PathPlannerLib

LabVIEW

Reference

Version 4.0.0

Table of Contents

Introduction

The PathPlanner LabVIEW library provides utility functions to read, create, and follow PathPlanner paths.

The library source code, package build specifications, and test package can be found here https://github.com/jsimpso81/PathPlannerLabVIEW

Function Help

Each VI includes help that can be accessed using the standard LabVIEW help toggle (Ctrl H).

TO DO YET

Function Examples

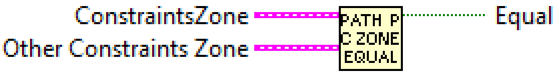
Many of the functions have examples that can be found under the LabVIEW "Find examples..." function. (Help -> Find Examples...). The function examples are easiest to find when "Directory Structure" is selected.

TO DO YET

Function Groups

ConstraintsZone

PathPlanner_ConstraintsZone_Equals



Compares two Constraints Zone definitions

Inputs:

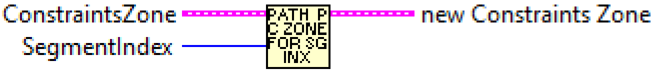
-- ConstraintsZone - cluster - first definition to compare

-- Other ConstraintsZone - cluster - other definition to compare

Outputs:

-- Equal - boolean - TRUE if equal.

PathPlanner_ConstraintsZone_ForSegmentIndex



Transform the positions of this zone for a given segment number.

For example, a zone from [1.5, 2.0] for the segment 1 will have the positions [0.5, 1.0]

Inputs:

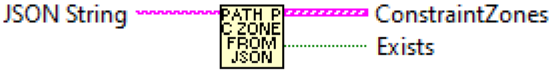
-- ConstraintsZone - cluster - definition of zone

-- segmentIndex - int - The segment index to transform positions for

Outputs:

-- NewConstraintsZone - cluster - The transformed zone

PathPlanner_ConstraintsZone_FromJSON



Create a constraints zone from json

Inputs:

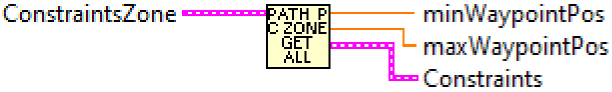
-- JsonString - string - String containing the JSON to parse.

Outputs:

-- ConstraintsZone - cluster - The constraints zone defined by the given json object

-- Exists -- boolean -- True if a constraints zone was found and parsed.

PathPlanner_ConstraintsZone_GetAll



Get the elements of the constraints zone cluster.

Inputs:

-- ConstraintsZone - cluster - Data structure containing constraints zone.

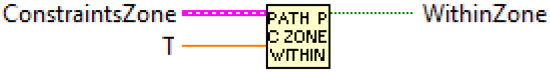
Outputs:

-- minWayPointPos - double - Waypoint relative starting position

-- maxWayPointPos - double - Waypoint relative end position

-- Constaints -- cluster -- Constraints to apply within this region.

PathPlanner_ConstraintsZone_IsWithinZone



Get if a given waypoint relative position is within this zone

Inputs:

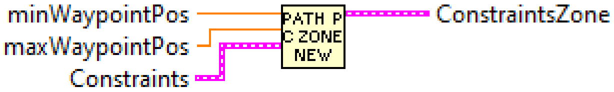
-- ConstraintsZone -- cluster -- data structure containing zone definition.

-- t - double - Waypoint relative position

Outputs:

-- WithinZone - boolean - True if given position is within this zone

PathPlanner_ConstraintsZone_New



Create a new constraints zone

Inputs:

-- minWaypointPos - double - Starting position of the zone

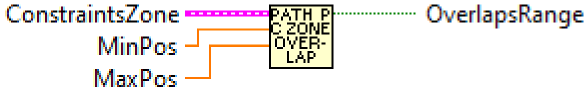
-- maxWaypointPos - double - End position of the zone

-- constraints - cluster - The constraints to apply within the zone

Outputs:

-- ConstraintsZone - cluster - data cluster with constraint

PathPlanner_ConstraintsZone_OverlapsRange



Get if this zone overlaps a given range

Inputs:

-- ConstraintsZone - cluster - zone definition.

