Follow us

Search

Search

Introduction to V-REP (robotic arm)

By Leopoldo Armesto in Tutorial on 18 August, 2015.

Perfil del autor: Leopoldo Armesto

This is a V-REP tutorial to assemble an articulated robotic arm starting from the beginning.

V-REP (Virtual Robot Experimentation Platform) is a Robotics simulator of with a large capacity, features, and APIs (set of functions and methods to be used by other software).

The V-REP Robot Simulator has an integrated development environment (IDE) in which each model or object is based on a distributed control architecture (can be controlled by an own script, a plugin, a remote client application through its API, etc). This makes the software an ideal multi-robot environment simulation platform. The drivers for robots can be written in c/c++, Python, Java, Lua, Matlab or Urbi.

The following list are just some of the applications when using V-REP: Simulation of automated manufacturing systems.

- Remote monitoring.
- Hardware control.
- Rapid prototyping and verification.
- Security monitoring.
- Educational robotics.
- Product presentation.

V-REP can be used as an individual application (using your IDE) or can be embedded in a main client application. It has an interpreter for Lua (programming language), which makes it very versatile with the ability to combine high functionality and low level control to obtain spectacular results with few lines of code.

Robotic arm links

In order to introduce the different functionalities of the software, we will use a robot arm, which we call Robot UC3M (arm robotic of the Carlos III University of Madrid, by Antonio Castro). It's a 3D printable robot arm that can be printed at a very low cost, which has been slightly adapted with respect to the original design. After the completion of this tutorial, you will have a better knowledge of the simulator to face the rest of the tutorials. The expected result of this tutorial is shown in the figure:

Username or E-mail

Password

Login with your Social ID

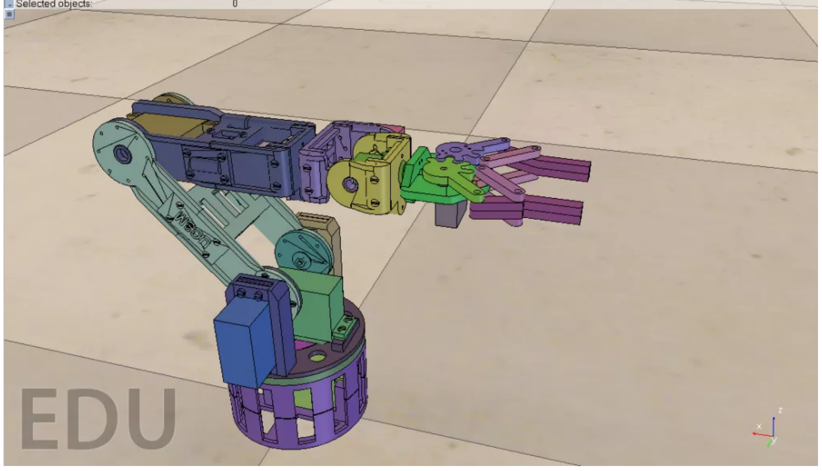
Remember Me

Log In

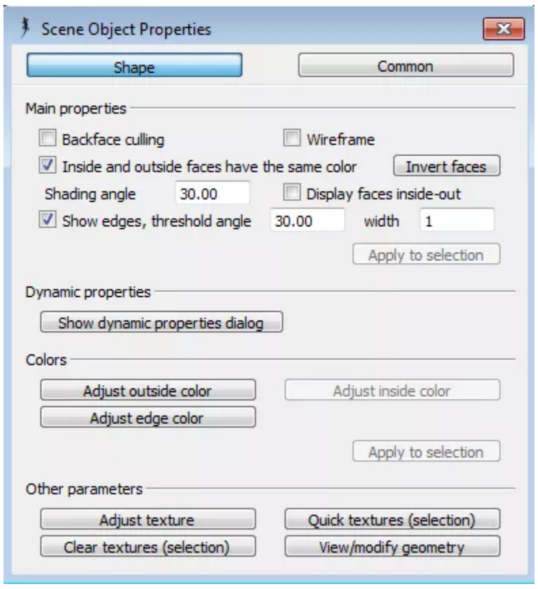
Register | Lost Password

News

No hay noticias



First of all, start a new session of V-REP. The simulator shows an empty scene by default. V-REP distinguishes two types of geometries, **pure** and **non-pure**. The **pure geometries** are primitive type **spheres**, **cylinders**, **prisms**, etc. that allow us the rapid calculation of collision, so they are better suited for *non-rendering* purposes. On the other hand, **non-pure geometries** often have a very high computational cost of collision calculation, since they are composed of facets. However, graphics cards are prepared for this purpose since they have specialized hardware that allows you to work with different geometries and render them, so they will be used for rendering purposes. Therefore, normally **non-pure geometries** will be used only for rendering (what the user sees), while **pure geometries** will be used to run own simulator routines such as collisions, dynamics, etc., that the user does not have to see. V-REP has a set of **sixteen layers** (structured in two rows of eight layers each) in which to locate each of the objects we create. By default **the first eight layers are visible** (the first row) and for convenience to make an object non-visible we will check its equivalent layer in the second row (the nine to the sixteen layers) and uncheck the corresponding first row layer. An object can be in multiple layers simultaneously. The layers in which an object is visible can be modified within the common object properties, as shown in the following figure and by double-clicking on the icon of the object and pressing the button "Common".


Scene Object Properties

Shape | Common

Main properties
- [ ] Backface culling
- [ ] Wireframe
- [✓] Inside and outside faces have the same color
- Invert faces
- Shading angle 30.00
- [ ] Display faces inside-out
- [✓] Show edges, threshold angle 30.00 width 1
- Apply to selection

Dynamic properties
- Show dynamic properties dialog

Colors
- Adjust outside color
- Adjust inside color
- Adjust edge color
- Apply to selection

Other parameters
- Adjust texture
- Quick textures (selection)
- Clear textures (selection)
- View/modify geometry

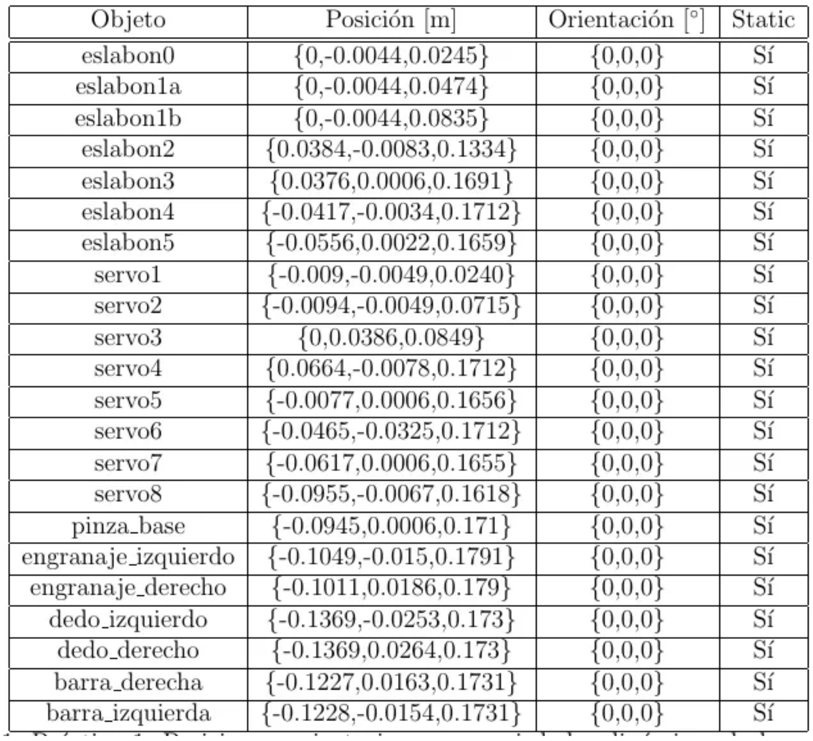
To generate non-pure geometries, we can either import them or create them. In our case, we import geometries from the STL files that are used to print the UC3M Robot in the files:

Ficheros V-REP Brazo Robótico 2.16 MB
Download

In [File, Import, Mesh], select the files (one by one): "eslabon0.stl", "eslabon1a.stl", "eslabon1b.stl", "eslabon3.stl", "eslabon4.stl", "eslabon5.stl", "eslabon6.stl", "servo1.stl", "servo2.stl", "servo3.stl", "servo4.stl", "servo5.stl", "servo6.stl", "servo7.stl", "servo8.stl", "pinza_base.stl", "engranaje_izquierdo.stl", "engranaje_derecho.stl", "dedo_izquierdo.stl", "dedo_derecho.stl", "barra_izquierdo.stl" and "barra_izquierda.stl". **Import units** are meters and the **Z vector points upwards**.

Change the name to each imported part to properly identify it. To do this, in the "Scene hierarchy" window, you will see an object with name "STL_Imported", which if we double click on its name, it will allows us to change it. To position/orient the object in the simulation environment must first select it and then click the icons "object/item shift" or "rotate object/item" , which can be found on the menu icon on the top.

Import the above mentioned pieces and place them at the position and orientation as indicated in table 1. Parts should be correctly positioned by default, while orientation of each part might depend on the own part geometry. For convenience, after you import each of the pieces, you need to set it by doing [Edit, Reorient bounding box, with reference of world]. After that, all pieces will be in the configuration that we coined as *HOME*. **Make sure that the *Static* option is enabled**, as specified in table 1. To do so, you have to check it within the dynamic properties of the object after you click the button *Show dynamic properties* dialog within the *Scene Object Properties* dialog box. Static objects can not be directly actuated through joints controlled by torque/force, indeed, its position and orientation will depend on the position and orientation of other objects. They have no dynamic, i.e. neither mass nor inertia and **usually we will use them for non-pure geometry objects**, although there may be cases, where we want a pure geometry without dynamic properties. If a joint had to move a dynamic object, then you must have to set it to force/torque mode or check the hybrid mode, although at the moment this is not necessary. **Make sure that the position and the orientation are the ones as indicated in table 1**.



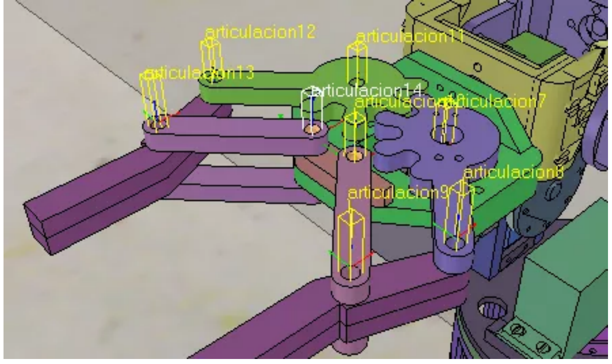
Objeto	Posición [m]	Orientación [°]	Static
eslabon0	{0,-0.0044,0.0245}	{0,0,0}	Sí
eslabon1a	{0,-0.0044,0.0474}	{0,0,0}	Sí
eslabon1b	{0,-0.0044,0.0835}	{0,0,0}	Sí
eslabon2	{0.0384,-0.0083,0.1334}	{0,0,0}	Sí
eslabon3	{0.0376,0.0006,0.1691}	{0,0,0}	Sí
eslabon4	{-0.0417,-0.0034,0.1712}	{0,0,0}	Sí
eslabon5	{-0.0556,0.0022,0.1659}	{0,0,0}	Sí
servo1	{-0.009,-0.0049,0.0240}	{0,0,0}	Sí
servo2	{-0.0094,-0.0049,0.0715}	{0,0,0}	Sí
servo3	{0,0.0386,0.0849}	{0,0,0}	Sí
servo4	{0.0664,-0.0078,0.1712}	{0,0,0}	Sí
servo5	{-0.0077,0.0006,0.1656}	{0,0,0}	Sí
servo6	{-0.0465,-0.0325,0.1712}	{0,0,0}	Sí
servo7	{-0.0617,0.0006,0.1655}	{0,0,0}	Sí
servo8	{-0.0955,-0.0067,0.1618}	{0,0,0}	Sí
pinza_base	{-0.0945,0.0006,0.171}	{0,0,0}	Sí
engranaje_izquierdo	{-0.1049,-0.015,0.1791}	{0,0,0}	Sí
engranaje_derecho	{-0.1011,0.0186,0.179}	{0,0,0}	Sí
dedo_izquierdo	{-0.1369,-0.0253,0.173}	{0,0,0}	Sí
dedo_derecho	{-0.1369,0.0264,0.173}	{0,0,0}	Sí
barra_derecha	{-0.1227,0.0163,0.1731}	{0,0,0}	Sí
barra_izquierda	{-0.1228,-0.0154,0.1731}	{0,0,0}	Sí

Table 1. Positions, orientations and dynamic properties of the UC3M robot parts.

Robotic arm joints

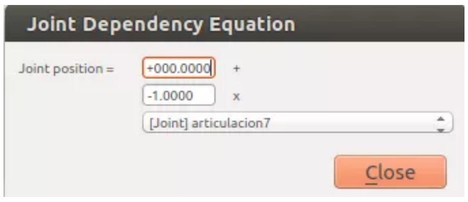
The joints may be of type revolution, prismatic or spherical. In our case, every joint of the robot is of the type revolution. To create a joint [Menu Add, Joint, Revolute]. We rename the newly created joint and introduce the name "articulacionX" where "X" is the number of the joint. For convenience the joint name starts from number 1.

Joints 1 to 6 correspond to the joints centered on the servo shafts of the robot arm, while joints of the 7 to 10 correspond to servo 8 and the holes of the left finger of the gripper and joints 11 to 14 are the holes of the right finger (including the right gear). Joints 10 and 14 (but joint 11) correspond to the holes in the pinza_base (connected to the bars). See the figure below for further clarification of name assigned to each of the joints of the gripper.



By default, the joints are in the mode *joint force/torque mode*, however, for the moment we will use *joint is in passive mode* for the joints of 1 to 7, since all objects are static.

In 11 joint, the right gear, is a dependent joint of 7 joint, so we will select the *joint is in dependent mode* and then press the button "Adjust dependency equation" and enter the equation as shown in the following figure. This mode will allow us to simulate the gear mechanism in such a way that the angle of a joint will be replicated by the other.


Joint Dependency Equation

Joint position = +000.0000 +
-1.0000 x
[Joint] articulacion7

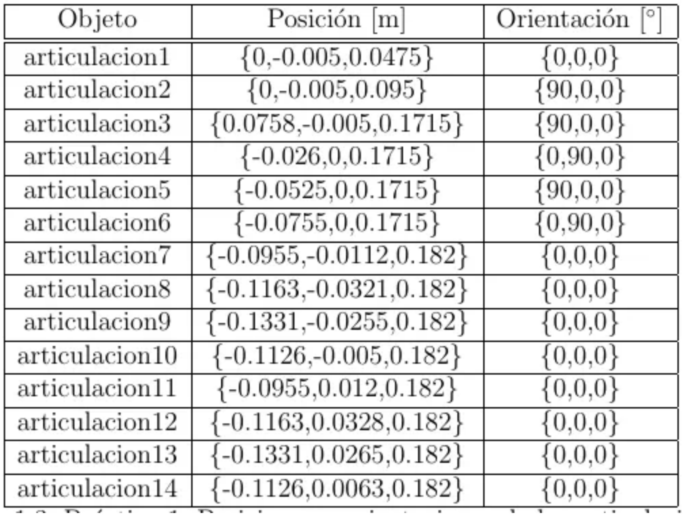
Joints from 8 to 14 (with the exception of joint 11) will be set to *joint is inverse kinematics mode*, that will allow V-REP to calculate the orientation that they should have in order to solve the mechanism of the gripper.

In the properties of each joint, *Scene Object Properties* dialog box, we can also customize its appearance such as the *Joint Length* and *Joint diameter* parameters. You can establish such values as specified in table 2. By default the joints appear in the second layer (in the section on the common properties), while for convenience, we will move them to the tenth layer (on the second row).

