Readme for Gomotion ROS Package

0/0/0000 0:00:00 AM

ROSGomotion.docx

This document presents a Robot Operating System (ROS) package for gomotion robot joint and Cartesian pose trajectory generation.

# Gomotion Types

|  |  |
| --- | --- |
| Typedefs | |
| typedef int | [**go\_result**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d44\gotypes_8h.html#a55d48b38cd959f63c7e8db8337a9792a) |
|  | |
| typedef double | [**go\_real**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d44\gotypes_8h.html#afd666a2393eebd71ee455846ac9def9b) |
|  | |
| typedef int | [**go\_integer**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d44\gotypes_8h.html#a7d30f606bb0f58ffe2b3bd71e5c8af5c) |
|  | |
| typedef unsigned char | [**go\_flag**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d44\gotypes_8h.html#ae890d9a0ddecc0d3073622cc4312092d) |
|  | |

|  |  |
| --- | --- |
| Enumerations | |
| enum | {    [**GO\_RESULT\_OK**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d44\gotypes_8h.html#a726ca809ffd3d67ab4b8476646f26635a474a770c6dd01e6ff6d9316bec185bdc) = 0,  [**GO\_RESULT\_IGNORED**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d44\gotypes_8h.html#a726ca809ffd3d67ab4b8476646f26635a2ce6982cf7db77b01c9757c92b8850ac),  [**GO\_RESULT\_BAD\_ARGS**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d44\gotypes_8h.html#a726ca809ffd3d67ab4b8476646f26635a774dfa8abb1bab8ff8599bbcacefa13e),  [**GO\_RESULT\_RANGE\_ERROR**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d44\gotypes_8h.html#a726ca809ffd3d67ab4b8476646f26635adbdc03b02695760e5650a9c03dba3d0e),    [**GO\_RESULT\_DOMAIN\_ERROR**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d44\gotypes_8h.html#a726ca809ffd3d67ab4b8476646f26635a59be6a201b1e7b8905a0c89eec00963b),  [**GO\_RESULT\_ERROR**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d44\gotypes_8h.html#a726ca809ffd3d67ab4b8476646f26635acbf54540acb86538b3f72b4b8e0a176d),  [**GO\_RESULT\_IMPL\_ERROR**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d44\gotypes_8h.html#a726ca809ffd3d67ab4b8476646f26635a9d6bb28a1830d96c3dc0c0154153d80a),  [**GO\_RESULT\_NORM\_ERROR**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d44\gotypes_8h.html#a726ca809ffd3d67ab4b8476646f26635a03cf3256a6fc44ee40192ed232d4c659),    [**GO\_RESULT\_DIV\_ERROR**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d44\gotypes_8h.html#a726ca809ffd3d67ab4b8476646f26635ac2434e23b7ae651aebe2452af7af1995),  [**GO\_RESULT\_SINGULAR**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d44\gotypes_8h.html#a726ca809ffd3d67ab4b8476646f26635a0f2b551fe19dd46697296b56b83ef1d7),  [**GO\_RESULT\_NO\_SPACE**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d44\gotypes_8h.html#a726ca809ffd3d67ab4b8476646f26635ad1cc08529bcdd199480d3d57e53b5d84),  [**GO\_RESULT\_EMPTY**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d44\gotypes_8h.html#a726ca809ffd3d67ab4b8476646f26635a24838323a80d8fd8a140c92473f740e3),   [**GO\_RESULT\_BUG**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d44\gotypes_8h.html#a726ca809ffd3d67ab4b8476646f26635a5524763e925f0b14dcba6a321574bba7)  } |
|  | |
| enum | { [**GO\_QUANTITY\_NONE**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d44\gotypes_8h.html#a0411cd49bb5b71852cecd93bcbf0ca2daaf19b8b96ec24aee3e4bc35efc855a24) = 0,  [**GO\_QUANTITY\_LENGTH**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d44\gotypes_8h.html#a0411cd49bb5b71852cecd93bcbf0ca2da847a795c80c454fb18dd2843b9a5ae37),  **[GO\_QUANTITY\_ANGLE](file:///X:\\src\\github\\johnmichaloski\\ROS\\nistfanuc_ws\\src\\gomotion\\doc\\doxygen\\html\\d7\\d44\\gotypes_8h.html" \l "a0411cd49bb5b71852cecd93bcbf0ca2da70505768da54108916ca9d850a345c16)** } |

# Go Math Representations

Positions are vectors that indicate where something is. In the three-dimensional world, three numbers are necessary to indicate position. Go Motion supports position vectors in several representations: Cartesian, cylindrical and spherical.

