### NIST 3D Person Tracking Camera Calibration Instructions

# Intrinsic Calibration

First, the stereo cameras must be calibrated using the checkerboard method. Checkerboard calibration of the ASUS sensor may also improve results, however we use the default calibration values for this device because they are known to be pretty good.

The purpose of intrinsic calibration is to produce a “dot ini” file for both the left and right cameras as well as the relative transformation between the two. These file contains information regarding the focal lengths of the cameras and the lens distortion parameters.

To produce “ini” files:

1. Make a copy of the current ini files in a new directory. The current one is located in the aravis\_camera\_driver/launch/3-28-2013. Usually one would just create a new directory using today’s data, and copy the current “ini” files into the new directory.
2. Edit the file aravis\_camera\_driver/launch/nist\_raw\_cameras.launch to point toward the new directory. The line you want to edit follows:

<param name="camera\_ini\_dir" value="$(find aravis\_camera\_driver)/launch/3-25-2013"/>

1. Next launch the stereo data using:

roslaunch aravis\_camera\_driver nist\_stereo\_data.launch

1. Then, get a checkerboard, measure and count its internal squares. Ours is 8 internal squares by 11 internal squares which are each 90mm.
2. Launch the stereo calibration routine. Which will ultimately modify the “ini” files. Do not destroy the original ones by failing to edit nist\_raw\_cameras.launch file before running the stereo calibration routine.
3. Move the Checkerboard around in front of the cameras until about 100 different images have been saved. Be sure to use the entire field of view, and tilt the checkerboard to various angles at various distances.
4. Click on the “Calibrate” button on the stereo calibration gui and wait for quite some time.
5. When it is done, click on the “Commit” button. The “ini” files should be written.
6. Because the nist\_raw\_cameras.launch file is used by all of the other launch files, the new calibration will now be used for all system launch files.

# Extrinsic Calibration

In order to know where the camera is relative to a global frame, one must perform an extrinsic calibration. This procedure involved several steps. First, fiducial targets must be printed and placed within the field of view of the cameras. These special fiducials were developed for automatic recognition using the ar-toolkit. Each target is unique. In the ar\_pose package, there is a list of patterns, and associated gif images. Simply select a set of targets, print them, and place them where they can be seen. The ar-toolkit automatically locates the target using a single camera. It does not use stereo. However, it uses the known size of the target to determine the distance and orientation of the fiducial from the camera. One is free to enlarge or reduce the printed target, however the exact size of the target must be determined for the range information to be correct.

The second step is to edit the text file “object\_4x4.” This file contains a list of all the targets, and their actual sizes.

Here is an example of one entry associated with fiducial pattern 1 and image file 4x4\_384\_80.gif which is associated with pattern file data/4x4/4x4\_80.patt.

#pattern 1

4x4\_1

data/4x4/4x4\_80.patt

215.9

0.0 0.0

Currently, we are using only one pattern which was 4x4\_80.patt which was printed, then enlarged using a photocopier to be 215.9mm on each side.

Next, one simply runs the extrinsic calibration launch file for the desired camera. There are three options:

1. ExCal\_B21.launch (For Basler\_21280021)
2. ExCal\_B24.launch (For Basler\_21280024)
3. ExCal\_Kinect.launch (For the Asus color camera)

These scripts bring up rvis. However, by default the camera frame is set to the Kinect device, and must be changed to the stereo frame when these cameras are being calibrated.

When the frame is set correctly, all the targets visible to the camera should be shown relative to the camera in rviz. Once this is verified, one may export the relative locations to a file by setting a ros parameter. Enter the following to export the data:

rosparam set /ar\_pose/Output\_marker\_xyz true

These launch files all rely on an environment variable being set to determine where they deposit their data. This environment variable is NIST\_EXCAL. The three launch files save the xyz data for each marker to a text file named either:

1. B21.txt
2. B24.txt
3. Kinect.txt

The xyz data is ordered as the markers appear in “object\_4x4” file.

The next step is to measure each markers location in the global frame. This may be accomplished using an IGSP system or even a tape measure.

Finally, one needs to compute the transformation between the coordinate frame of the camera and the world coordinate fame. Ideally, one would compute the optimal transform by minimizing the sum of squared reprojection error. However, one may compute the transform which minimizes the sum of squared Euclidian distance between each corresponding point using the matlab/octave scrip absOrient.m. To do so, simply load both files and call the function.

* P1 = load(‘B21.txt’);
* P2 = load(‘IGPS\_cal\_data.txt’);
* absOrient(P1’,P2’);

Note, that the script computes only the orientation, not the translation vector. The translation is simply the vector between the centers of mass of each collection of points. Or, one may use the translation for any one of the points provided by ar-toolkit.