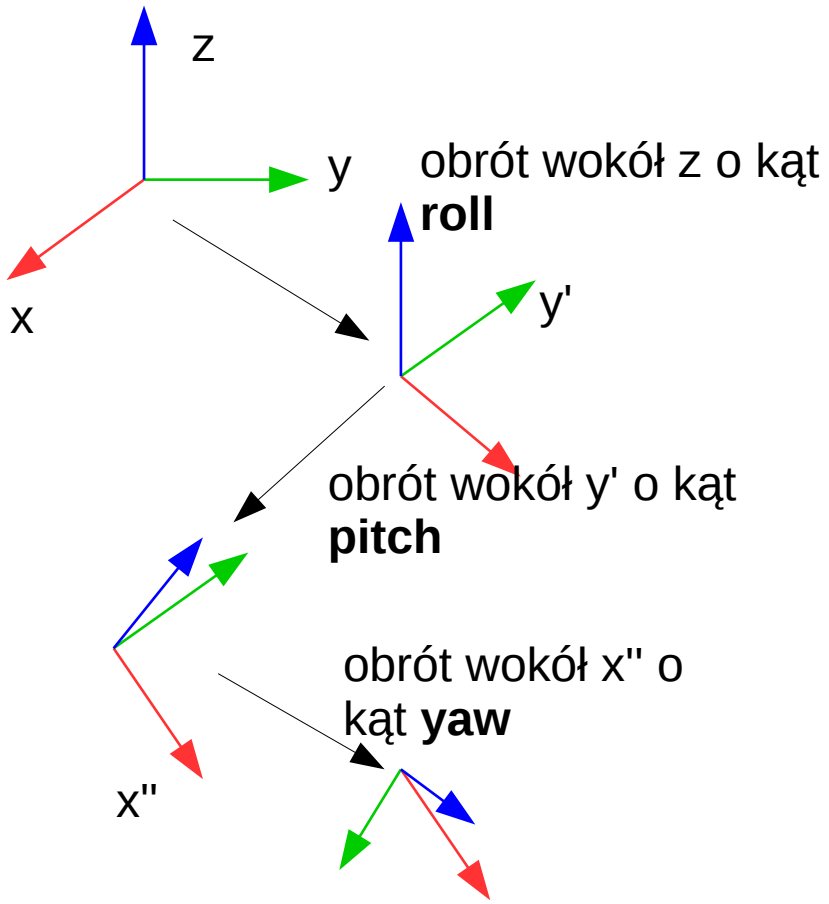
Opis orientacji



$$ {}^{A}_{B}R = \begin{bmatrix} r_{11} & r_{21} & r_{31} \\ r_{12} & r_{22} & r_{32} \\ r_{13} & r_{23} & r_{33} \end{bmatrix} $$

- Macierz ${}^{A}_{B}R$ opisuje orientację układu B względem układu A
- det (R) = 1
- $R^{-1}=R^{T}$

Kąty roll-pitch-yaw



- Notacja RPY służy do określenia orientacji przy pomocy 3 parametrów:
  - kąta *roll*
  - kąta *pitch*
  - kąta *yaw*
- Obracamy kolejno wokół osi **ruchomych**: Rz – Ry – Rx
- (ćwiczenie)

RPY <-> macierz rotacji

- Z RPY (alpha, beta, gamma) do macierzy rotacji:

$$\begin{bmatrix} \cos(\alpha)\cos(\beta) & \cos(\alpha)\sin(\beta)\sin(\gamma)-\sin(\alpha)\cos(\gamma) & \cos(\alpha)\sin(\beta)\cos(\gamma)+\sin(\alpha)\sin(\gamma) \\ \sin(\alpha)\cos(\beta) & \sin(\alpha)\sin(\beta)\sin(\gamma)+\cos(\alpha)\cos(\gamma) & \sin(\alpha)\sin(\beta)\cos(\gamma)-\cos(\alpha)\sin(\gamma) \\ -\sin(\beta) & \cos(\beta)\sin(\gamma) & \cos(\beta)\cos(\gamma) \end{bmatrix}$$

- Z macierzy rotacji do RPY:

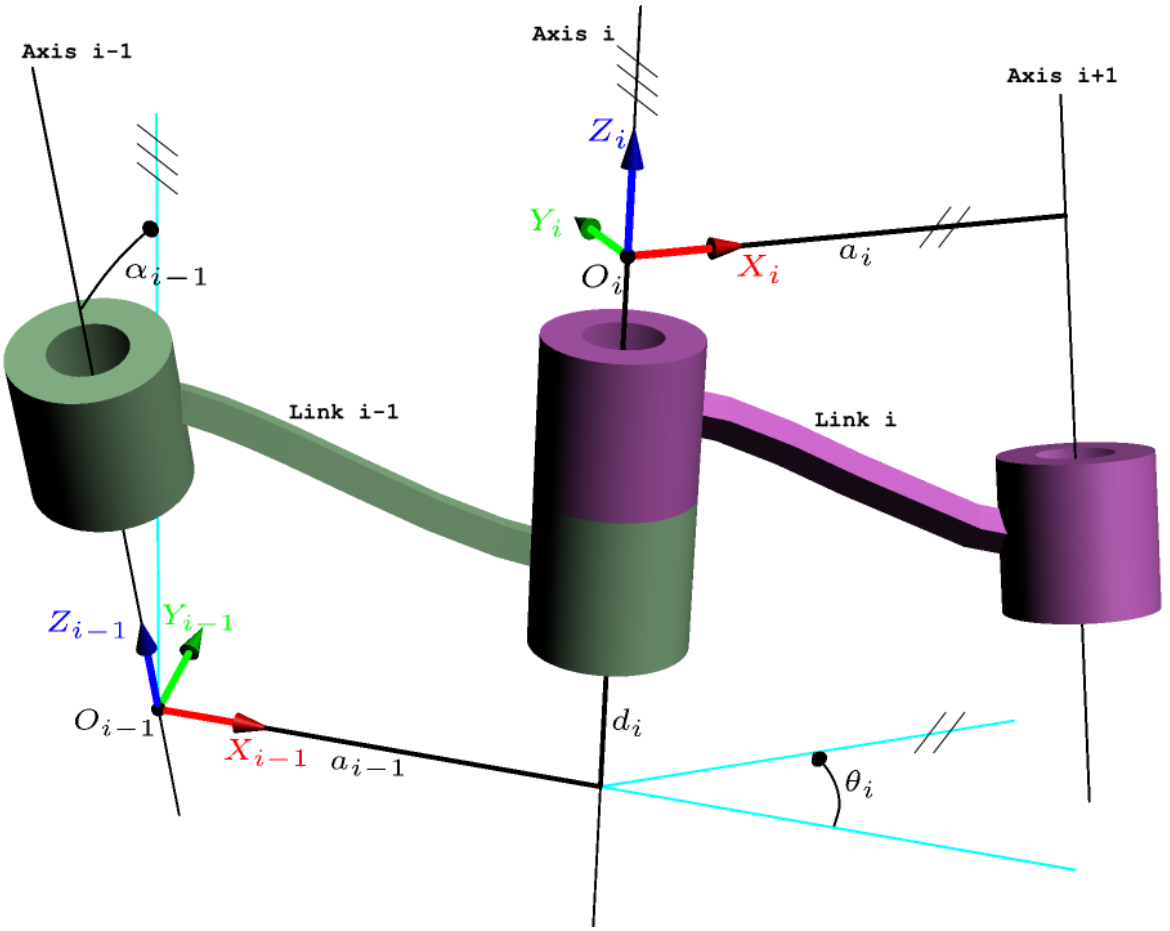
$$\begin{cases} \beta = \tan_2^{-1}(-r_{31}, \sqrt{r_{11}^2 + r_{21}^2}) \\ \alpha = \tan_2^{-1}(r_{21}, r_{11}) \\ \gamma = \tan_2^{-1}(r_{32}, r_{33}) \end{cases}$$

W RobWorku:

RPY<> rpy(0.1, 0.5, 0.8);
Rotation3D<> r = rpy.toRotation3D();