The Cartesian representation uses three numbers **x**, **y** and **z** to represent distances from the origin along three perpendicular axes. The cylindrical representation uses three numbers **r**, **theta**and **z** to represent radial distance away from the origin, angle around the origin and distance up and down from the origin respectively. The spherical representation uses three numbers **theta**,**phi** and **r** to represent angle down from the zenith, angle around the origin and radius from the origin respectively. Go Motion has functions that convert position in one representation position in another representation, so the choice of which representation to use can be made for convenience. Cartesian representations will be assumed unless otherwise specified.

Orientations are vectors that indicate how something is rotated. In the three-dimensional world, three numbers are necessary to indicate orientation. Go Motion supports orientation vectors in several representations: roll, pitch and yaw; Euler angles; quaternions; rotation vectors and rotation matrices. Some of these representations use more than three numbers, exploiting redundancy to make calculations with these representations more efficient. For example, a quaternion uses four numbers, and a rotation matrix uses nine numbers.

Vectors are usually written as a column of numbers enclosed in vertical bars, like this:

| 1 |   
| 2 | - a vector depicted in its column form   
| 3 |

This can be unwieldy in text documentation, so vectors may also be written as a row of numbers enclosed in parentheses, like this:

( 1 2 3 ) - a vector depicted in its row form

The interpretation of a vector depends on the quantity it represents. The vector shown above could mean a translation of 1, 2 and 3 units in the x, y and z directions if the vector were a Cartesian position, or a rotation of 1, 2 and 3 units around the x, y and z directions if the vector were an orientation in roll, pitch and yaw.

Both position and orientation are needed to fully describe where something is and how it is rotated. The combination of position and orientation is called a 'pose'. Poses can be shown in row form like this pose representing a Cartesian position of ( 1 2 3 ) and an orientation in roll, pitch and yaw of ( 30 -30 90 ):

( 1 2 3 ; 30 -30 90)

A semicolon is used to separate the position from the orientation.

**Go Math Reference Frames**

Regardless of the representation chosen, the numbers that indicate position and orientation of an object depend on the established origin. Several origins may be established for convenience, for example one fixed on the world and one that moves with a tool. These origins may differ from each other in both position and orientation. The establishment of the position and orientation of an origin is a 'reference frame'. The term 'coordinate frame' is used interchangeably with 'reference frame'.

When several reference frames are being used, they are denoted as identifiers in braces, for example,

{A} - a reference frame called 'A'.

{world} - the world reference frame.

{tool} - the world reference frame.

To convert the representation of a pose in one reference frame to its representation in another, one needs to know the position and orientation of one origin with respect to the other. This difference between the two origins is called a 'transform'.

Poses and transforms are similar things; both include position and orientation. Whether something is a pose or a transform depends on how one is using it. Poses are used to indicate the position and orientation of things with respect to an established reference frame. Transforms are used to indicate the position and orientation of reference frames with respect to other reference frames. If a 'thing' happens to be a reference frame, its pose is its transform.

## **Go** Math Nomenclature

The letter **P** is used to denote positions and the letter **R** is used to denote orientations. As usual, trailing subscripts denote the identity of quantities, for example,

Phand - the position of the hand.

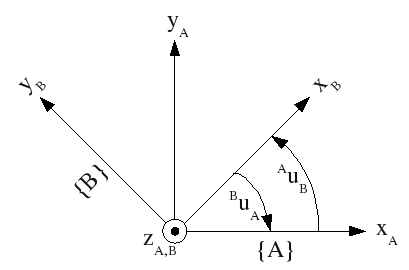
Rhead - the orientation of the head.

Leading superscripts denote the reference frame in which the quantity is expressed, for example,

APhand - the position of the hand with respect to the {A} reference frame.

BRhead - the orientation of the head with respect to the {B} reference frame.