-- minPos - double - The minimum waypoint relative position of the range

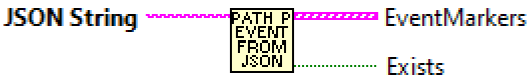
- -- maxPos - double - The maximum waypoint relative position of the range

Outputs:

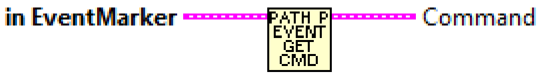
- -- OverlapsRange - boolean - True if any part of this zone is within the given range

EventMarker

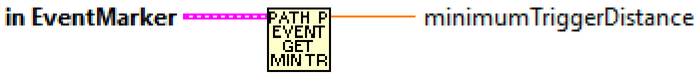
PathPlanner_EventMarker_FromJSON



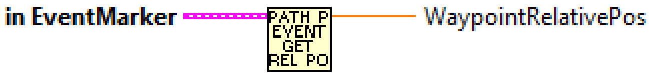
PathPlanner_EventMarker_GetCommand



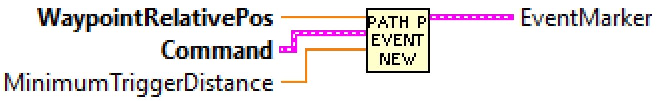
PathPlanner_EventMarker_GetMinimumTriggerDistance



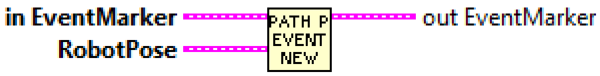
PathPlanner_EventMarker_GetWaypointRelativePos



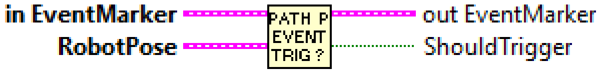
PathPlanner_EventMarker_New



PathPlanner_EventMarker_Reset

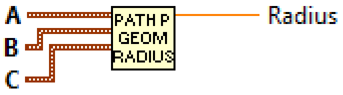


PathPlanner_EventMarker_ShouldTrigger

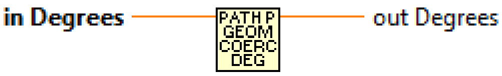


GeomUtil

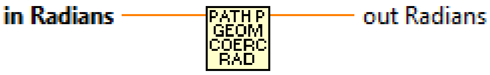
PathPlanner_GeomUtil_CalculateRadius



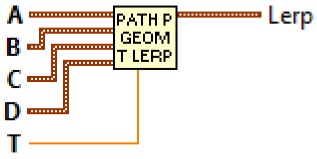
PathPlanner_GeomUtil_CoerceHeadingDegrees



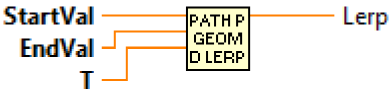
PathPlanner_GeomUtil_CoerceHeadingRadians



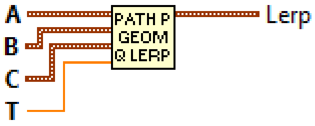
PathPlanner_GeomUtil_CubicLerp



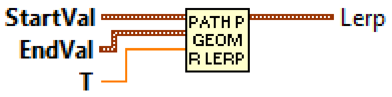
PathPlanner_GeomUtil_DoubleLerp



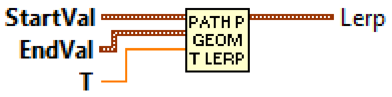
PathPlanner_GeomUtil_QuadraticLerp



PathPlanner_GeomUtil_RotationLerp

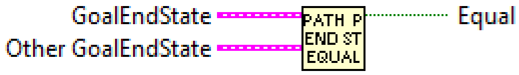


PathPlanner_GeomUtil_TranslationLerp



GoalEndState

PathPlanner_GoalEndState_Equals



Determines if two Goal End State definitions are equal

Inputs:

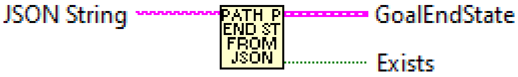
-- GoalEndState - cluster - goal end state definition

-- Other GoalEndState - cluster - goal end state definition

Outputs:

-- Equal - boolean - TRUE if both definitions are the same.

PathPlanner_GoalEndState_FromJSON



Create a goal end state from json

Inputs:

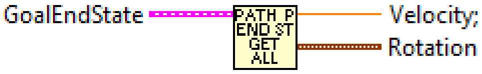
-- JSON String - string - string to parse for GoalEndState

Outputs:

-- GoalEndState - cluster - The goal end state defined by the given json. If not found, default is returned.

-- exists - boolean - TRUE if GoalEndState was found and parsed in the JSON string.

PathPlanner_GoalEndState_GetAll



Get the goal end velocity and end rotation

Inputs:

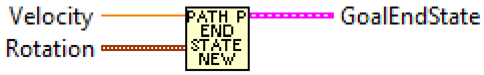
-- GoalEndState - cluster - definition data structure

Outputs:

-- Goal end velocity (M/S)

-- Goal rotation

PathPlanner_GoalEndState_New



Describes the goal end state of the robot when finishing a path */

Create a new goal end state

Inputs:

-- velocity - double - The goal end velocity (M/S)

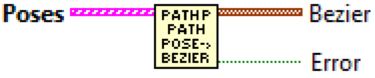
-- rotation - rotation2d - The goal rotation

Outputs:

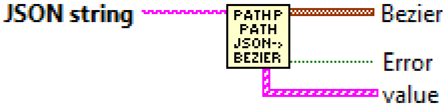
-- GoalEndState - cluster - data structure

Path

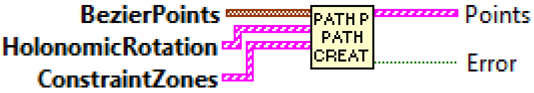
PathPlanner_Path_BezierFromPoses



PathPlanner_Path_BezierFromWaypointsJSON



PathPlanner_Path_CreatePath



Create a path from an array of PathPlanner Waypoints.   This routine calculates the trajectory states. It handles combining separate paths when reversal is flagged.

Inputs:

- PathPlannerWaypoints -- An array of PathPlannerWaypoints. This array must contain at least 2 entries.

- maxVel -- Max velocity of the path

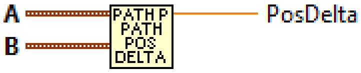
- maxAccel -- Max velocity of the path

- reversed -- Should the robot follow the path reversed

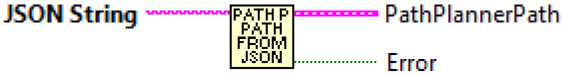
Outputs:

- PathPlanner Trajectory -- The generated path (path planner trajectory) can be converted to trajectory)

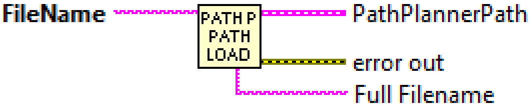
PathPlanner_Path_Equals



PathPlanner_Path_FromJSON



PathPlanner_Path_FromPathFile



Load a path file from storage.   This loads the waypoints from a file and calculates the trajectory states. It handles combining separate paths when reversal is flagged.

