Addressing the problem of camera extrinsic calibration using Hand-Eye Calibration technique.

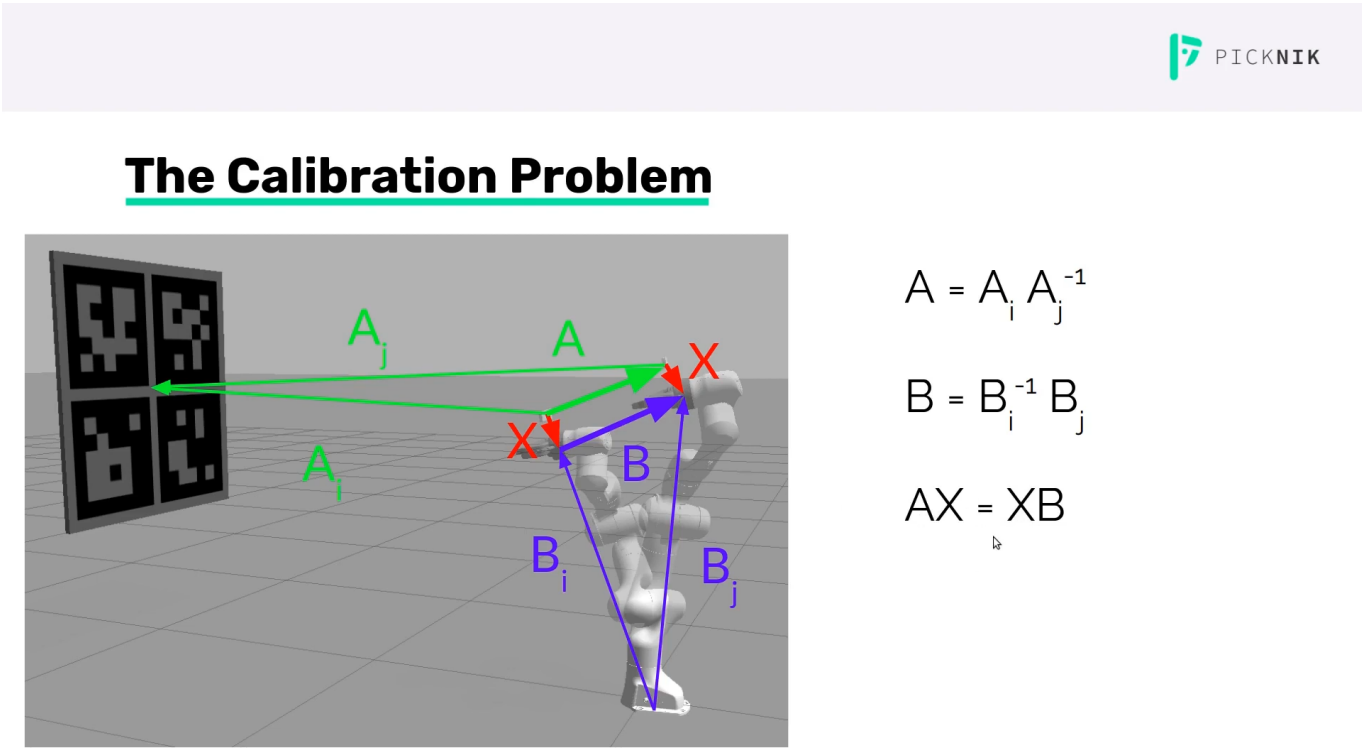
The following instructions are written for and tested on Ubuntu 20.04, ROS Noetic - Moveit1.

Written by Prasanth Suresh (ps32611@uga.edu).

The problem:

As of today(07/20/22), this feature is only available on ROS-Noetic. This technique addresses the problem of finding the camera location wrt the world coordinate system based on the robot's transformation tree wrt the world origin and the calibration marker tag it sees in various poses.

It essentially solves the problem shown in the image below (in this scenario, the camera is attached to the end effector, but the camera could also be rigidly mounted somewhere):



If it is the above case, we move the endeffector, and by extension, the camera to view a steadily placed marker tag at a distance. If the camera is mounted elsewhere, we attach the marker stiffly to the endeffector and move the marker around so that the camera captures the marker in different poses.

Make sure to include more rotational angles in either case, that helps build more diverse input points and ergo a more accurate solution.