Table 2 shows the properties that every joint should have, while table 3 shows their position and orientation (**check the position and orientation are correct**). Properties of joints are located in the dialog *Scene Object Properties*, being joints of 1 to 8 and joint 12 not cyclic (the *Position is cyclic* box must be disabled) with the ranges specified in table 2. For joints from 9 to 14 (but joint 12), since they are not motorised or limited by any mechanism, we leave them with the cyclic option enabled.

Objeto	Modo	Long. [m]	Diám. [m]	Pos. Min. [°]	Rango [°]
articulacion1	Pasivo	0.003	0.1	-90	180
articulacion2	Pasivo	0.1	0.01	-20	90
articulacion3	Pasivo	0.1	0.01	-67.5	90
articulacion4	Pasivo	0.025	0.01	-90	180
articulacion5	Pasivo	0.1	0.01	-90	180
articulacion6	Pasivo	0.025	0.01	-90	180
articulacion7	Movimiento	0.02	0.0025	-15	45
articulacion8	Cin. Inversa	0.02	0.0025	-	-
articulacion9	Cin. Inversa	0.02	0.0025	-	-
articulacion10	Cin. Inversa	0.02	0.0025	-	-
articulacion11	Dependiente	0.02	0.0025	-30	45
articulacion12	Cin. Inversa	0.02	0.0025	-	-
articulacion13	Cin. Inversa	0.02	0.0025	-	-
articulacion14	Cin. Inversa	0.02	0.0025	-	-

Table 2. Properties of joints.

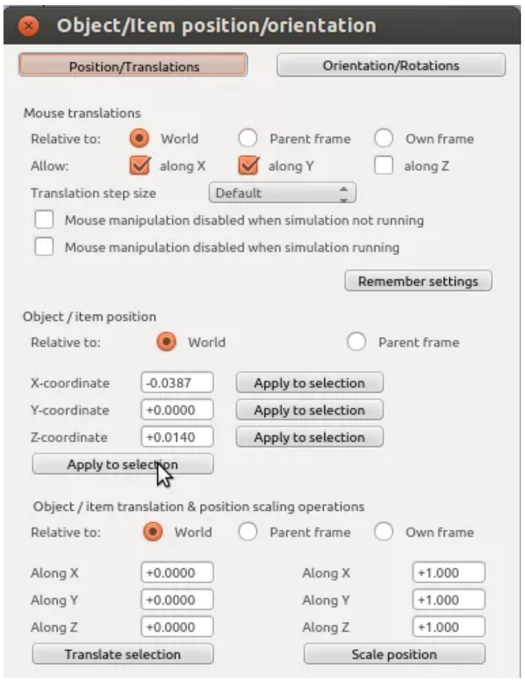


Objeto	Posición [m]	Orientación [°]
articulacion1	{0,-0.005,0.0475}	{0,0,0}
articulacion2	{0,-0.005,0.095}	{90,0,0}
articulacion3	{0.0758,-0.005,0.1715}	{90,0,0}
articulacion4	{-0.026,0,0.1715}	{0,90,0}
articulacion5	{-0.0525,0,0.1715}	{90,0,0}
articulacion6	{-0.0755,0,0.1715}	{0,90,0}
articulacion7	{-0.0955,-0.0112,0.182}	{0,0,0}
articulacion8	{-0.1163,-0.0321,0.182}	{0,0,0}
articulacion9	{-0.1331,-0.0255,0.182}	{0,0,0}
articulacion10	{-0.1126,-0.005,0.182}	{0,0,0}
articulacion11	{-0.0955,0.012,0.182}	{0,0,0}
articulacion12	{-0.1163,0.0328,0.182}	{0,0,0}
articulacion13	{-0.1331,0.0265,0.182}	{0,0,0}
articulacion14	{-0.1126,0.0063,0.182}	{0,0,0}

Table 3. Positions and orientations of the joints.

