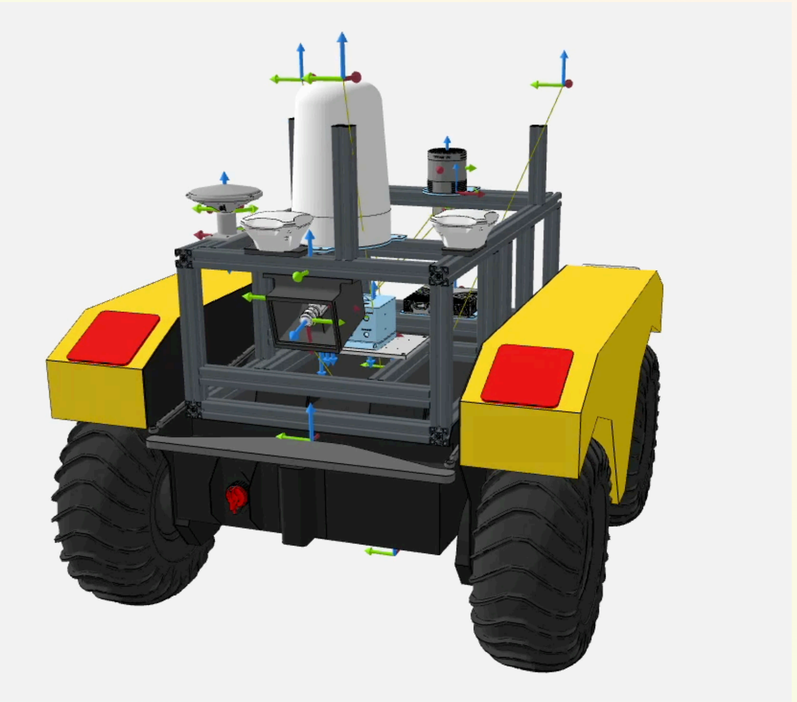
ONSHAPE-TO-URDF SEMINAR

NORLAB - MATEJ BOXAN - SUMMER 2024

OVERVIEW



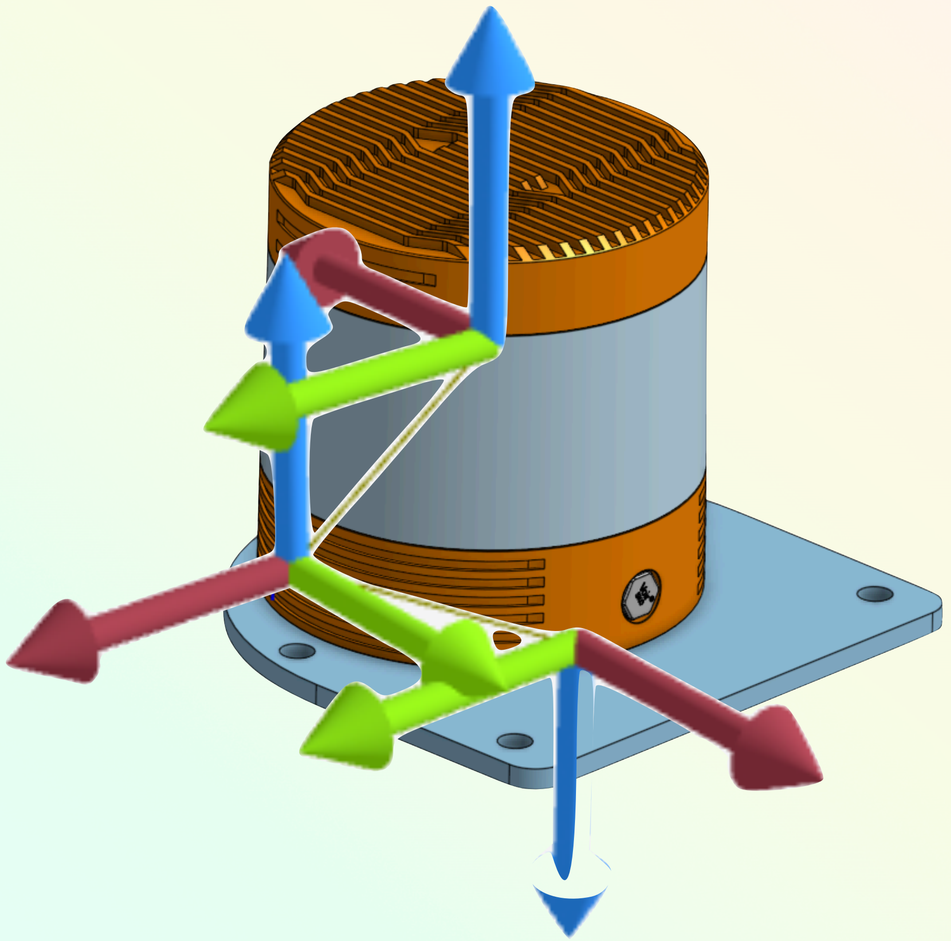
- Onshape-to-robot is a Python tool that converts Onshape assemblies into .urdf files
- Urdf = basically .xml or .html file
- Xacro = urdf with added sauce
- https://onshape-to-robot.readthedocs.io/en/latest/index.html
- We will create an Onshape assembly and export it to urdf together