The figure below shows a rotation of reference frame {B} with respect to reference frame {A} by an angle **u**. Note that the rotation can be viewed as a rotation of {B} with respect to {A}, denotedAuB, or as a rotation of {A} with respect to {B}, denoted BuA. The heads of arrows are attached to the 'of the' frames, while the tails of arrows are based on the 'with respect to' frames. Angles are taken as positive according the the right hand rule, so in this figure AuB is a positive number of about 45 degrees, while BuA is a negative number of the same magnitude.



*Figure* 1.

Transforms from one reference frame to another are denoted with leading subscripts and superscripts. The leading subscript denotes the original frame, and the leading superscript denotes the new frame. If a transform is purely rotation, it is denoted with an **R**, for example,

BAR - a rotation from the {A} frame to the {B} frame.

If a transform includes both a rotation and a translation, it is denoted with a **T**, for example,

worldtoolT - a transform from the {tool} frame to the {world} frame.

To convert a quantity from one frame to another, pre-multiply the quantity by the transform. Algebraically, the leading subscript of the transform must match the leading superscript of the quantity, for example,

BP = BAR \* AP

This is read, 'the position with respect to the B frame is the rotation from the A frame to the B frame times the position with respect to the A frame.' Another example is,

worldP = worldtoolT \* toolP

This is read, 'the position with respect to the world frame is the transform from the tool frame to the world frame times the position with respect to the tool frame.'

One can generate a rotational transform by defining an orientation vector with the appropriate angles. If using a roll, pitch and yaw representation, the rotational transforms in the figure above (each about the **z** axis) is a yaw:

|   0   |   
|   0   | = ARB   
| AuB |

|   0   |   
|   0   | = BRA   
| BuA |

and negative angles work as well, for

|    0   |   
|    0   | = ARB   
| -BuA |

## **Go** Math Examples

Given a reference frame {B} rotated 30 degrees with respect to the **z** axis of reference frame {A}, as in Figure 1 above, transform from points in the {B} frame to points in the {A} frame like this:

go\_rpy rot;

go\_cart pt\_in\_b;

go\_cart pt\_in\_a;

// The angle of {B} with respect to {A} is 30 degrees.

// Go uses angles in radians.

rot.r = 0, rot.p = 0, rot.y = [GO\_TO\_RAD](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d2c\gomath_8h.html#a257fd1bdbb4294c1bf1d2c34bfeaba42)(30);

pt\_in\_b.x = 1, pt\_in\_b.y = 2, pt\_in\_b.z = 3;

// Multiply a transform and a point to get a new point.

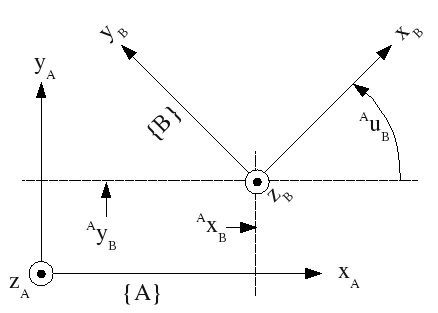
[go\_rpy\_cart\_mult](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d1\d0f\namespacegomotion.html#ac89b091b7958b0916f0bfffc89022751)(&rot, &pt\_in\_b, &pt\_in\_a);

go\_cart\_print(&pt\_in\_a);

will print the postion of the point in the {A} frame:

-0.133975 2.232051 3.000000

Here is a more complex example of a full transform, including both translation and rotation.



*Figure* 2.

The frame {B} is translated and rotated with respect to {A}. AxB is the amount of translation of {B} in the **x** direction of {A}. Likewise, AyB is the amount of translation of {B} in the **y** direction of {A}.AuB is the rotation of the {B} frame about the **z** axis of the {A} frame.

To convert points in the {B} frame to points in the {A} frame, do this:

go\_pose pose;

go\_rpy rot;

go\_cart pt\_in\_b;

go\_cart pt\_in\_a;

// The translation of {B} with respect to {A} is about (2 1 0).

pose.tran.x = 2, pose.tran.y = 1, pose.tran.z = 0;

// The angle of {B} wrt {A} is about 30 degrees, made into radians.

rot.r = 0, rot.p = 0, rot.y = [GO\_TO\_RAD](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d2c\gomath_8h.html#a257fd1bdbb4294c1bf1d2c34bfeaba42)(30);

// A 'go\_pose' uses quaternions for rotations, so we have to

// convert a roll-pitch-yaw to a quaternion.