In addition to the joints, you need we create objects of type *dummy*. They are objects whose purpose is no other than the get its position or link them with certain objects. Typically we will use them to calculate the inverse kinematics of robot. We will create four *dummy* type objects and position them in the two common holes between the "pinza_base" object and objects of bars (two *dummies* objects on each side) in [Menu Add, Dummy]. Rename the objects *dummy* to "dummyIzqPinza", "dummyIzqBarra" for the hole on the left, and "dummyDerPinza", "dummyDerBarra" for the *dummies* of the hole on the right. **It is important that pairs of *dummies* (belonging to the same kinematic chain, either left or right) have the same position and orientation. Thus, we need to position *dummies* on the left in the same position as the joint 10 and *dummies* on the right in the same position as joint 14**.

V-REP allows you to copy the properties of an object from another object. You can, for example, select "dummyIzqPinza" and, while holding the SHIFT button pressed, select the joint object (from which we want to copy the properties). Then, to copy the position property, access the "Object/Item position/orientation" dialog box by clicking on the icon rotation and press the button "Apply to selection", as shown in the following figure.



Object/Item position/orientation

Position/Translations | Orientation/Rotations

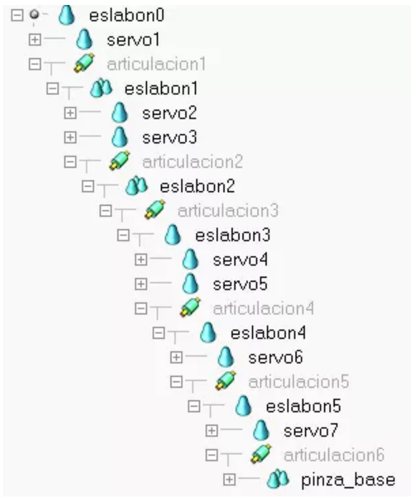
Mouse translations
Relative to: [✓ World] Parent frame Own frame
Allow: [✓ along X] [✓ along Y] along Z
Translation step size: Default
Mouse manipulation disabled when simulation not running
Mouse manipulation disabled when simulation running
Remember settings

Object / item position
Relative to: [✓ World] Parent frame
X-coordinate: -0.0387 Apply to selection
Y-coordinate: +0.0000 Apply to selection
Z-coordinate: +0.0140 Apply to selection
Apply to selection

Object / item translation & position scaling operations
Relative to: [✓ World] Parent frame Own frame
Along X: +0.0000 Along X: +1.000
Along Y: +0.0000 Along Y: +1.000
Along Z: +0.0000 Along Z: +1.000
Translate selection | Scale position

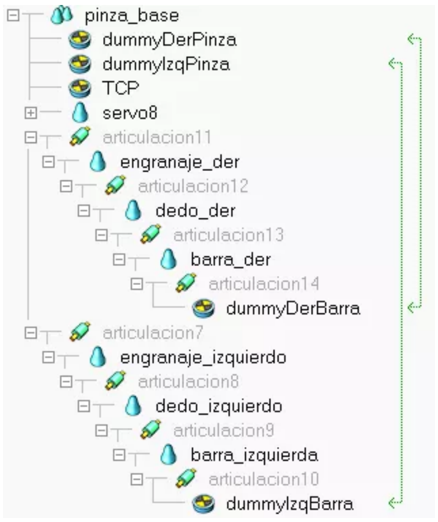
Parent relationship

Now is the time to establish the relationship between links, the joints and the rest of the simulation objects. The base object is the eslabon0, to which we will activate the option *Object model is based* on the common properties of the *Scene Object Properties* dialog box. You will see that a dot appears just to the left of the icon in objects that are base models after activation (now when copying and pasting a model based object will also copy all child objects). This will allow to treat all the robot as a single object (i.e.: to facilitate copy between different scenes). The following figure shows the parent relationship between different objects. To make a *child* child of another object (*parent*), you simply need to select the *child* object on its icon, drag it until the *parent* object and drop it. To undo the parent relationship, repeat the operation, but releasing the object on the main scene.



```
eslabon0
  servo1
  articulacion1
    eslabon1
      servo2
      servo3
      articulacion2
        eslabon2
          articulacion3
            eslabon3
              servo4
              servo5
              articulacion4
                eslabon4
                  servo6
                  articulacion5
                    eslabon5
                      servo7
                      articulacion6
                        pinza_base
```

Expanding pinza_base we have:



```
pinza_base
  dummyDerPinza
  dummyIzqPinza
  TCP
  servo8
  articulacion11
    engranaje_der
      articulacion12
        dedo_der
          articulacion13
            barra_der
              articulacion14
                dummyDerBarra
  articulacion7
    engranaje_izquierdo
      articulacion8
        dedo_izquierdo
          articulacion9
            barra_izquierda
              articulacion10
                dummyIzqBarra
```

In general, in the case of open kinematic chains, *we will make links depending on the joint with the same number*. The object eslabon0 does not move and therefore (at the moment) does not depend on any other object.