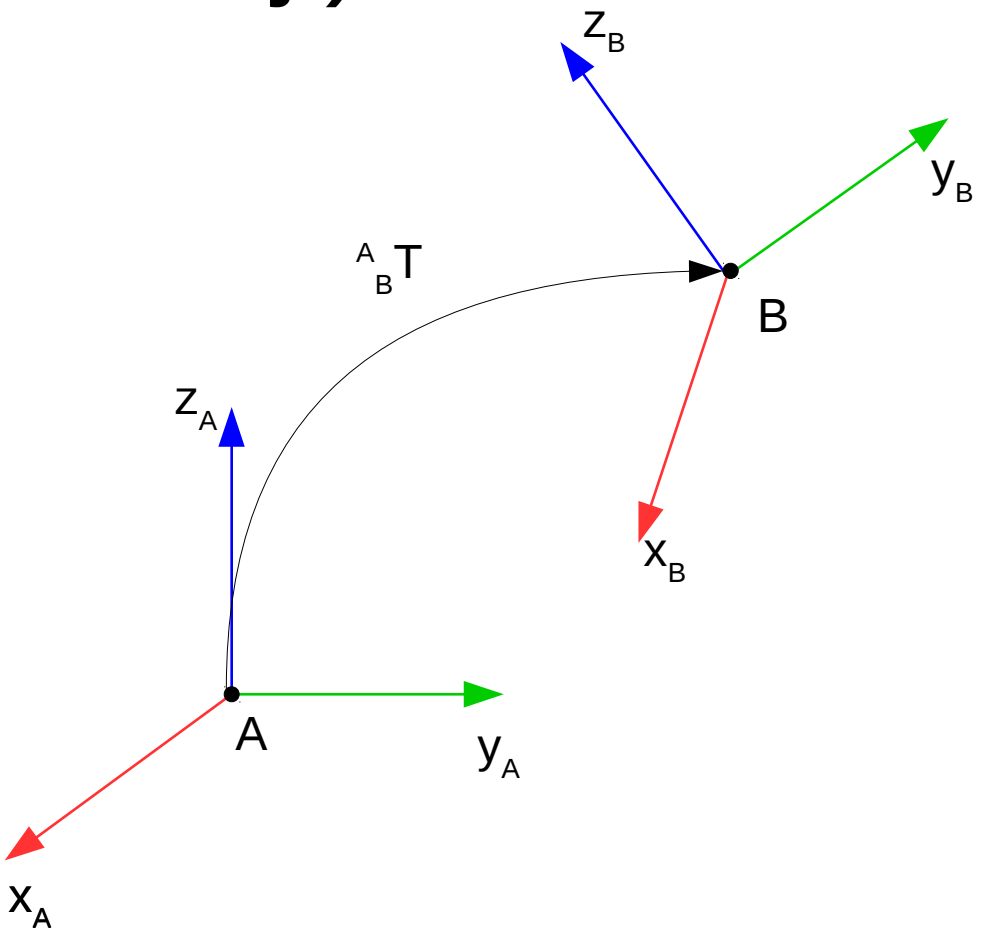
Rotation3D<> rot;
RPY<> rpy(rot);

Parametry D-H (Craig)



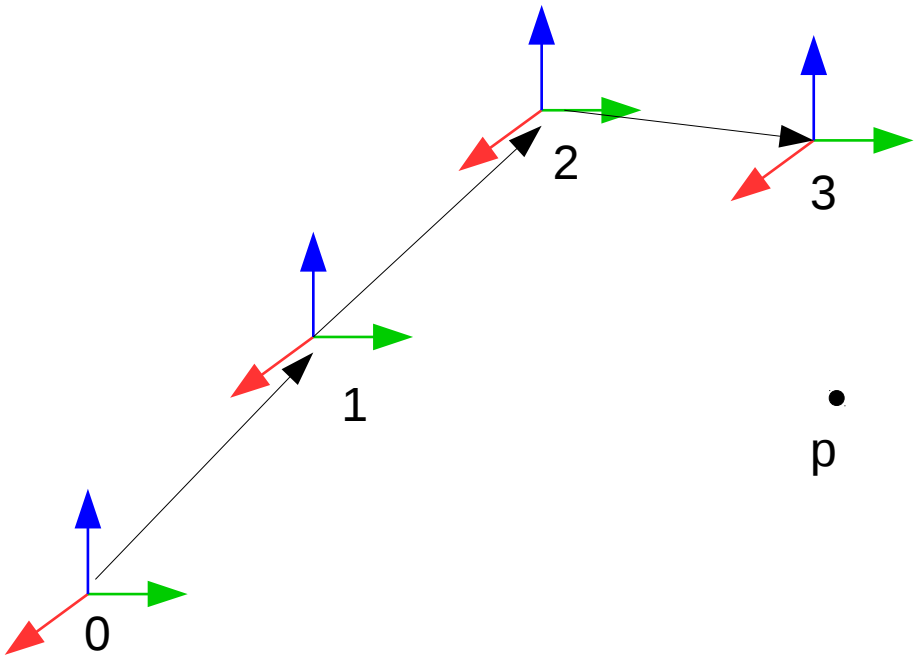
Macierz przekształcenia (transformacji)

$$ {}^{A}_{B}T = \begin{bmatrix} r_{11} & r_{21} & r_{31} & p_x \\ r_{12} & r_{22} & r_{32} & p_y \\ r_{13} & r_{23} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} $$



- Macierz ${}^{A}_{B}T$ 4x4 opisuje układ B względem układu A.
- ${}^{B}_{A}T = {}^{A}_{B}T^{-1}$

