LMI TECHNOLOGIES

+1 604 636 1011
+1 604 516 8368
info@lmi3D.com
www.lmi3D.com
1673 Cliveden Avenue
Delta BC V3M 6V5
Canada

Title: Hand-Eye Calibration for 3D Robotic Systems Using Gocator
Revision 1.1

Table of Contents

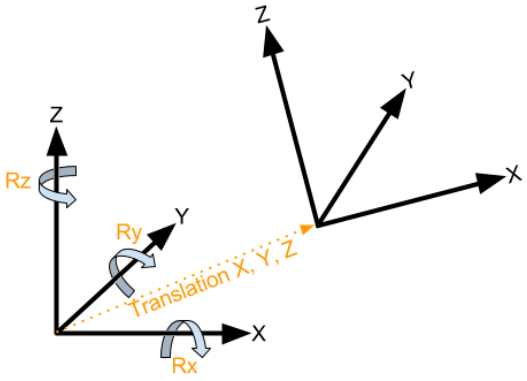
+1 604 636 1011
+1 604 516 8368
info@lmi3D.com
www.lmi3D.com
1673 Cliveden Avenue
Delta BC V3M 6V5
Canada

1 Overview

Gocator is factory pre-calibrated, meaning you can get accurate measurements or point clouds in sensor coordinates out of box. But in many robotic applications using Gocator, the sensor is mounted on a robot flange (end effector) to extend Gocator's reachability and field of view, for example, to perform inspection or pick and place. In most of these applications, the results from Gocator have to be transformed from sensor coordinates to other coordinates (i.e., robot base coordinates), and sent to a robot controller to perform certain tasks. To achieve that, you must perform hand-eye calibration (tool calibration) of the Gocator to the robot flange.

2 Coordinate System and Rigid Transformation

A rigid transformation of coordinates can be represented by 6 parameters: translations (x, y, z) and rotations (Rx, Ry, Rz). Hand-eye calibration of a Gocator sensor to the robot flange finds the transformation (x y, z, Rx, Ry, Rz) from sensor coordinates to robot flange coordinates. We will denote this transformation as X.



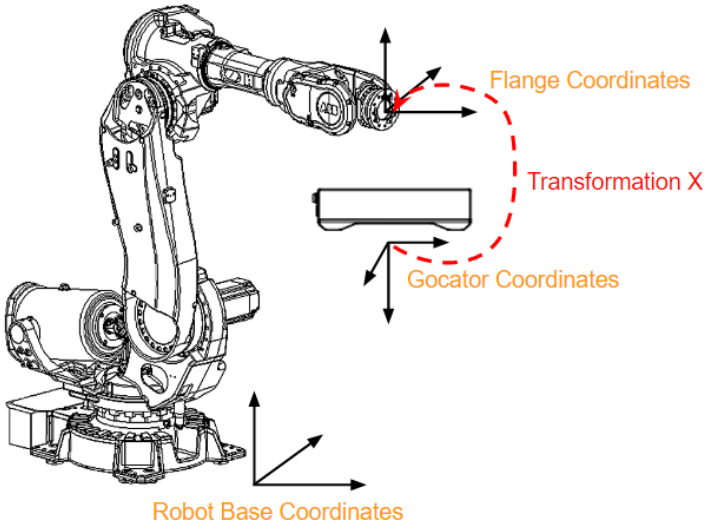
After obtaining X, you can transform any measurement "m" from Gocator in sensor coordinates to robot base coordinates (or part coordinates) by using the following formula:

$$m' = [T] \cdot [X] \cdot m$$

Where "m'" is the measurement in robot base coordinates (or part coordinates); T is the transformation from the flange coordinates to the base coordinates (or part coordinates), which can be obtained from the robot controller.

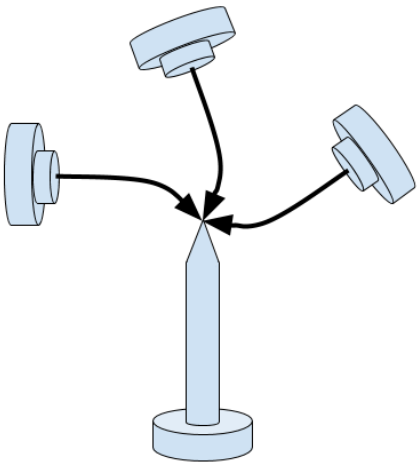
NOTE	[T] is the matrix representation of the transformation parameters (x,y,z, Rx, Ry, Rz).