Inputs:

- FileName -- The name of the path to load.   Absolute or relative.
- maxVel -- Max velocity of the path
- maxAccel -- Max velocity of the path
- reversed -- Should the robot follow the path reversed

Outputs:

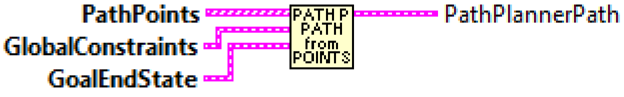
- PathPlanner Trajectory -- The generated path (path planner trajectory) can be converted to trajectory)
- Error -- error cluster

Note:

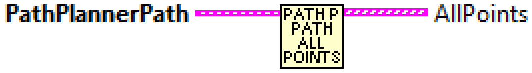
-- Default path for roboRIO is   /home/lvuser/natinst/LabVIEW Data

-- Default path for Windows is: %USERPROFILE%\Documents\LabVIEW Data

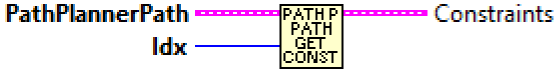
PathPlanner_Path_FromPathPonts



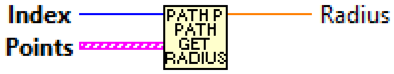
PathPlanner_Path_GetAllPathPoint



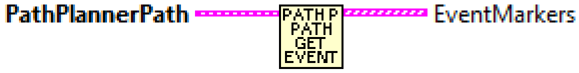
PathPlanner_Path_GetConstraintsForPoint



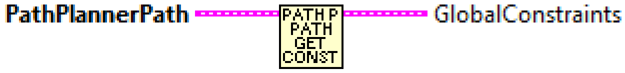
PathPlanner_Path_GetCurveRadiusAtPoint



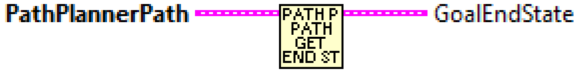
PathPlanner_Path_GetEventMarkers



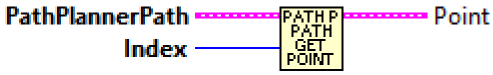
PathPlanner_Path_GetGlobalConstraints



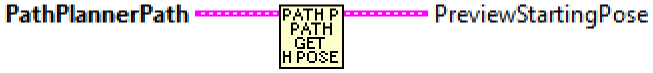
PathPlanner_Path_GetGoalEndState



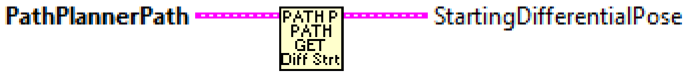
PathPlanner_Path_GetPoint



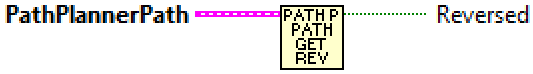
PathPlanner_Path_GetPreviewStartingHolonomicPose



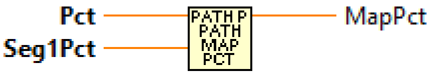
PathPlanner_Path_GetStartingDifferentialPose