The dependency problem:

While just cloning moveit_calibration package and resolving the dependencies using:

```
rosdep install -y --from-paths . --ignore-src --rosdistro noetic
```

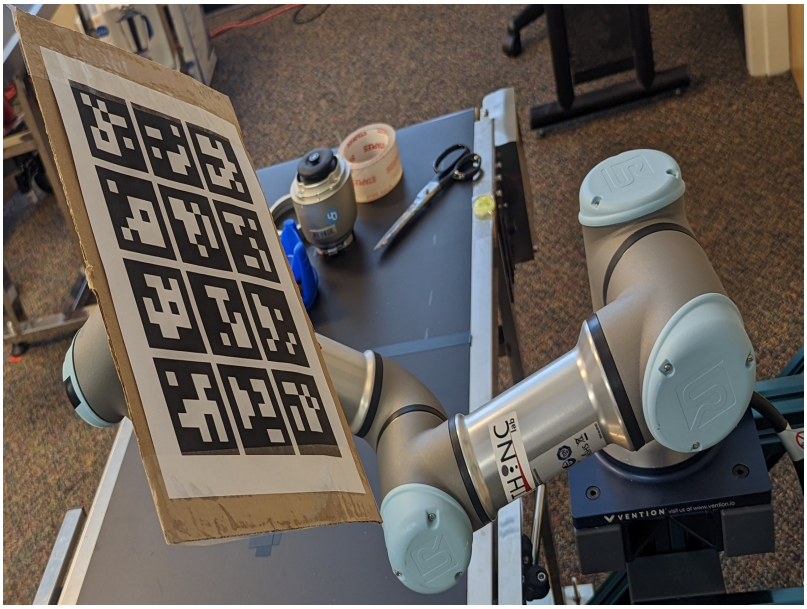
And building the package should get you setup, there are some things you need to be wary of:

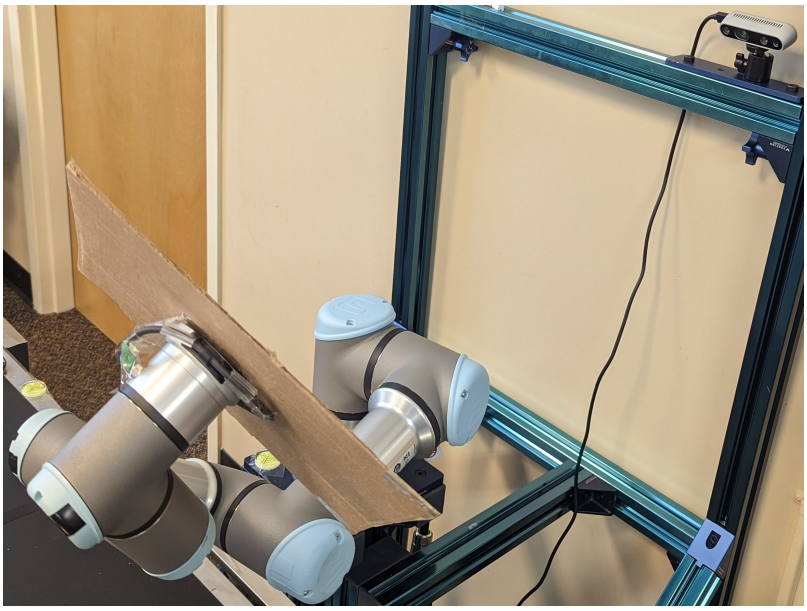
1. If you're using realsense camera, rosdep install command above will try install the debian version of realsense driver. Cancel it or uninstall it once it finishes and make sure to use this method and check if the versions match.
2. There's a broken library that the handeye calibration moveit plugin depends on. Uninstall everything related to libopenblas and instal from source as mentioned here.

Now you're good to go!