+1 604 636 1011
+1 604 516 8368
info@lmi3D.com
www.lmi3D.com
1673 Cliveden Avenue
Delta BC V3M 6V5
Canada



3 Calibration Using Built-in Procedures

Many robot controllers use built-in procedures, such as teaching TCP (tool center point), to perform tool calibration, which can also be used for Gocator as a tool. These procedures typically calculate X translations by using a single point constraint with multiple robot poses and X rotations by moving the tool along its coordinate axes.



The typical TCP teaching procedure:

1. Place a target (for example, a pin) in front of the robot.
   The target remains stationary during the procedure.
2. Jog the robot in multiple (at least 3) poses while the TCP remains on the target, sending the position to the robot controller.
3. Move the TCP along its axes (X, Y, Z) and send the positions to the robot controller (only required if rotations of the tool are needed).
4. Click on “Calculate” in the controller.
   The result of the transformation should be calculated by the built-in algorithm of the controller.

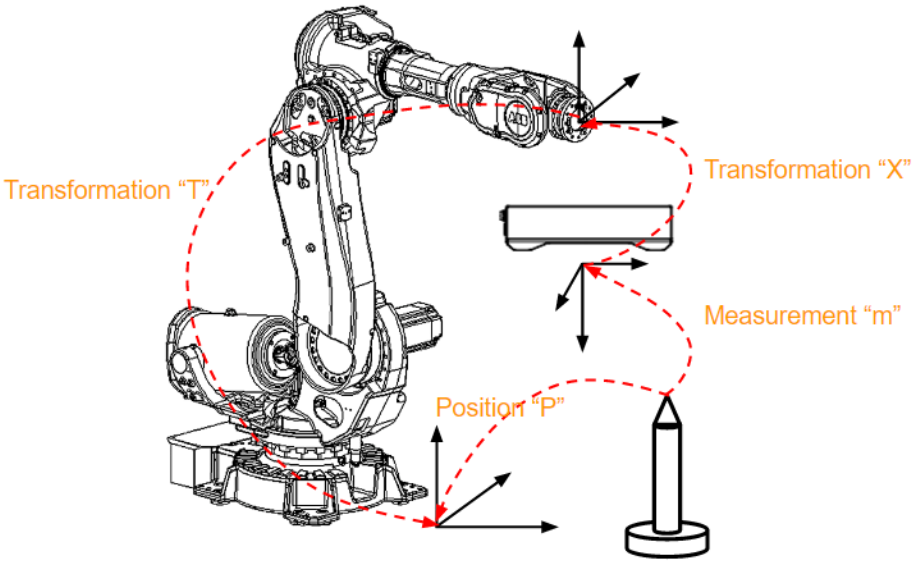
+1 604 636 1011
+1 604 516 8368
info@lmi3D.com
www.lmi3D.com
1673 Cliveden Avenue
Delta BC V3M 6V5
Canada

With Gocator, we can implement the procedure above by measuring a feature on a target (hole, corner or sphere, etc.), and recording all the positons by placing the sensor origin on the target (for example, measuring the center of a sphere = (0, 0, 0)).

Implementing the procedure above is very straightforward and no development is required. However, it is very difficult to align the sensor origin to the target accurately. And because all steps are manual, it is very time-consuming if the user needs to set up the calibration procedure multiple times.

4 Calibration by User-Defined Procedures

To improve the accuracy of alignment between the sensor origin and the target, and to reduce the time required to perform the calibration, you can automate the process by implementing your own calculation functions. The basic idea is the same as above, that is, building up equations by using geometric (point, line, plane, etc.) constraints. Then solve the equations to get the X. For example, by using a point constraint, you can obtain the following equations by measuring a static target:



$$P_1 = [T_1] \cdot [X] \cdot m_1;$$

$$P_2 = [T_2] \cdot [X] \cdot m_2;$$

.....

$$P_n = [T_n] \cdot [X] \cdot m_n$$

Where P_i is the position of the target in robot base coordinates. Since the target is static for all the measurements, the following is true: $P_1 = P_2 = \ldots = P_n$. So we can build up a system of linear equations of $[T_i] \cdot [X] \cdot m_i = [T_j] \cdot [X] \cdot m_j$, which can be formulated as:

$$A \cdot X = b$$

Where A is the derivative matrix of X, and b is the residuals of the target measurements ($P_i - P_j$). Then X can be calculated by linear least equation algorithm.

+1 604 636 1011
+1 604 516 8368
info@lmi3D.com
www.lmi3D.com
1673 Cliveden Avenue
Delta BC V3M 6V5
Canada

NOTE	Gocator 3000 (G3) series sensors use the left-handed coordinate system, but most robot controllers use the right-handed coordinate system. You can change the G3's coordinates to right-handed by flipping X values or Y values after retrieving the values from the sensor.