Składanie przekształceń



- ${}^{0}_{3}T = {}^{0}_{1}T\ {}^{1}_{2}T\ {}^{2}_{3}T$
- ${}^{3}_{0}T = ?$
- ${}^{0}p = {}^{0}_{3}T\ {}^{3}p$
- ${}^{3}p = ?\ {}^{0}p$

Język XML

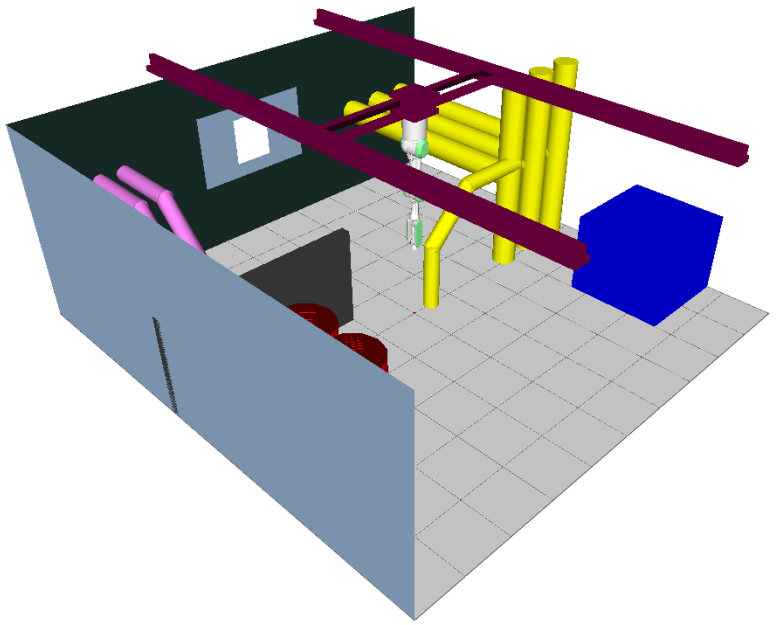
- XML (ang. *eXtensible Markup Language*) – uniwersalny język znaczników przeznaczony do reprezentowania różnych danych w strukturalizowany sposób.
- Tutorial: https://www.w3schools.com/xml/default.asp
- Struktura danych jest "drzewiasta"

```
<dokument nazwa="aaa">
  <studenci>
    <student id="1">
       <imie> Ania </imie>
       <nazwisko> Nowak </nazwisko>
    </student>
    <student id="2"> <!-- ... --> </student>
  </studenci>
  <blabla/>
  <ddd>
    <a> 12345 </a>
  <ddd>
  <!-- komentarz -->
</dokument>
```

Struktura projektu w RobWorku

- Katalog projektu:
  - *scene.wc.xml* (struktura kinematyczna komórki roboczej)
  - *Scene.dwc.xml* (opis właściwości dynamicznych komórki)
  - *setup.prox.xml* (określenie kolizji między obiektami)
  - Katalog *geometry*:
    - (modele geometryczne obiektów w scenie: .stl, .tri, .obj, .ac)
  - Katalog *robot*:
    - *robot.xml* (struktura kinematyczna urządzenia)
    - *robot.dwc.xml* (opis wł. dynamicznych urządzenia)
    - *setup.prox.xml* (określenie kolizji między elementami)
    - Katalog *geometry*:
      - (modele geometryczne członów robota: .stl, .tri, .obj, .ac)

Przykład: PA10gantry



```
pa10gantry/
├── Geometry
│   ├── Environment.ac
│   ├── Environment.stl
│   ├── floor.stl
│   ├── Gantry0.ac
│   ├── Gantry0.stl
│   ├── ...
├── MitsubishiPA10
│   ├── Geometry
│   │   ├── Rob-0.stl
│   │   ├── Rob-0.tri
│   │   ├── Rob-1.stl
│   │   ├── Rob-1.tri
│   │   ├── ...
│   │   └── welding-gun.stl
│   ├── pa10.wc.xml
│   └── setup.prox.xml
└── Scene.wc.xml
```

Wzór scene.wc.xml

```
<WorkCell
name="Single PA10
Demo">

<!-- definicje
obiektów i urz?dze?
-->

<CollisionSetup file
="setup.prox.xml"/>

</WorkCell>
```