[go\_rpy\_quat\_convert](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d1\d0f\namespacegomotion.html#a220f82039af859bfd9413107e7fbdf13)(&rot, &pose.rot);

pt\_in\_b.x = 1, pt\_in\_b.y = 2, pt\_in\_b.z = 3;

// Multiply a transform and a point to get a new point.

[go\_pose\_cart\_mult](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d1\d0f\namespacegomotion.html#a8fddcbebee69b64a2536721c33087e3c)(&pose, &pt\_in\_b, &pt\_in\_a);

go\_cart\_print(&pt\_in\_a);

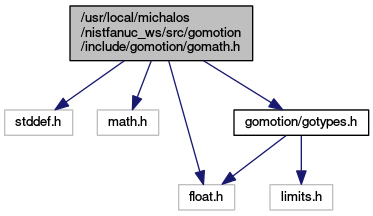
This gives the transformed point in the {A} frame as

1.866025 3.232051 3.000000

Declarations for pose math functions.

#include <stddef.h>  
#include <math.h>  
#include <float.h>  
#include "[**gomotion/gotypes.h**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d44\gotypes_8h_source.html)"

Include dependency graph for gomath.h:



**Trajectory Planning Algorithms**

The Go Motion trajectory planning algorithms are based on smooth velocity profiling with bounded speed, acceleration and jerk, called "constant jerk" or "S-curve" velocity profiling. This gives smoother control than "trapezoidal" velocity profiling, which transitions instantaneously between acceleration and no acceleration and incurs spikes in unbounded jerk.

Constant-jerk (CJ) profiling is shown in Figure 1, a plot of the speed versus time. There are 7 phases to the motion. Phase 1 is a jerk phase, where the acceleration varies smoothly from 0 at time 0 to *a1* at time *t1* following the jerk (change in acceleration per unit time) *j0*. Phase 2 is an acceleration phase, with constant acceleration *a1* throughout. Phase 3 is a jerk phase (or de-jerk phase) with constant (negative) jerk slowing down the acceleration from *a1* to 0. Phase 4 is a constant speed phase at speed *v3*. Phase 5 is a constant-jerk counterpart to phase 3, where the deceleration varies smoothly from 0 to *-a1*. Phase 6 is a constant-acceleration counterpart to phase 2. Phase 7 is a constant-jerk counterpart to phase 1, where the deceleration varies smoothly from *-a1* to 0 and motion stops.

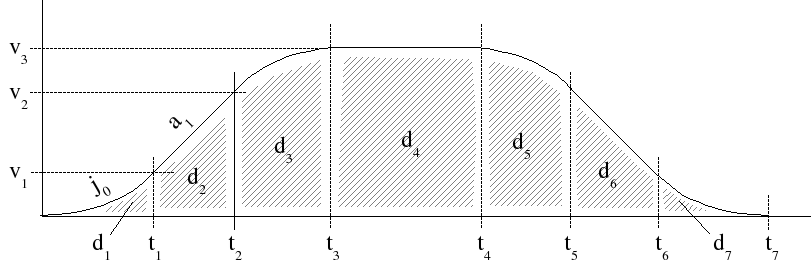
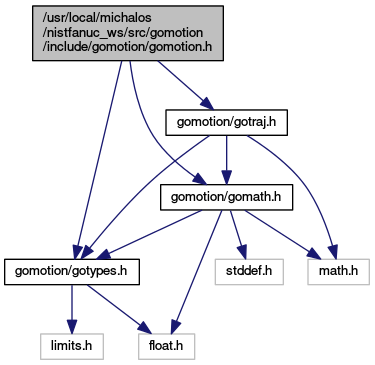


Figure 1. Constant jerk velocity profiling.

