# READING OPTITRACK DATA WITH MATLAB

ERIC CRISTOFALO (EMC73@BU.EDU)
MULTI-ROBOT SYSTEMS LAB
BOSTON UNIVERSITY
CREATED: 2014/07/21
MODIFED: 2016/05/16

## 1. INTRODUCTION

This document serves as a tutorial for reading Optitrack data into MATLAB. The MATLAB code was developed by Dingjiang Zhou and was adapted by Eric Cristofalo; it is available for download here https://drive.google.com/folderview?id=0B-oDd9I8HgnzUFhuVGxvaTR3c2c&usp=sharing .

The coordinate systems of the Optitrack system are detailed in Section 2, the Optitrack camera calibration procedure is documented in Section 3, and the general information for the MATLAB functions is in Section 4. This document assumes you are connected to the same network as the Optitrack computer (BU (802.1x) or TRENDnet652).

Please note that there are two ways to use this code, the first as MATLAB only .m files and the second using a C program compiled in MATLAB as a MEX file. The MEX file can only be used on Linux operating systems at the moment, but will run this section of the program approximately 40x faster than the MATLAB only equivalent.

The reading optitrack into MATLAB download includes the following items:

- OptitrackToMATLAB_UserDocumentation.pdf
- readOptitrackExample.m
- optitrackSetup.m
- readOptitrack.m
- parseNatNet.m
- quatrn2rot.m
- rot2ZYXeuler.m
- parseNatNetMex.c (required for linux only)

## 2. COORDINATE SYSTEMS

There are many options for setting up a coordinate system in the Optitrack environment; two useful ones will be described here before beginning the tutorial.

The ‘Optitrack’ coordinate system is generated using the calibration square in the calibration process (Section 3). This coordinate system is an x-z ground plane frame and the raw data (x, y, and z position and quaternions) will be given in this frame. Optitrack provides quaternions with respect to YZX Euler angles, so some transformation is required to obtain the usual ZYX (yaw, pitch, roll) Euler angles we are familiar with. This coordinate system has the origin at the center of the calibration square (Figure 1).

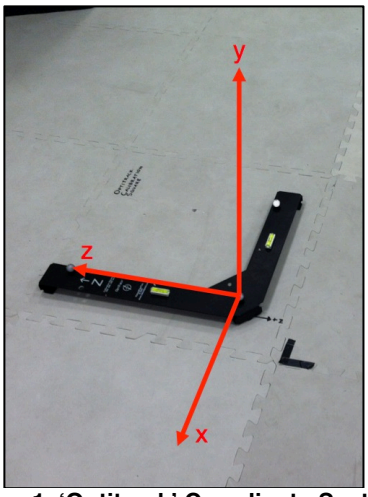



**Figure 1. 'Optitrack' Coordinate System.**

The 'XY Plane' coordinate system is calculated by transforming the raw Optitrack data. This frame uses an x-y ground plane due to the convenience for ground robotics. It has Euler angles denoted in the traditional fashion (yaw, pitch, roll) and the origin is located at the center of the calibration square (Figure 2).