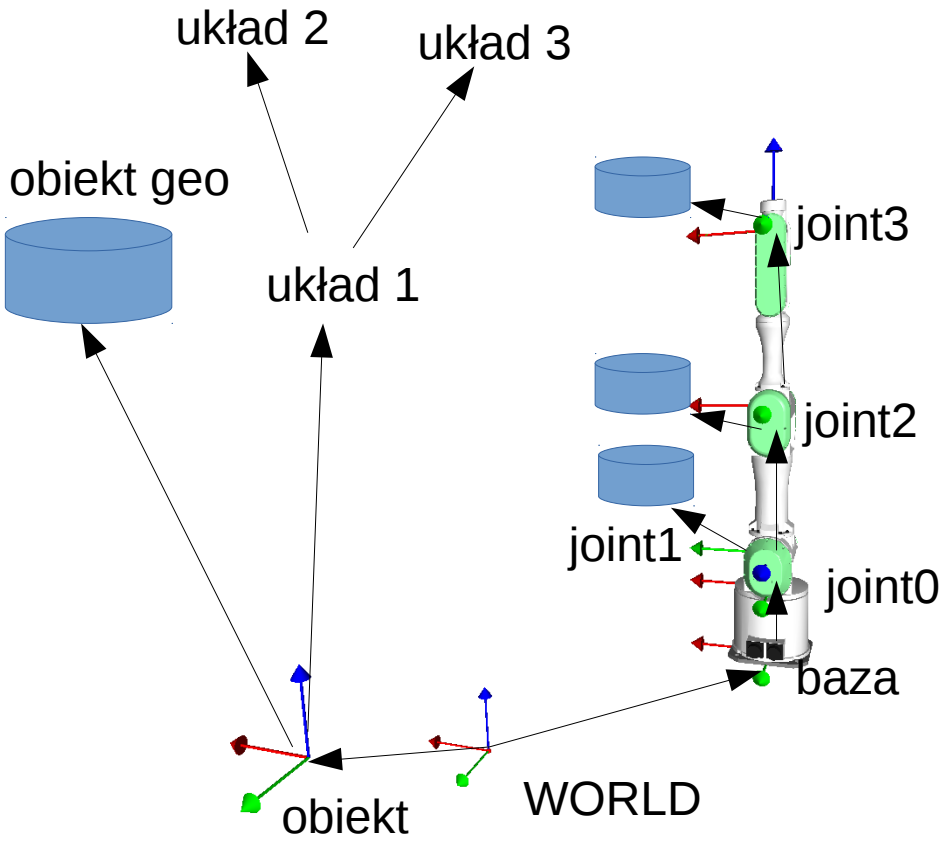
- *root tag* + nazwa komórki
- **tutaj będą definicje obiektów, układów odniesienia i urządzeń**
- dołącz plik konfiguracyjny kolizji

setup.prox.xml

```
<CollisionSetup>

  <Exclude>

    <FramePair
first="podloga"
second="pudelko" />

    <FramePair first="baza"
second="robot.Joint1" />

    <!-- inne pary -->

  </Exclude>

</CollisionSetup>
```

- plik opisuje pomiędzy którymi układami należy pomijać kolizje

Struktura kinematyczna komórki



- struktura kinematyczna komórki ma topografię drzewa
- każdy układ (frame!) ma "rodzica"
- układ posiada od 0 do wielu "potomków"
- w RW możemy jako potomka dodać tzw. "drawable" czyli wyświetlaną geometrię

Definicja układu odniesienia

```
<WorkCell name="a1">

  <Frame name="uklad1"
reframe="WORLD" type="Fixed">

    <RPY>0 0 0</RPY>

    <Pos>1 1 1</Pos>

  </Frame>

</WorkCell>
```

- komórka robocza
- name: nazwa układu
- reframe: względem którego jest zdefiniowana jego pozycja
- type: rodzaj (Fixed=nieruchoma, Movable=ruchoma)
- RPY: orientacja w postaci kątów roll-pitch-yaw (w stopniach)
- Pos: pozycja jako wektor x-y-z (w metrach)