For closed chains cinematic such as the gripper, we will use *dummies* such as "dummyIzqPinza" and "dummyDerPinza" depending on "pinza_base" and activate the inverse kinematic mode to make sure the chain is properly closed. The *dummies* of the bar (the other end of the kinematic chain) must be children of the last of the joints.

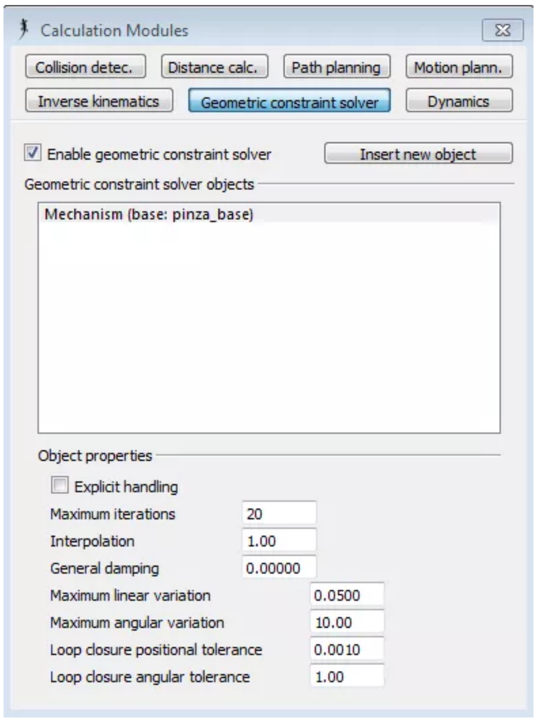
The gripper mechanism

Now, we will set up the gripper mechanisms, so that VREP calculates the values of the joints (from 8 to 14, with the exception of the 11 joint). The aim is to establish a geometric constrain between **dummies** objects that we created earlier, so that whatever the value of the 7 joint (associated with the servo 8), VREP should calculate corresponding angles so that the **dummies** always remain coincident.

Select in the properties of each of the *dummies* to be linked to to the corresponding *dummy* and in the drop-down menu "Link type" select "GCS, overlap constraint". You can observe that in the window "Scene Hierarchy" the *dummies* are now linked with an arrow's color (green for the case of the relationship "constraint overlap GCS").

V-REP allows you to set certain relationships to the calculation of inverse kinematics. At the moment, we will just use the GCS (*Geometric Constraint Solver*) tool that allows you to set the values of the joints in the case of closed chain mechanisms, as in the case of the gripper mechanism. To access to "Calculation

Modules" dialog box, select [Menu Tools, Calculation modules] with "pinza_base" object selected. Among all the options, we will focus, for the moment in the calculation carried out by the "Geometric constraint solver" (see the following figure), by pressing the corresponding button. If you press the button "Insert new object", we have established the geometric relationship between the objects.



[✓] Enable geometric constraint solver

Geometric constraint solver objects

Mechanism (base: pinza_base)

Object properties

[ ] Explicit handling

Maximum iterations	20
Interpolation	1.00
General damping	0.00000
Maximum linear variation	0.0500
Maximum angular variation	10.00
Loop closure positional tolerance	0.0010
Loop closure angular tolerance	1.00

Now, we are ready to simulate the robot. If you press the button "Start / Resume simulation" right in the toolbar at the top, the simulation will start. At the moment, our simulation it does nothing impressive, but at least you have already managed to assemble your first robot in V-REP and learn several basic aspects.

Like this:

Loading...

← ROFI, biped to humanoid robot

Acquisition of biosignals using BiTalino and LabVIEW →

-->