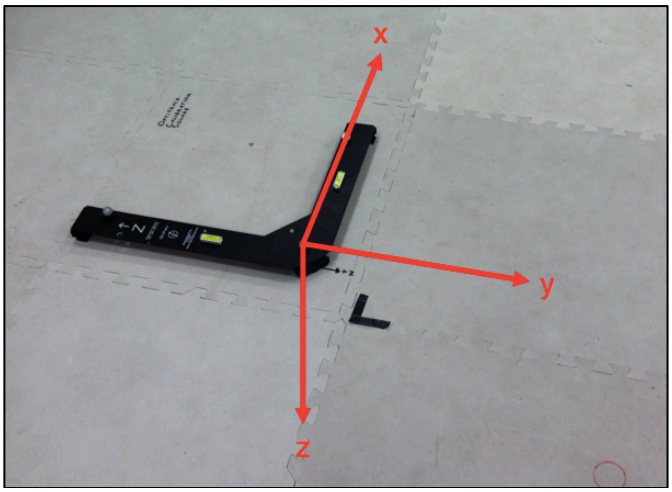



**Figure 2. 'XY Plane' Coordinate System.**

# 3. Optitrack Calibration

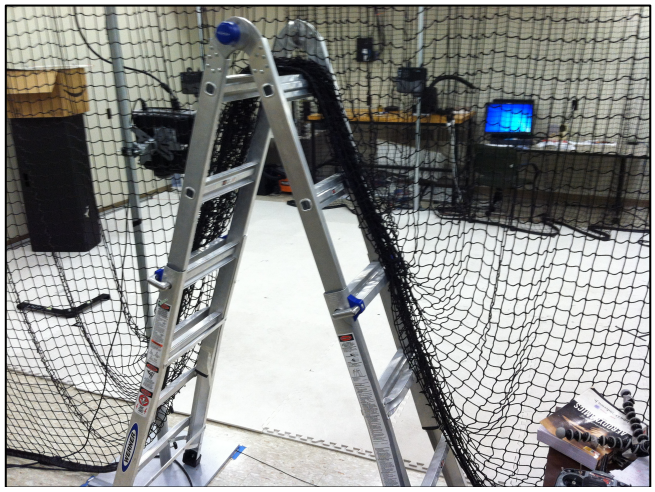
Optitrack camera calibration should be completed when the position estimation error has increased, Optitrack cameras have been moved, or when approximately one month has passed since the last calibration. You can also use calibration data to simply make new 'projects' in the Motive software, I recommend every individual using Optitrack has their own project. Optitrack uses three specific file formats to save the wanding timeline data (.tim), the calibration files (.cal), and the projects themselves (.ttp). Creating new calibrations and making new projects using calibration data will be detailed here.

## New Calibration

To perform a new camera calibration:

1. Use the Optitrack Computer (Windows 7) near the robotic workspace
    a. Username: Optitrack
    b. Password: optitrack

2. Open **Motive** from the desktop
3. Select *Perform Camera Calibration* from the pop-up menu
4. Remove all Optitrack reflective markers or shiny objects from the workspace. You can verify nothing is being picked up by the system by checking the individual camera views that will be displayed on screen.
5. Select *Block Markers*. This hides any existing reflective objects in the workspace.
6. Select *Start Wanding*
7. Use calibration wand
    a. Remove the three marker calibration wand from the wall
    b. Tighten all the markers onto the wand for greater accuracy
8. Move the wand around the entire workspace (close to the floor, net and corners of room) making sure to never hit or move the markers on the wand (this will decrease the calibration accuracy)
    a. The goal is to obtain 6000-10000 points in each camera view (the more the better).
    b. Some cameras will be stubborn for acquiring points, use individual camera view focus on calibrating each individual camera.
9. When finished wanding, select *Apply Results*
10. Select *Apply* to save the calibration
11. Place the calibration square in the calibration tile where you want the origin of the workspace to be. The default position of the calibration tile is shown below.



12. Select *Set Ground Frame*
    a. Default offset = 55 mm
    b. Offset when using the calibration tile = 26.75 mm (because the calibration square is lower than normal)
    c. The coordinate system is now the ‘Optitrack’ frame described in Section 2.
13. Select *Save Ground Frame*
14. All set, simply save this project (using the save icon, or *file -> save project*) with a unique project name to use it in for future use!

## New Project Using Existing Calibration

To create a new project using an existing calibration:

1. Use the Optitrack Computer (Windows 7) near the robotic workspace
    a. Username: Optitrack
    b. Password: optitrack
2. Open **Motive** from the desktop
3. Select *Open Camera Calibration* from the pop-up menu

4. Place the calibration square in the calibration tile where you want the origin of the workspace to be.
5. Select *Set Ground Frame*
    a. Default offset = 55 mm
    b. Offset when using the calibration tile = 26.75 mm (because the calibration square is lower than normal)
    c. The coordinate system is now the 'Optitrack' frame described in Section 2.
6. Select *Save Ground Frame*
7. Save the project, *file -> save project* with your unique project name.