CONVERSIONS

ONSHAPE -> URDF

In Onshape: parts connected with Mates

In urdf: links connected with joints



Simple mate -> link

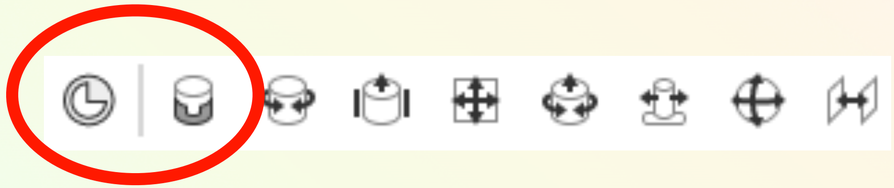
Fastened mate -> joint (type="fixed")

Other mates -> different joints

BUT ONLY WHEN USING A CORRECT PREFIX IN ONSHAPE

IT'S ALL ABOUT MATES

NAMING CONVENTIONS AND OTHER JOYS

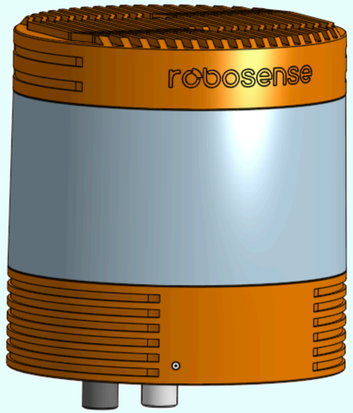


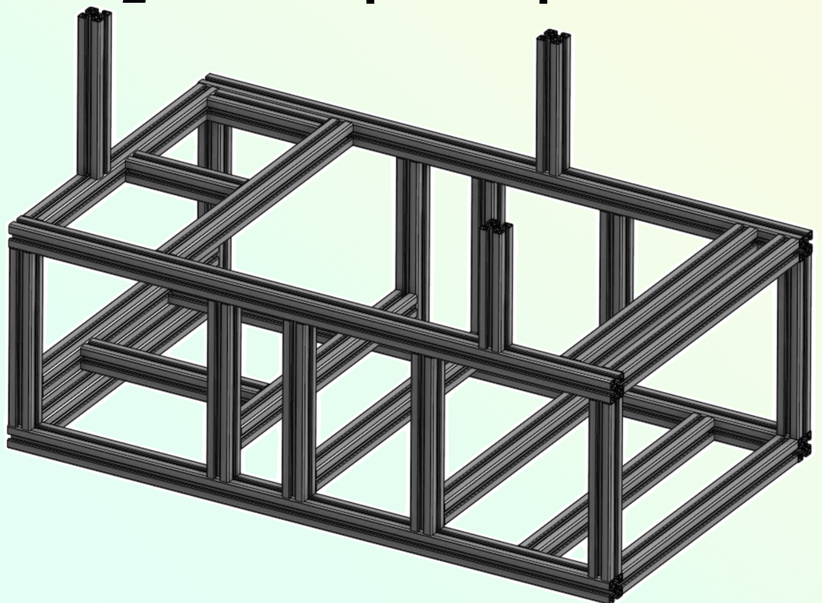
- Use link_<name> of you want to rename a link in urdf (link_base_link -> base_link), otherwise the part name is used.
- Use mount_<name> for a mate that you attach to (mount_RS128_plate)
- Use mate_<name> for a mate that's on the part being attached (mate_RS128_plate)
- Use dof_<name_from_name_to> for a Fastened mate to appear in urdf (dof_RS128_plate_frame -> RS128_plate_frame)

- The values of <name> in mount_<name> and mate_<name> do not appear in the urdf file. Part names are used instead.

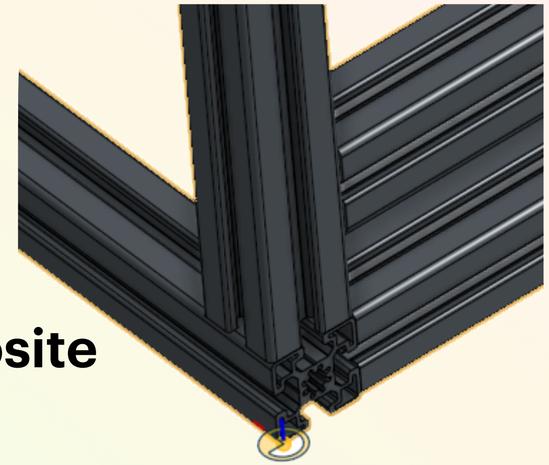
STEPS: IMPORT TO ASSEMBLY

- Copy project *norlab_onshape-to-urdf_seminar* from Norlab team
- Create a closed composite part and rename it to `structure_extrusions_composite`
- Create a new assembly called `assembly URDF export` and import the composite part
- Import from warthog's project: *rs128_plate* and *robosense_128* composite part
- Pay attention to project versions!



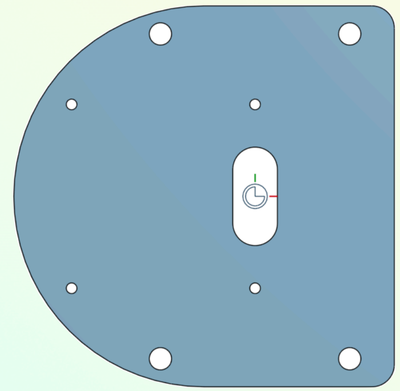


STEPS: CREATE MATES



RS128 PLATE AND ALIMINUM EXTRUSIONS

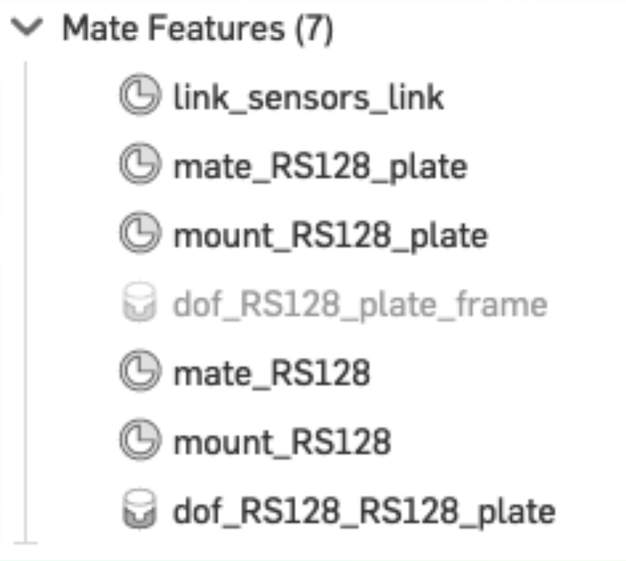
- Create a Mate connector in the front bottom right corner of the composite part and name it `link_sensors_link`
- Create a Mate connector in the middle of the big hole on the bottom of the RS128 plate and name it `mate_RS128_plate`
- Create a Mate connector in the to mount the RS128 plate and name it `mount_RS128_plate`
- Create a Fastened mate between `mate_RS128_plate` and `mount_RS128_plate` and name it `dof_RS128_plate_frame`



STEPS: CREATE MATES

RS128 AND RS128 PLATE

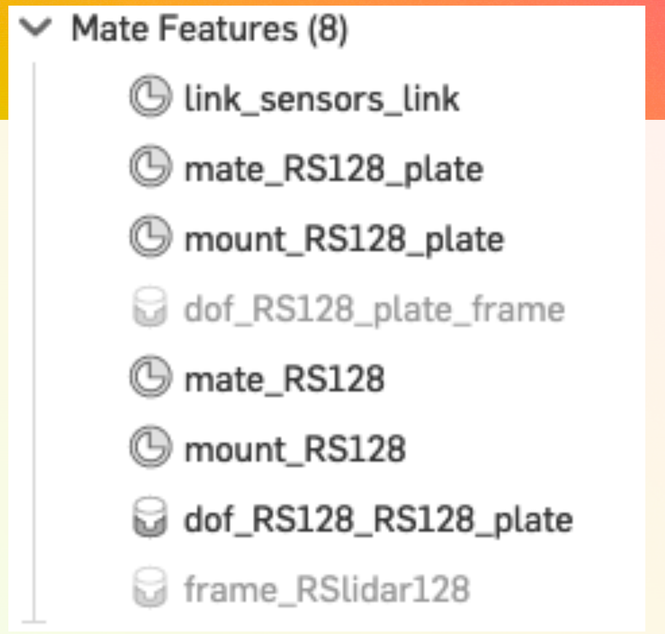
- Create a Mate connector in one of the screw holes in the RS128 and name it `mate_RS128`
- Create a Mate connector in one of the screw holes of the RS128 plate and name it `mount_RS128`
- Create a Fastened mate between `mate_RS128` and `mount_RS128` and name it `dof_RS128_RS128_plate`
- We should now have 7 Mate Features in our document

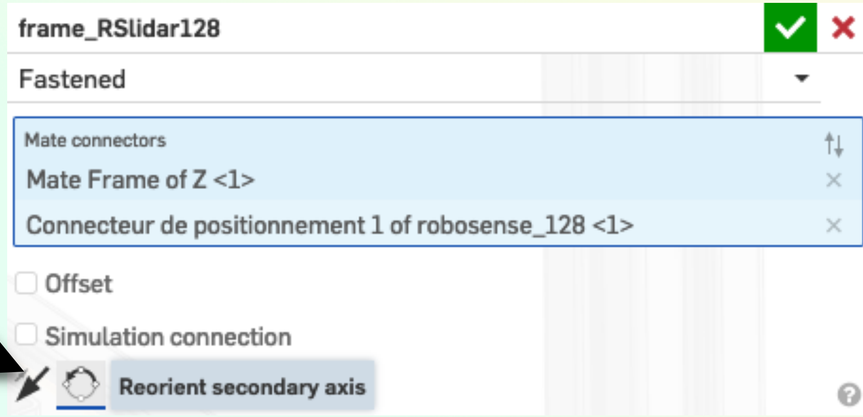


STEPS: CREATE MATES

RS128 OPTICAL SENSOR ORIGIN FRAME

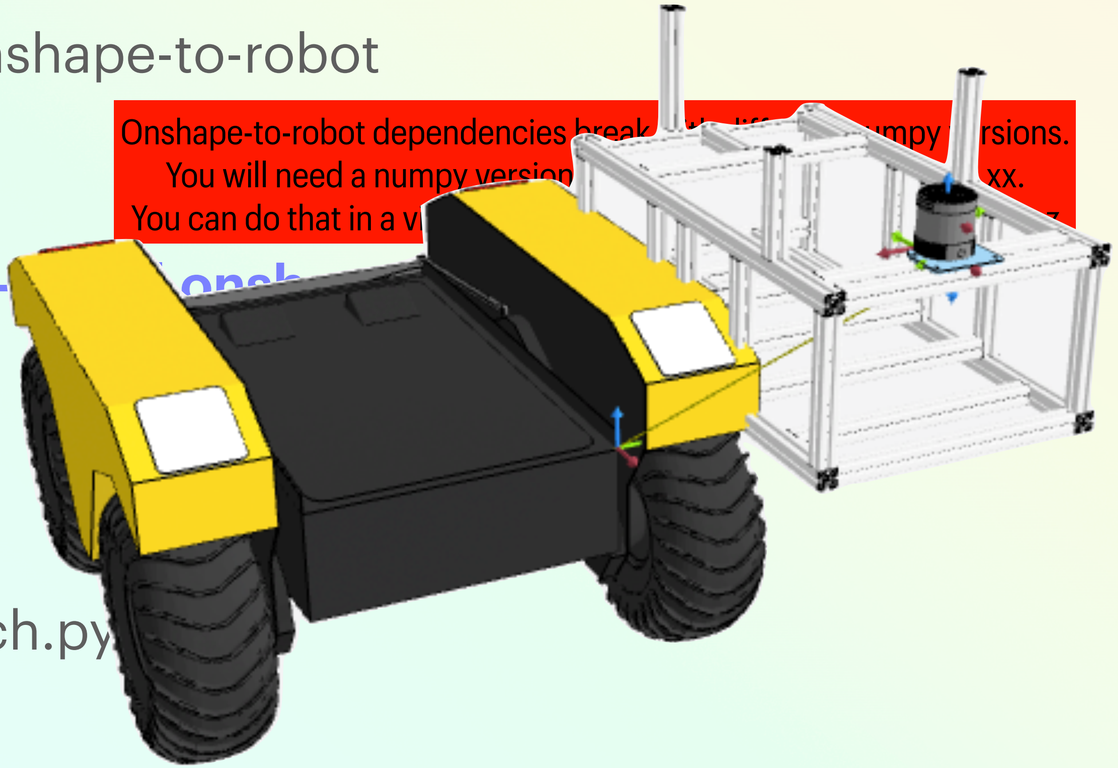
- Insert *Frame URDF export* assembly from the norlab team
- Create a Fastened mate between the mate in robosense_128 `Connecteur de...` and `Mate Frame 1` in Frame <1>'s sub-assembly.
- Name it `frame_RS128_link` -> `RS128_link`
- Rotate it to match the orientation of the `Connecteur de...`
- We need these custom frames to define our sensors frames.





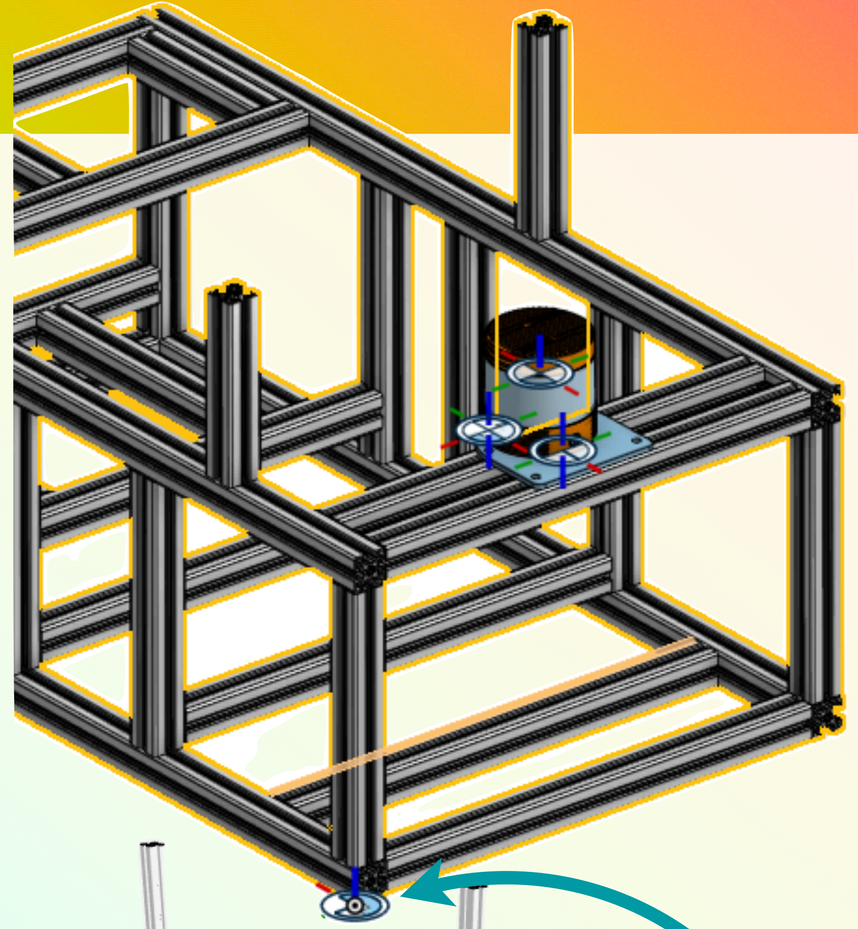
STEPS: EXPORT URDF

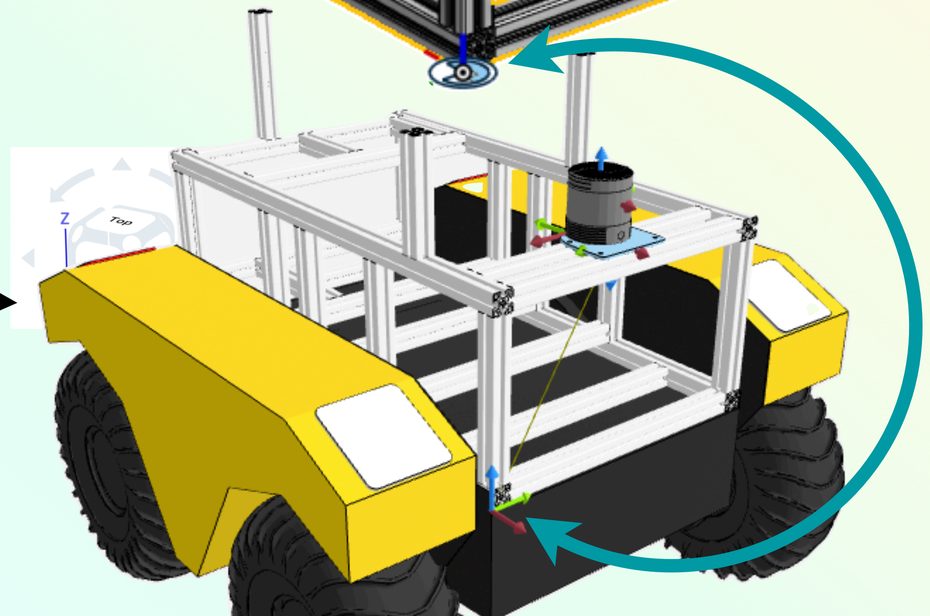
- **Clone the norlab-robot repo in your ros2_ws/src:**
  git clone -b onshape-to-urdf-seminar git@github.com:norlab-ulaval/norlab_robot.git
- **Install the onshape-to-robot tool:** pip install onshape-to-robot
- **Update the configuration file:**
  - **Access key and secret key from** https://dev-
  - **Document ID and assembly name**
- **Call** onshape-to-robot .
- **Call** ros2 launch norlab_robot description.launch.py
- **Visualize with your preferred method**



STEPS: FIX URDF

- Orientation and translation between the sensors frame and base_link comes from the Assembly origin
- Create a new mate connector called `origin` in the assembly origin
- Create a new Fastened mate between `link_sensors_link` and `origin`
- Make sure that you rotate the Fastened mate to copy the Onshape world orientation
- Regenerate the urdf, build and visualize again
- Note that the `sensors_link` is oriented differently in Onshape and urdf





STEPS: ROS MESSAGES AND CALIBRATION

- **On ROS side, make sure that your message publisher uses the correct frame_id: whatever was after the frame_ prefix in onshape.**
- **Calibrated frames are in the calibrated.urdf.xacro file.**
- **Add a new link named `RS128_calibrated_link` in calibrated.urdf.xacro:**
  <link name="RS128_calibrated_link" />
- **Add a new joint named `RS128_calibrated_joint`:**
  <joint name="RS128_calibrated_joint" type="fixed">

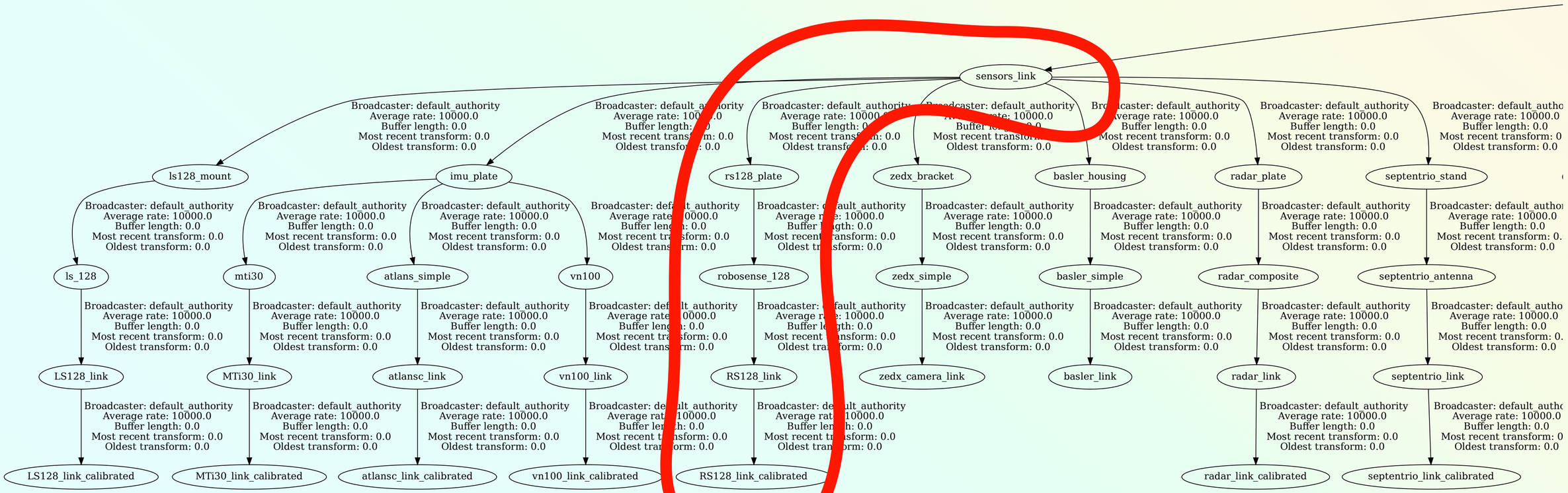
    <origin xyz="0.0 0.0 0.0" rpy="0 0 0" />

    <parent link="RS128_link" />

    <child link="RS128_calibrated_link" />

  </joint>

WARTHOG'S TREE



PRACTICE MAKES PERFECT

… SO LET’S REDO THE REST!

… OR MAYBE NEXT TIME!