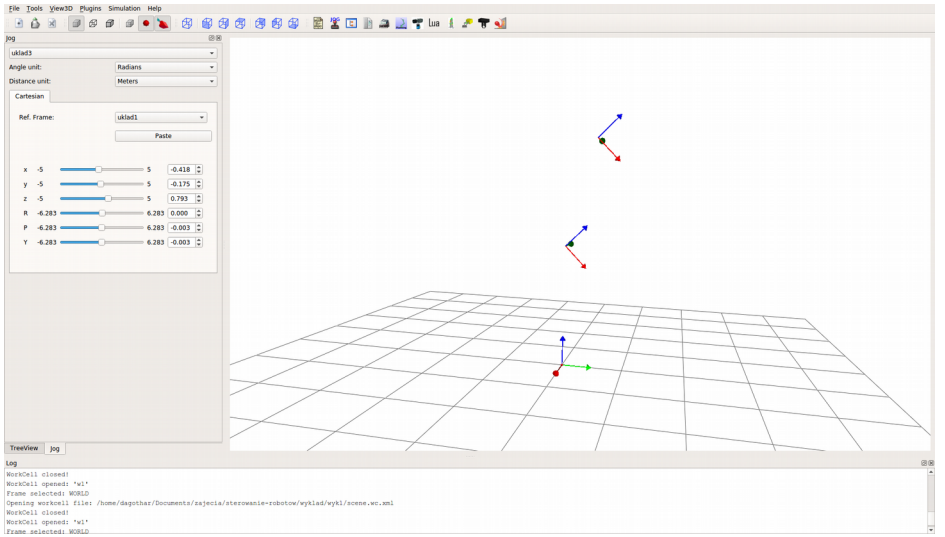
Użycie macierzy transformacji

```
<WorkCell name="a1">

  <Frame name="uklad2"
reframe="WORLD" type="Fixed">

    <Transform>

      1 0 0 2

      0 1 0 0.1

      0 0 1 -1

    </Transform>

  </Frame>

</WorkCell>
```

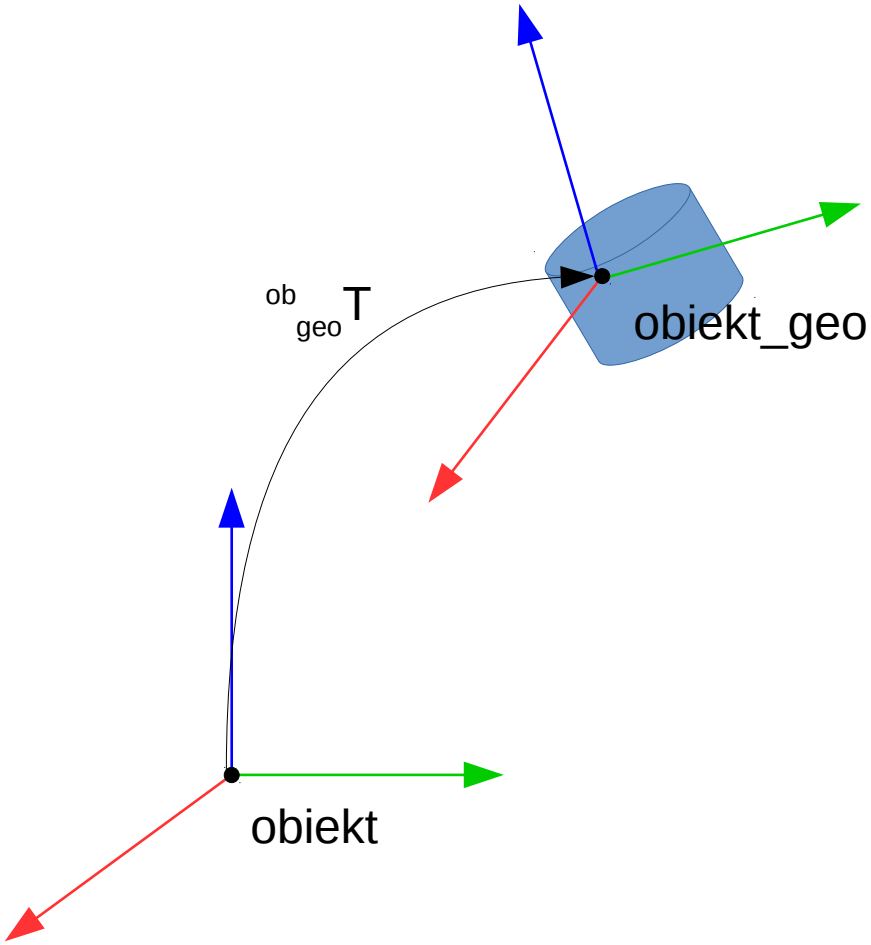
- macierz rotacji 3x3
- wektor przesunięcia x-y-z
- pomijamy wiersz 0 0 0 1!

Ruchome układy odniesienia

```
<WorkCell name="a1">

  <Frame name="uklad1"
reframe="WORLD" type="Fixed">

    <RPY>90 45 0</RPY>

    <Pos>0 0 1</Pos>

  </Frame>

  <Frame name="uklad2"
reframe="uklad1" type="Movable">

    <RPY>0 0 0</RPY>

    <Pos>0 0 1</Pos>

  </Frame>

</WorkCell>
```