# 4. READING OPTITRACK DATA

Once Optitrack is calibrated and you have a project.ttp file, the Optitrack data for any number of robots can be sent to MATLAB in the following manner:

1. Use the Optitrack Computer (Windows 7) near the robotic workspace
    a. Username: Optitrack
    b. Password: optitrack
2. Open **Motive** from the desktop
3. Select *Open Existing Project* from the pop-up menu and select your .ttp file
4. Place the robot in the workspace
    a. The position and orientation you place the robot will be the zero position for all Euler angles.
5. In **Motive**, click and select the observable markers that make up the robot.
6. *Right click*, and select *rigid body – create from selected markers*
7. If there are multiple robots, place them in the workspace and create rigid bodies for them as well
8. Go to *view -> rigid body properties* to open the rigid body tab which will contain information about any selected rigid body.
9. Under *advanced -> dynamic constraints,* you can set a low pass filter on the output to reject measurements that are deviate significantly from the past measurements.
10. To see a video image, right click on the display of a camera view in the bottom of the screen, then select video type-MJPEG Mode
    a. For calibrating and tracking, the cameras should always be in **precision mode**
11. To send data, go to *View -> Data Streaming*
12. Check the box that indicates *Broadcast Frame Data*
13. Select the following parameters:
    a. stream markers - true
    b. stream rigid bodies- true
    c. type - multicast
    d. command port - 1510
    e. data port - 1511
    f. local interface - 192.168.0.1
    g. multicast interface - 239.255.42.99
14. Save the .ttp project

Other Camera Options:
Under the cameras tab, the camera settings are available for adjustment: frames per second, exposure, threshold, and LED brightness. The LED adjusts the brightness of the LEDs on each camera. If the

software is picking up reflections off miscellaneous surfaces, this can be turned down. The frames per second (FPS) entry is typically set to 120.

## MATLAB (ALL OPERATING SYSTEMS)

For using the MATLAB only option, download the package contents and add this folder to your MALTAB path. Alternatively, you can work out of this folder's directory. Open readOptitrackExample.m and simply run the program. Optitrack 6DOF data should be output in the command window. If it does not seem to be working and yields no errors, make sure a rigid body has been created in Motive and the data is streaming.

The function readOptitrack.m is used to read the Optitrack raw data and output the pose of the rigid bodies in either the 'Optitrack' or 'XY Plane' coordinate system (which can be specified from the input). The output structure, opti, contains all rigid body information. Raw optitrack data (with quaternions) is under opti.rigidBodies.SE3. Useful optitrack data (position and Euler angles in specified frame) is under opti.pose. This opti.pose matrix is a 6xn array in the following format:

| | Rigid Body 1 | Rigid Body 2 | … | Rigid Body n |
|---|---|---|---|---|
| X Position (m) | 1.456 | 1.633 | | 1.612 |
| Y Position (m) | 0.642 | 0.846 | | 0.256 |
| Z Position (m) | -0.011 | -0.011 | | -0.011 |
| Roll Angle (rad) | 0.001 | 0.001 | | 0.004 |
| Pitch Angle (rad) | 0.000 | -0.001 | | 0.001 |
| Yaw Angle (rad) | 0.002 | 0.740 | | 0.001 |

Note: the z-position is always negative (upwards) and the position denote the position of the markers, NOT necessarily the position of the robot.

## MATLAB/C (LINUX ONLY)

When working on Linux, use the provided parseNatNetMex.c program to run parseNatNet.m ~40x faster. To compile this C program, follow the following instructions:

1. Ensure you have an accepted compiler installed on your computer by typing the following into the MATLAB command window
   ```
   mex –setup
   ```
2. If you do, compile the parseNatNetMex.c program in MATLAB
   ```
   mex parseNatNetMex.c
   ```
3. Uncomment the following lines in the MATLAB functions:
   a. Line 52 in **optitrackSetup.m** (comment line 51)
   b. Line 53 in **readOptitrack.m** (comment line 52)