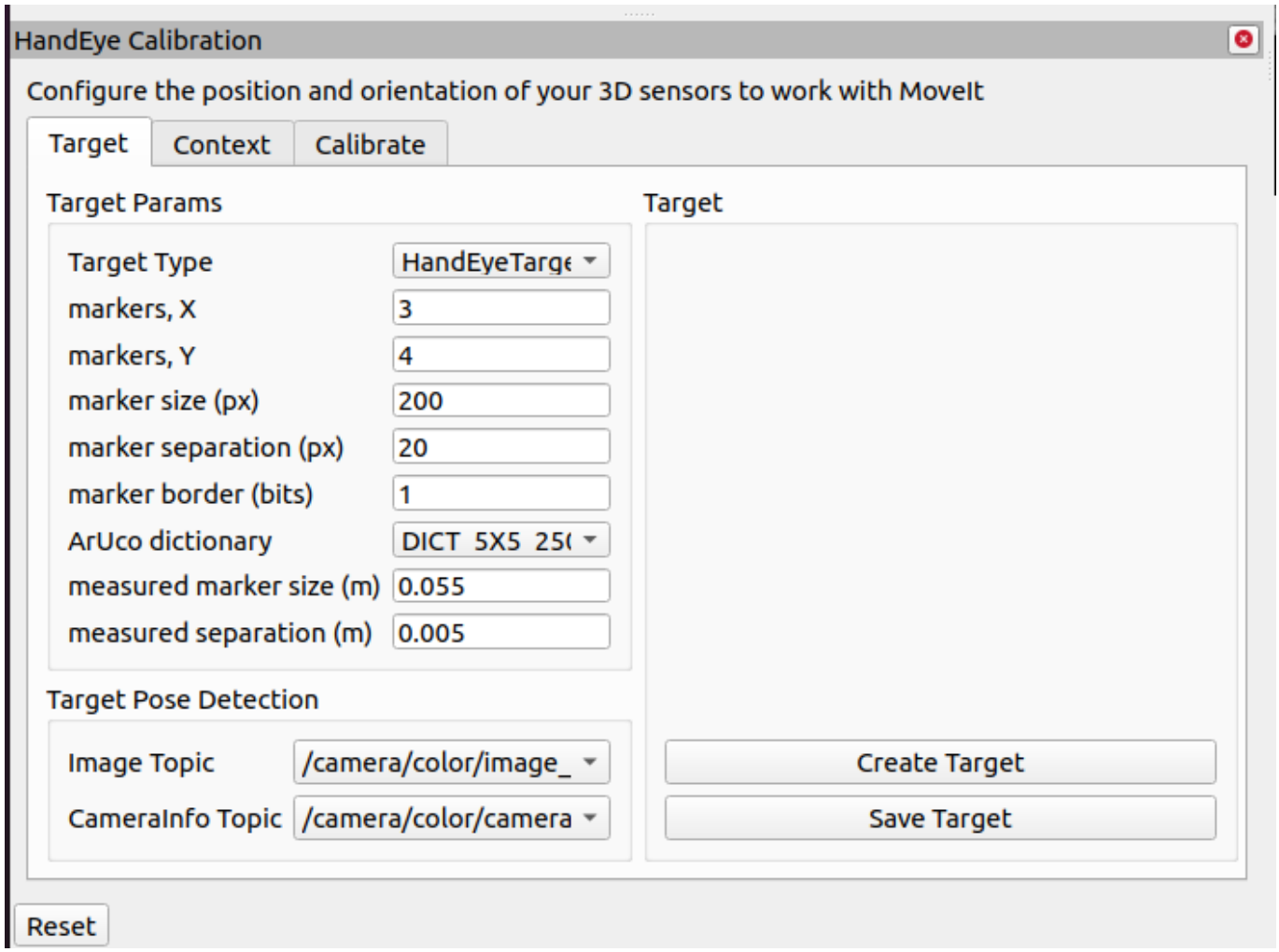
The steps:

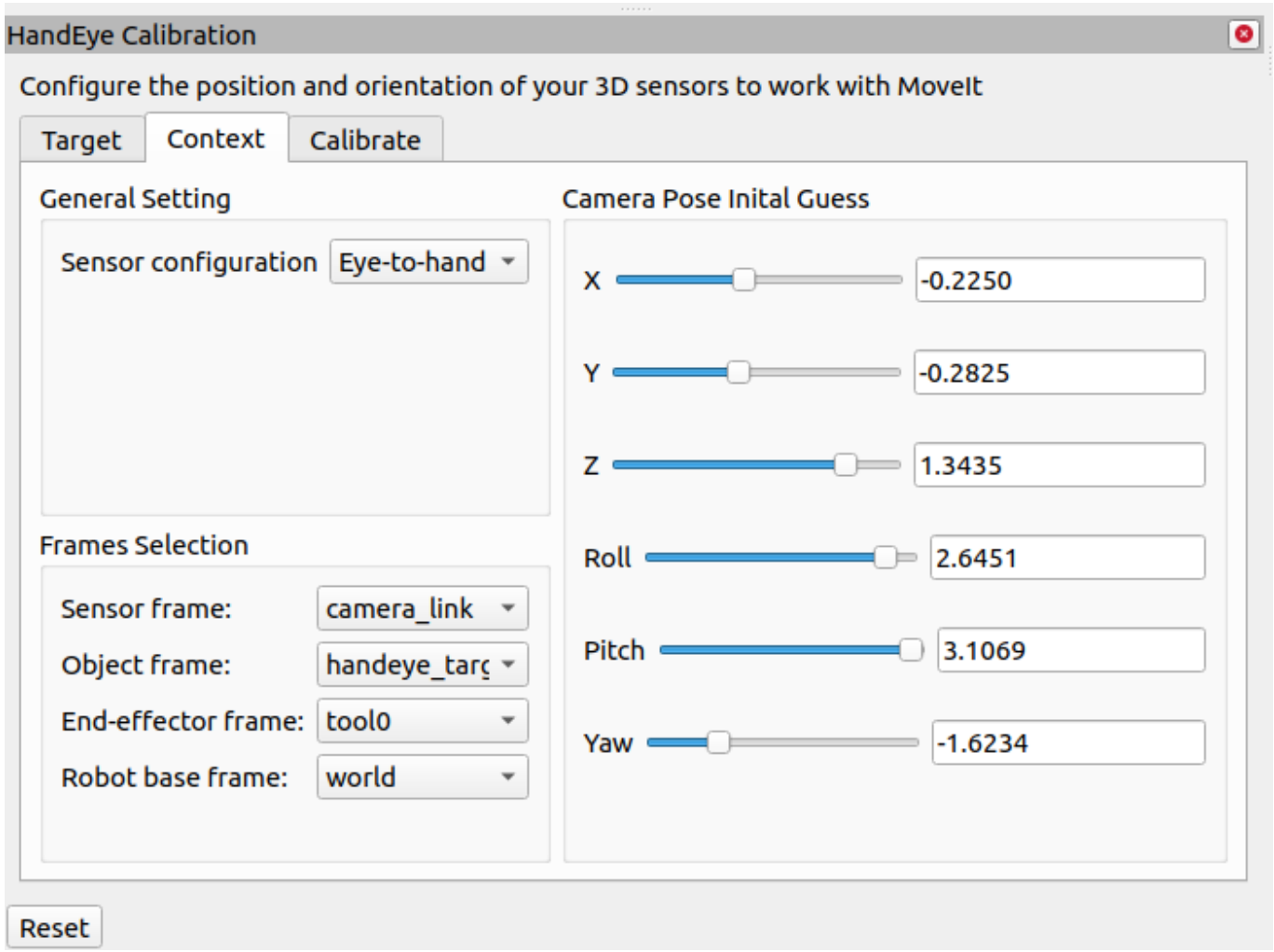
Since camera attached to hand (or eye-in-hand) has been explained in these instructions, I'm going to show the eye-to-hand case:

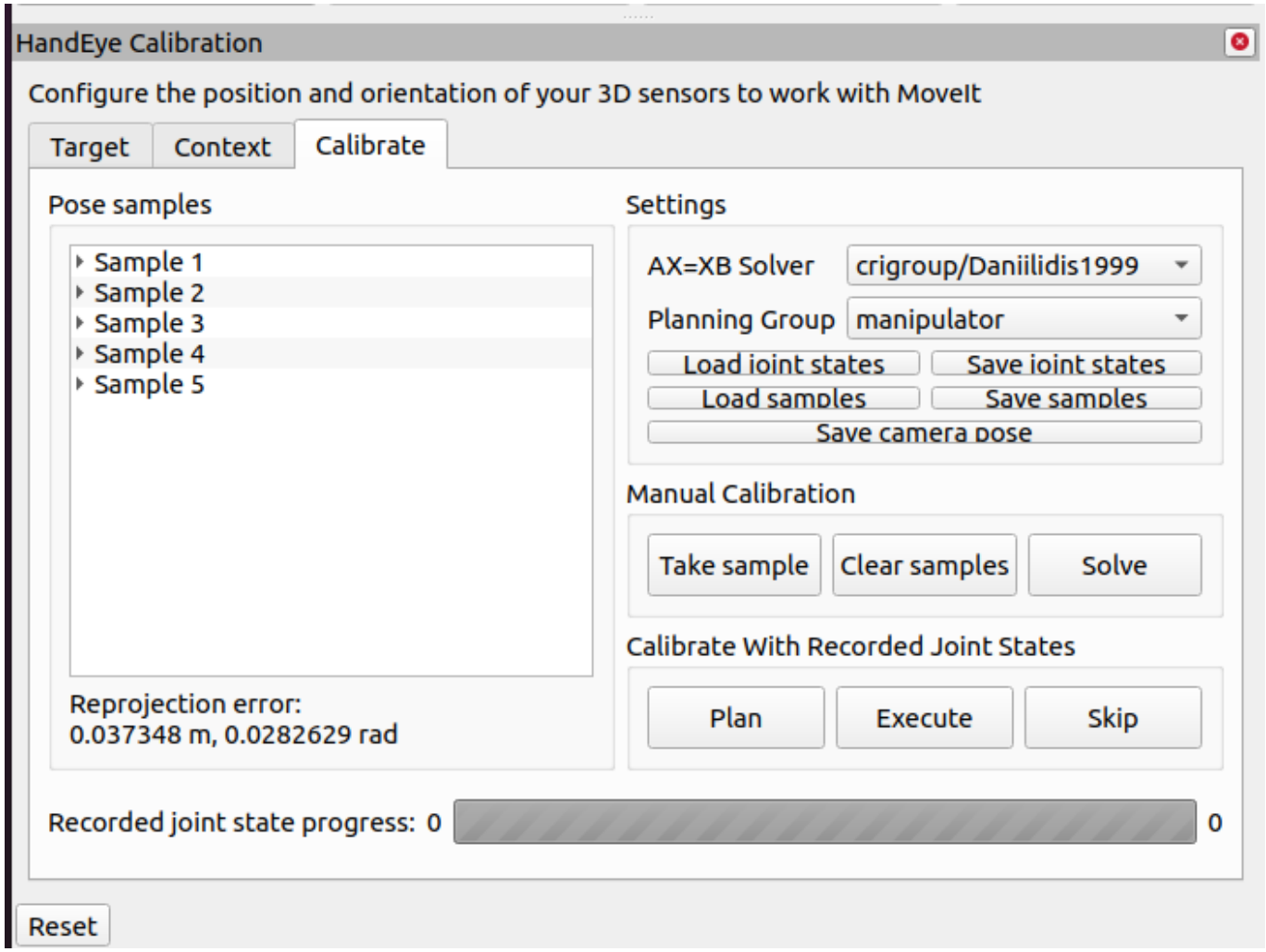




To get the plugin up and running, the default ros instructions will work. But follow the below settings(tested and verified):


HandEye Calibration
Configure the position and orientation of your 3D sensors to work with MoveIt
Target
Context
Calibrate
Target Params
Target Type
HandEyeTarge
markers, X
3
markers, Y
4
marker size (px)
200
marker separation (px)
20
marker border (bits)
1
ArUco dictionary
DICT_5X5_25(
measured marker size (m)
0.055
measured separation (m)
0.005
Target Pose Detection
Image Topic
/camera/color/image_
CameraInfo Topic
/camera/color/camera
Target
Create Target
Save Target
Reset


HandEye Calibration
Configure the position and orientation of your 3D sensors to work with MoveIt
Target
Context
Calibrate
General Setting
Sensor configuration
Eye-to-hand
Frames Selection
Sensor frame:
camera_link
Object frame:
handeye_targ
End-effector frame:
tool0
Robot base frame:
world
Camera Pose Inital Guess
X
-0.2250
Y
-0.2825
Z
1.3435
Roll
2.6451
Pitch
3.1069
Yaw
-1.6234
Reset



Once you take 5 samples of the marker, the plugin should automatically print the camera transformation matrix to the terminal like this:

from base frame 'world'to sensor frame 'camera_link'

Reprojection error:

0.037348 m, 0.0282629 rad

[INFO] [1658327615.156458109]: Publish camera transformation

0.0525243 -0.998017 0.0346943 -0.224953

0.87717 0.0627139 0.476067 -0.282496

-0.477299 0.00542768 0.878724 1.34355

0 0 0 1

Then you can use an online tool like this one to change from rotation matrix to roll-pitch-yaw angles and the first 3 values of the last column denote the x-y-z translations.