Modele geometryczne

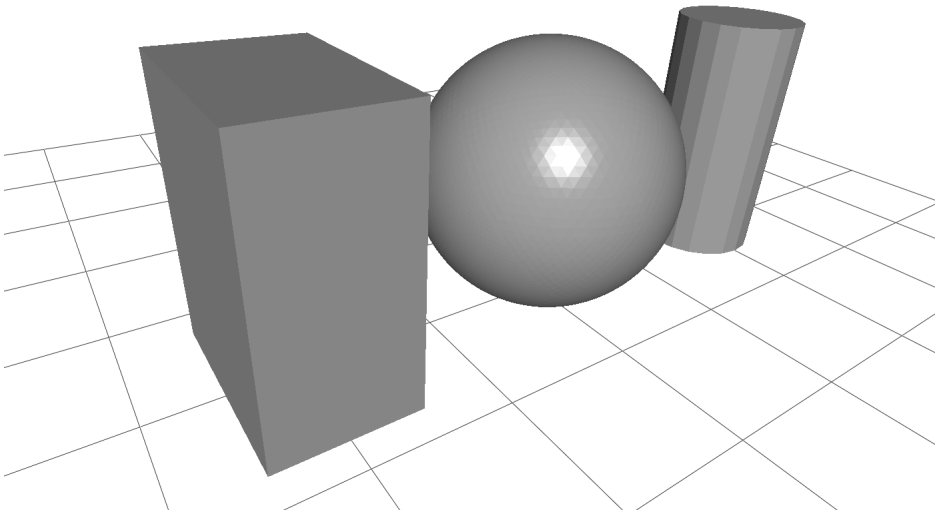


- Model ma własny układ współrzędnych.
- Należy podać jak wyświetlany model jest umieszczony względem układu współrzędnych obiektu:

```
<Frame name="b2" refframe="WORLD">
  <RPY>0 0 0</RPY> <Pos>0 0 0</Pos>
  <Drawable name="b2geo">
    <RPY>0 0 0</RPY>
    <Pos>0 0 0.5</Pos>
    <Sphere radius="0.5"/>
    </Drawable>
</Frame>
```

Geometryczne "prymitywy"

```
<Frame name="a1" refframe="WORLD">
    <RPY>0 0 0</RPY> <Pos>1 0 0</Pos>
    <Drawable name="a1geo">
      <RPY>0 0 0</RPY> <Pos>0 0 0.5</Pos>
      <Box x="0.5" y="0.75" z="1"/>
    </Drawable>
  </Frame>

  <Frame name="b2" refframe="WORLD">
    <RPY>0 0 0</RPY> <Pos>0 0 0</Pos>
    <Drawable name="b2geo">
      <RPY>0 0 0</RPY> <Pos>0 0 0.5</Pos>
      <Sphere radius="0.5"/>
    </Drawable>
  </Frame>

  <Frame name="d1" refframe="WORLD">
    <RPY>0 0 0</RPY> <Pos>-1 0 0</Pos>
    <Drawable name="d1geo">
      <RPY>0 0 0</RPY> <Pos>0 0 0.5</Pos>
      <Cylinder radius="0.25" z="1"/>
    </Drawable>
  </Frame>
```



Modele z plików

- RobWork obsługuje m.in. formaty:
  - STL
  - obj (kolor!)
  - tri
  - ac / ac3d (kolor!)
  - ...
- Pliki tworzymy w: SolidWorks, 3D studio, Blender, ... (Meshlab)

```
<Drawable name="BaseGeo" refframe="Base">
  <RPY>-90 0 0</RPY>
  <Pos>0 0 0</Pos>
  <Polytope file = "Geometry/Rob-0"/>
</Drawable>
```

- ac > stl

Definicja urządzenia szeregowego (1)

- Definiujemy urządzenia w oddzielnych plikach (np. *./pa10/pa10.wc.xml*) i dołączamy je do pliku sceny:

```
<Frame name="PA10"
refframe="WORLD">
    <RPY> 0 0 0 </RPY>
    <Pos> -1 0 0.5 </Pos>
 </Frame>
 <Include
file="pa10/pa10.wc.xml" />
```

- W katalogu *pa10* umieszczamy:
  - definicję struktury kinematycznej robota (*pa10.wc.xml*)
  - określenie ustawień kolizji (*setup.proxy.xml*)
  - katalog z modelami członów (np. *geometry*)

Definicja urządzenia szeregowego (1)

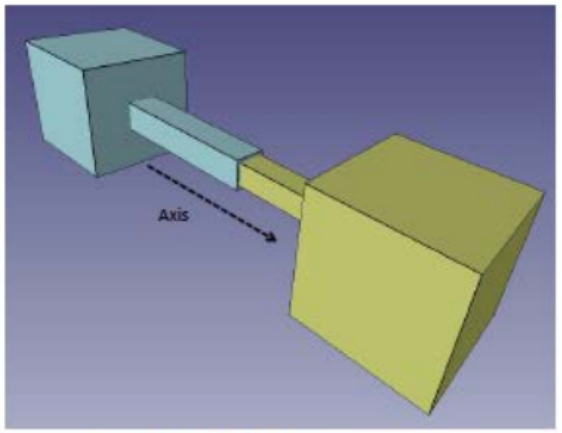
```
<SerialDevice name="PA10">
  <Frame name="Base" />

  <Joint name="Joint1" type="Revolute">
    <RPY>0 0 0</RPY>
    <Pos>0 0 1</Pos>
    <PosLimit min="-180" max="180"/>
    <VelLimit max="100"/>
    <AccLimit max="100"/>
  </Joint>
  <!-- ... -->

  <Drawable name="BaseGeo" refframe="Base">
    <RPY>-90.0 0.0 0.0</RPY><Pos>0.0 0.0 0.0</Pos>
    <Polytope file = "Geometry/Rob-0"/>
  </Drawable>

  <CollisionSetup file="setup.prox.xml" />
  <!-- ... -->

</SerialDevice>
```

- nazwa urządzenia
- układ bazy
- definicje członów (typ=Revolute/Prismatic)
- limity położenia, prędkości i przyspieszenia
- modele geometryczne (wyświetlane)
- ustawienia kolizji

Rodzaje przegubów

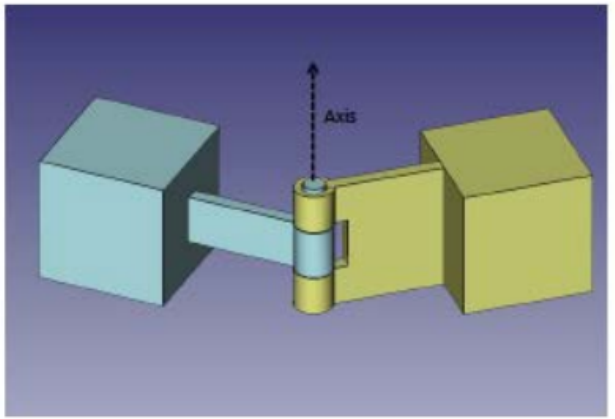
- Przegub przesuwny (prismatic):

```
<Joint name="j1" type="Prismatic">
  <RPY>0 0 0</RPY>
  <Pos>0 0 0.1</Pos>
  <PosLimit min="-1" max="1"/>
</Joint>
```



- Przegub obrotowy (revolute):

```
<Joint name="j1" type="Revolute">
  <RPY>0 0 0</RPY>
  <Pos>0 0 0.1</Pos>
  <PosLimit min="-180" max="180"/>
</Joint>
```



RPY/Pos, D-H (Craig)

- Notacja RPY/Pos:

```
<Joint name="j1"
type="Revolute">
  <RPY>0 0 0</RPY>
  <Pos>0 0 0.1</Pos>
  <PosLimit min="-
180" max="180"/>
</Joint>
```

- Notacja D-H (człon obrotowy)

```
<DHJoint name="j2" alpha="0" a="0.1"
d="0.1" offset="0">
  <PosLimit min="-180" max="180"/>
</DHJoint>
```

- Notacja D-H (człon przesuwny):

```
<DHJoint name="j3" alpha="0" a="0.1"
theta="0.1" offset="0">
  <PosLimit min="-1" max="1"/>
</DHJoint>
```

Ustawienia kolizji urządzenia

```
<CollisionSetup>
    <Exclude>
        <FramePair first="Base" second="Joint1"/>
        <FramePair first="Base" second="Joint2"/>
        <FramePair first="Base" second="Joint3"/>
        <FramePair first="Joint1" second="Joint2"/>
        <FramePair first="Joint1" second="Joint3"/>
        <FramePair first="Joint1" second="Joint4"/>
        <FramePair first="Joint1" second="Joint5"/>
        <FramePair first="Joint1" second="Joint6"/>
        <FramePair first="Joint2" second="Joint3"/>
        <FramePair first="Joint2" second="Joint4"/>
        <FramePair first="Joint2" second="Joint5"/>
        <FramePair first="Joint2" second="Joint6"/>
        <FramePair first="Joint3" second="Joint4"/>
        <FramePair first="Joint3" second="Joint5"/>
        <!-- ... -->
    </Exclude>
</CollisionSetup>
```

- wyłączamy wykrywanie kolizji pomiędzy sąsiednimi parami członów

Ćwiczenie: SCARA

- Zapisać plik XML z definicją struktury kinematycznej robota typu SCARA.
- Pominąć geometrię członów.
- +0.25