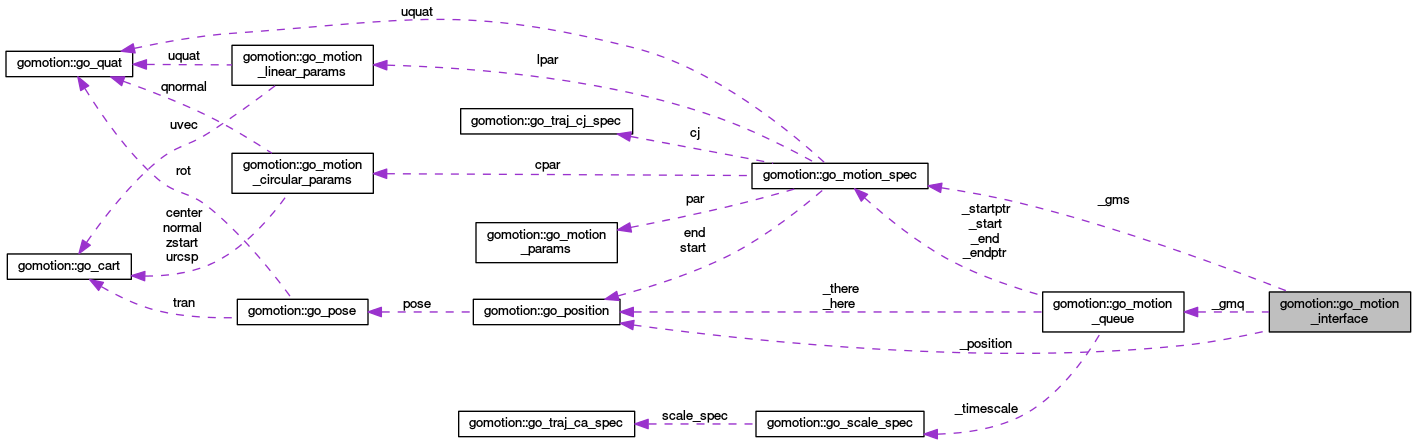
Declarations for motion queue manipulation. [More...](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d3\df0\gomotion_8h.html#details)

#include "[**gomotion/gotypes.h**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d44\gotypes_8h_source.html)"  
#include "[**gomotion/gomath.h**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\d2c\gomath_8h_source.html)"  
#include "[**gomotion/gotraj.h**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d7\dc0\gotraj_8h_source.html)"

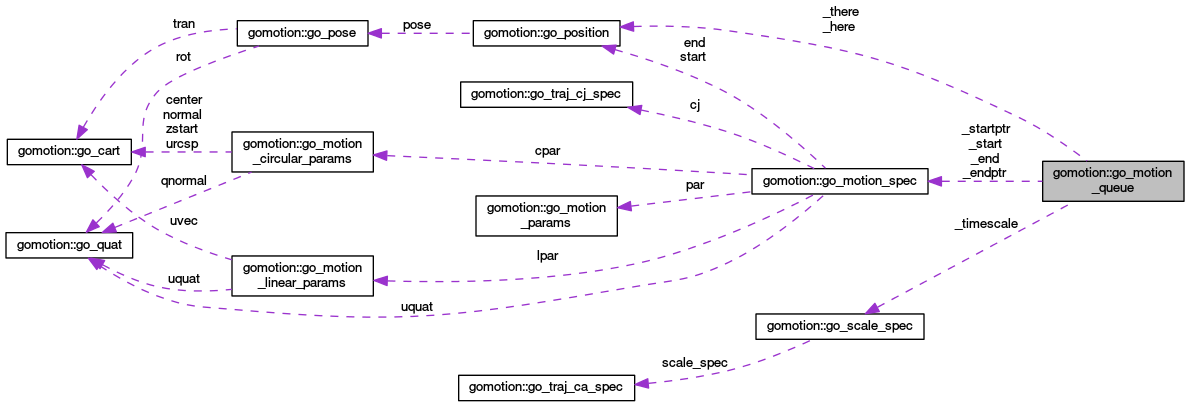
Include dependency graph for gomotion.h:



# Gomotion ROS Package



Collaboration diagram for gomotion::go\_motion\_queue:



#include <[**gomotion.h**](file:///X:\src\github\johnmichaloski\ROS\nistfanuc_ws\src\gomotion\doc\doxygen\html\d3\df0\gomotion_8h_source.html)>

Collaboration diagram for gomotion::go\_motion\_spec:

