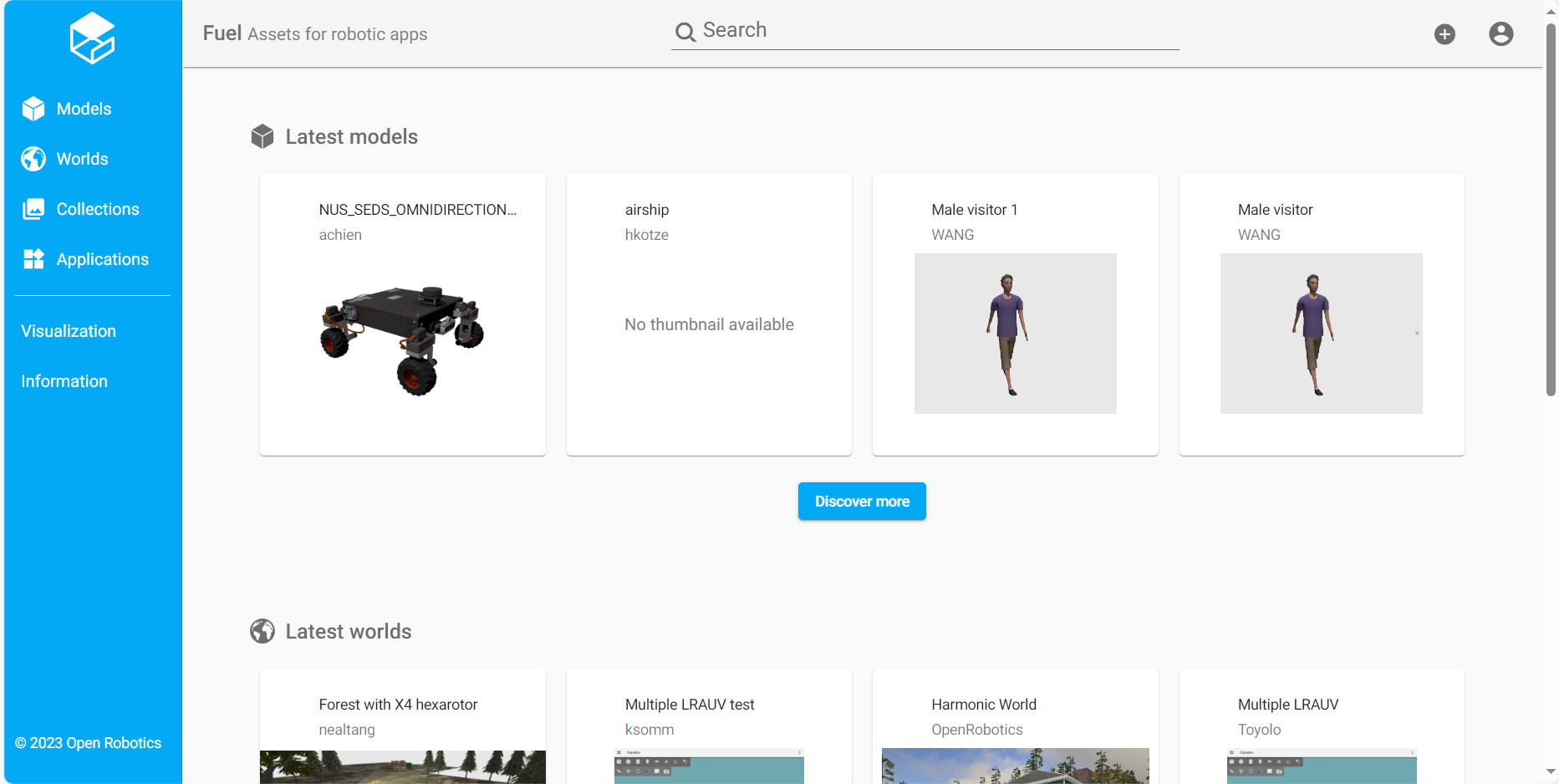
Modelle

# Fuel

Fuel ist die eine Sammlung an vorgefertigten kostenlosen SDF-Files, diese können in eigene Projekte eingebunden werden. Dabei kann es sich um Robotermodelle, Welten oder Wettermodelle handeln. Über den folgenden Link erreicht man Fuel:

<https://app.gazebosim.org/fuel>



Die fertigen Modelle können auf zwei Arten in die eigene Welt integriert werden, über die Website,

<include>

<uri>

https://fuel.gazebosim.org/1.0/OpenRobotics/models/Coke

</uri>

</include>

oder über den Download der Files auf den lokalen Rechner, dabei muss der heruntergeladene Orner im gleichen Verzeichnis liegen, wie die SDF-File.

<include>

<uri>

model://Coke

</uri>

</include>

# Eigenen Roboter erstellen

Jedes Modell beginnt mit dem model-Tag. In diesem sind alle Eigenschaften des Roboters enthalten.

<model name='vehicle\_blue'>

...

...

</model>

In diesen Tag kann man Eigenschaften wie die Startposition und -Orientation hinterlegen.

<pose relative\_to='world'>0 0 0 0 0 0</pose>

Man kann sehen, dass die Position in Relation zum Weltkoordinatensystem angegeben wird.

Ein Modell besteht immer aus Links, die durch Joints verbunden werden. Die Links sind dabei die 3D-Objekte und Joints Achsen und Gelenke an denen sich die Objekte bewegen können.

Ein Link wird folgendermaßen erzeugt:

<link name='chassis'>

<pose relative\_to='\_\_model\_\_'>0.5 0 0.4 0 0 0</pose>

...

...

</link>

Jedem Link kann auch eine eigen Position innerhalb des Modells zugewiesen werden, diese sind relativ zum Modellursprung.

Mit dem inertial-Tag wird die Masse bestimmt und die Auswirkungen von Kräften auf das Modell abgebildet.

<inertial> <!--inertial properties of the link mass, inertia matix-->

<mass>1.14395</mass>

<inertia>

<ixx>0.095329</ixx>

<ixy>0</ixy>

<ixz>0</ixz>

<iyy>0.381317</iyy>

<iyz>0</iyz>

<izz>0.476646</izz>

</inertia>

</inertial>

Der visual-Tag definiert das Aussehen des Modells.

<visual name='visual'>

<geometry>

<box>

<size>2.0 1.0 0.5</size>

</box>

</geometry>

<material>

<ambient>0.0 0.0 1.0 1</ambient>

<diffuse>0.0 0.0 1.0 1</diffuse>

<specular>0.0 0.0 1.0 1</specular>

</material>

</visual>

Da das Kollisionsmodell in den meisten Fällen einfacher ist als das Originalmodell, gibt es den collision-Tag. Dies wird gemacht, weil die Kollisionsberechnung mit komplexen Modellen sehr rechen intensiv sind.

<collision name='collision'>

<geometry>

<box>

<size>2.0 1.0 0.5</size>

</box>

</geometry>

</collision>

Danach kann der link-Tab geschlossen werden.

In der verlinkten XML-Datei ist ein Modell mit vier Links hinterlegt:

<https://github.com/JuSteif/GazeboLeitfaden/blob/main/snippets/ModellLinks.xml>

In diesem Beispiel wird für das Caster-Rad ein Frame genutzt. Dieser beinhaltet eine Pose auf die mehrere andere Links zugreifen können, was das gruppieren von diesen vereinfacht.

<frame name="caster\_frame" attached\_to='chassis'>

<pose>0.8 0 -0.2 0 0 0</pose>

</frame>

Der zweite Hauptbaustein aus dem ein Roboter besteht sind Joints. Diese halten die Links zusammen und bestimmen die Freiheitsgrade zwischen diesen.

<joint name='left\_wheel\_joint' type='revolute'>

<pose relative\_to='left\_wheel'/>

<parent>chassis</parent>

<child>left\_wheel</child>

<axis>

<xyz expressed\_in='\_\_model\_\_'>0 1 0</xyz>

<limit>

<lower>-1.79769e+308</lower>

<upper>1.79769e+308</upper>

</limit>

</axis>

</joint>

Wichtige Elemente des Joints sind der parent- und child-Tag. Ein Link kann mehrere Children haben, aber nur einen Parent. Dadurch entsteht eine Baumstruktur mit einem Link als Wurzel. Im axis-Tag wird angegeben um welche Achsen sich der Link bewegen kann. Wenn dieser Tag weggelassen wir, kann sich der Link in alle Richtung drehen.

Im Folgenden wurde das Robotermodell um die zugehörigen Joints ergänzt:

<https://github.com/JuSteif/GazeboLeitfaden/blob/main/snippets/ModellLinksJoints.xml>