URFD Changing worlds

Now that we know a bit about urdf and xacro, we can start working towards getting our factory simulation ready.

This is the factory that you've seen in the welcome video at the start of this course.

We have a slight problem though: this version of it is not currently setup correctly.

To get it into a shape where we can start thinking about mobile robot navigation and picking and placing parts, we'll have to make some changes.

First of all, we can't have all these bins here.

The mobile robot that will be introduced in week 3 cannot come close enough to the robot like this.

And we actually only need a bin at the other end of the factory for the turtlebot to deliver the part it's carrying to.

So we're going to repurpose this bin for that and move it over here.

Apart from these things, we also need to add the second robot to the factory and add something to mount it on.

So all in all, we have the following tasks to complete:

remove four out of five bins,

move the remaining bin to the correct location,

we add a mount,

and add the second robot.

Let's get started with the first task: removing the bins.

First open the file that contains the urdf in a text editor.

I'm going to use Atom as an editor, but any text editor will do.

Using an editor with support for xml syntax highlighting though is a good idea, as it makes reading a urdf or xacro file a bit easier.

So now that we have opened the main -- or top-level -- xacro file, let's have a quick look around.

As you can see here, the xacro declares a name and a namespace, which we'll ignore for now.

The name is the name of our scene. It says 'robot' there, but that is just the name of the root element, we can definitely model other things with urdf, as we will see later.

Next up is a definition of an empty link with the name 'world'.

It's always a good idea to have a clear starting point of your scene, a link that everything else is connected to.

Typically we choose names like 'base_link' or 'world'.

As we'll also be working with Gazebo later, we have a 'world' here.

We then see a number of xacro:include statements followed by a call to the macro they define.

Here for instance the model for the UR10 near the conveyor belt is imported.

And here we see the model actually being added to the world.

Notice the two arguments it gets passed: prefix and joint_limited.

After importing all the building blocks of the world and adding the models they define, they are still unconnected, they have not been placed anywhere yet.

So further down we see a number of joints that take care of this.

Here we have the joint that connects the suction gripper to the robot's toolframe, and here we have the joints that place the bins next to the conveyor and the robot.

Notice how each joint always has a 'parent' and a 'child', where the child link refers to what should be connected and the parent link refers to what something should be connected TO.

The origin elements specify the relative offset between the parent and the child. In other words: at which distance from the parent should the child link be placed in the world?

So now that we have an idea of the structure of the file, let's start changing it.

As we decided earlier, we need to remove some of the bins.

To remove a model from a urdf world, at least two things should be removed: the joint that connects it to its parent, and the macro call that adds it to the world.

When you're removing a model of which only a single instance exists in the world, you could also just as well remove the model import statement, as it is not needed any longer. But you could also just leave it, as the model definition will just be ignored if it’s never instantiated.

Removing the bins is relatively straightforward: there are 5 model instantiations (or macro calls) and 5 fixed joints... one for each bin.

We'll have to remove 4 of the joints and 4 of the macro calls.

Let's start with the joints.

They are at the end of the file, so let's scroll down and select the last four. We'll leave 'bin_1_joint', as that is needed for the bin that we keep around.

So we select it... and we delete the lines.

With the joints gone we must remove the model instantiations as well.

Before we do that though, let's see what happens if we'd forget that.

And instead of launching RViz to see what happens, let's use the 'check_urdf' tool we saw earlier.

So we have to go back to the terminal and

we have to change to the correct directory.

Now 'check_urdf' only knows how to check urdfs, and the factory file is a xacro, so we'll first have to convert the xacro to a urdf.

And we can do that like this.

Now we can use the 'check_udrf' tool.

As expected, ‘check_urdf’ is not happy. It finds two root links, and as we remember, urdf only supports scenes with a single root, so this is not a valid urdf.

To fix this problem, we'll just have to remove the model instantiations or xacro macro calls.

So we have to go back to the editor, we scroll up a bit to where the instantiations are, we select them and then we delete them.

Now everything should be fine.

A quick check with check_urdf.

That looks good.

So now let's see the result in RViz.

Use the file 'visualize_hrwros.launch' to start the visualization.

Great. Now there is only a single bin left.

Only thing remaining is to move the bin to the correct location.

As a quick reminder: we're going to move the bin to here, behind the pallet with the boxes on it.

Remember that all joints define transforms between links and the origins of joints allow us to specify the exact position relative to the center of the factory world in this case.

The joint for the bin already has an origin, so we just need to update that one with the right values for the X and Y coordinates.

Bin 1 is currently at 0.3 meters in the negative-X direction and 1.2 meters in the positive-Y direction.

Relative to the world frame.

So let's update those values to move the bin.

The location behind the pallet is at 8.0 meters in the negative-X direction and at 2.2 meters in the negative-Y direction.

Done.

Let's start RViz again to see the result.

Don't forget to save the file.

Switch to the terminal and start RViz again using the 'visualize_hrwros.launch' file.

It looks like everything is in order.

The bin is nicely situated behind the pallet with the boxes and there's plenty of space remaining to place the second robot next to it. The turtlebot should also have no problems reaching this location.

With that we've completed our first two tasks.

But we're far from done, as we're still missing the second robot.

Continue to the next video to learn how we can add things to a world and start working on the two remaining tasks.