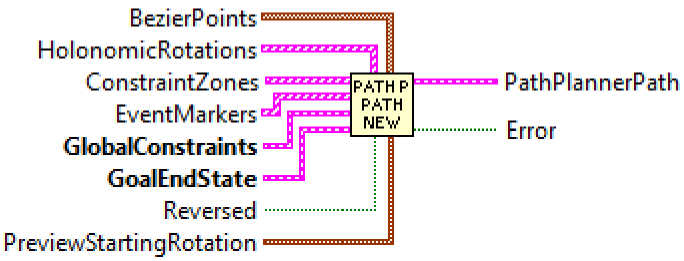
PathPlanner_Path_IsReversed



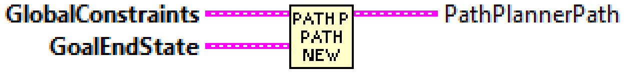
PathPlanner_Path_MapPct



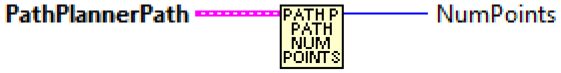
PathPlanner_Path_New



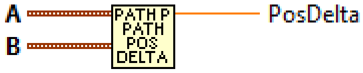
PathPlanner_Path_New_Empty



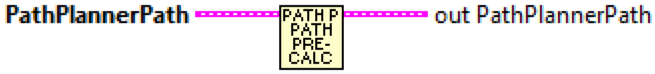
PathPlanner_Path_NumPoints



PathPlanner_Path_PositionDelta

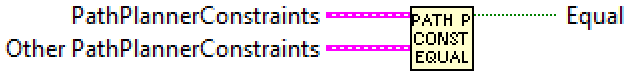


PathPlanner_Path_PreCalcValues

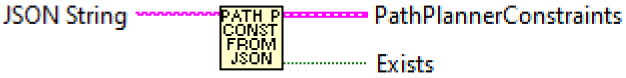


PathConstraints

PathPlanner_PathConstraints_Equals



PathPlanner_PathConstraints_FromJSON



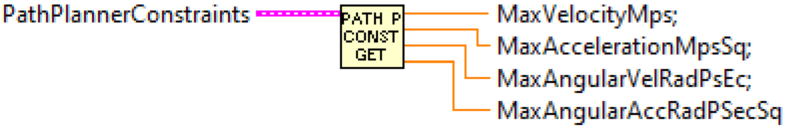
Create a path constraints object from json

- JSON String - string - string to parse for path constraints object

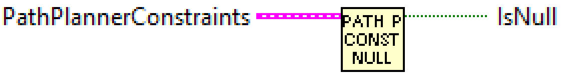
Outputs:

- PathConstraints - cluster - The path constraints defined by the given json

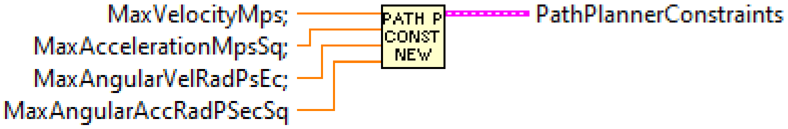
PathPlanner_PathConstraints_GetAll



PathPlanner_PathConstraints_IsNull



PathPlanner_PathConstraints_New

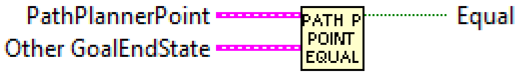


Create a new path constraints object

- maxVelocityMps Max linear velocity (M/S)
- maxAccelerationMpsSq Max linear acceleration (M/S^2)
- maxAngularVelocityRps Max angular velocity (Rad/S)
- maxAngularAccelerationRpsSq Max angular acceleration (Rad/S^2)

PathPoint

PathPlanner_PathPoint_Equals



Determines if two Path Point definitions are equal

Inputs:

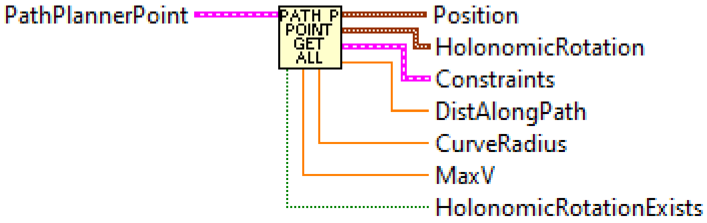
-- PathPoint - cluster - point definition

-- Other PathPoint - cluster - point definition

Outputs:

-- Equal - boolean - TRUE if both definitions are the same.

PathPlanner_PathPoint_GetAll



Gets elements of PathPoint

Inputs:

-- PathPoint - cluster - point definition

Outputs:

-- Position - Translation2d - position of point

-- HolonomicRotation - Rotation2d - rotational orientation of point

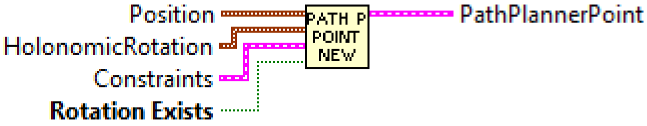
-- Constraints - cluster - contraints at this oint

-- DistAlongPath - double -

-- CurveRadius - double -

-- MaxV - double

PathPlanner_PathPoint_New



Create a path point

Inputs:

-- position Position of the point

-- holonomicRotation Rotation target at this point

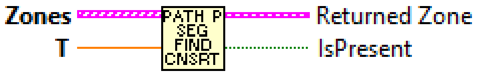
-- constraints The constraints at this point

Outputs:

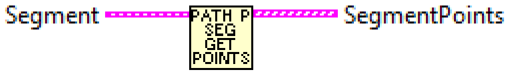
-- PathPlannerPoint - cluster - point definition

PathSegment

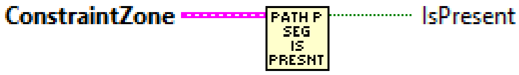
PathPlanner_PathSegment_FindConstraintZone



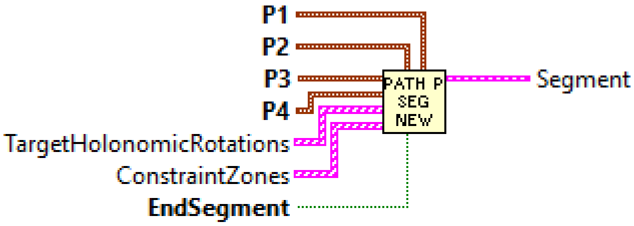
PathPlanner_PathSegment_GetSegmentPoints



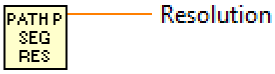
PathPlanner_PathSegment_IsPresent



PathPlanner_PathSegment_New

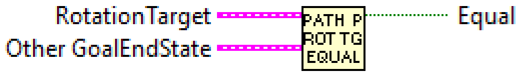


PathPlanner_PathSegment_Resolution

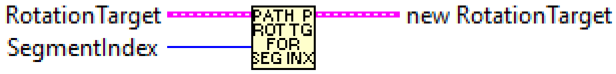


RotationTarget

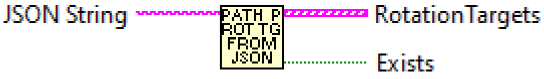
PathPlanner_RotationTarget_Equals



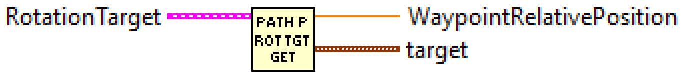
PathPlanner_RotationTarget_ForSegmentIndex



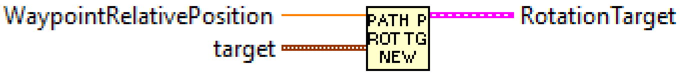
PathPlanner_RotationTarget_FromJSON



PathPlanner_RotationTarget_GetAll

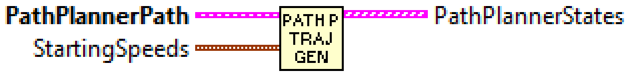


PathPlanner_RotationTarget_New

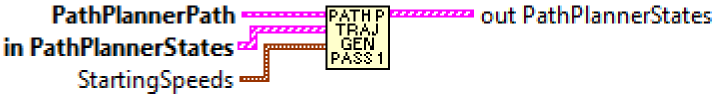


Trajectory

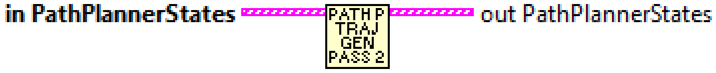
PathPlanner_Trajectory_GenerateStates



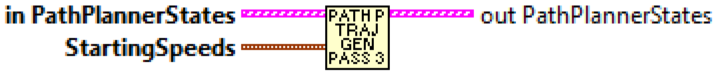
PathPlanner_Trajectory_GenerateStates_Pass1



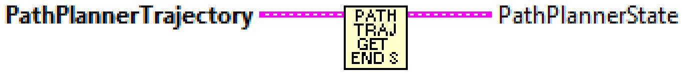
PathPlanner_Trajectory_GenerateStates_Pass2



PathPlanner_Trajectory_GenerateStates_Pass3



PathPlanner_Trajectory_GetEndState



Get the end state of the path

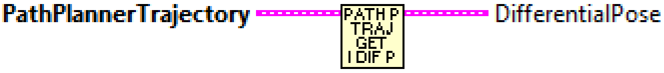
Inputs:

-- PathPlannerTraectory -- PathPlanner Trajectory data cluster

Outputs:

- PathPlannerState -- The last state in the path

PathPlanner_Trajectory_GetInitialDifferentialPose

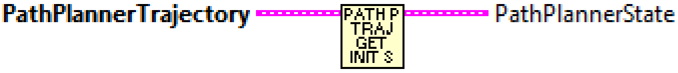


Inputs:

-- PathPlannerTraectory -- PathPlanner Trajectory data cluster

Outputs:

PathPlanner_Trajectory_GetInitialState



Get the initial state of the path

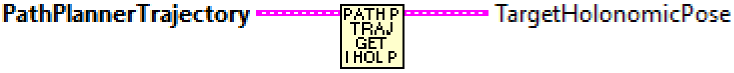
Inputs:

-- PathPlannerTraectory -- PathPlanner Trajectory data cluster

Outputs:

- PathPlannerState -- The first state in the path

PathPlanner_Trajectory_GetInitialTargetHolonomicPose

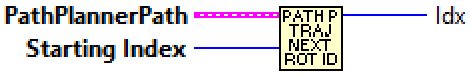


Inputs:

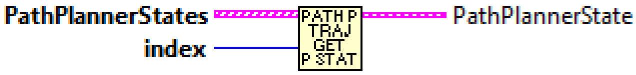
-- PathPlannerTraectory -- PathPlanner Trajectory data cluster

Outputs:

PathPlanner_Trajectory_GetNextRotationTargetIdx



PathPlanner_Trajectory_GetState



Get a state in the path based on its index.

In most cases, using sample() is a better method.

Inputs:

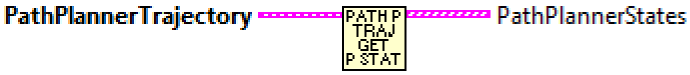
-- PathPlannerTraectory -- PathPlanner Trajectory data cluster

-- index -- The index of the state to retrieve

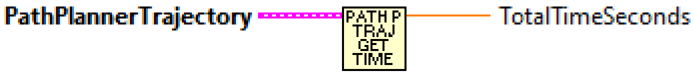
Outputs:

- PathPlannerState -- The state at the given index

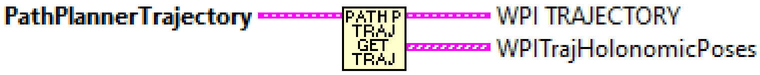
PathPlanner_Trajectory_GetStates



PathPlanner_Trajectory_GetTotalTime



PathPlanner_Trajectory_GetWPITrajectory



Convert a PathPlanner trajectory into a LabVIEW / WPILib Trajectory.

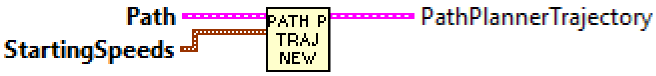
Inputs:

-- PathPlannerTrajectory -- PathPlanner Trajectory data cluster

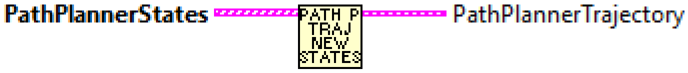
Outputs

-- Traectory -- LabVIEW traectory library (WPlib style) trajectory data cluster.

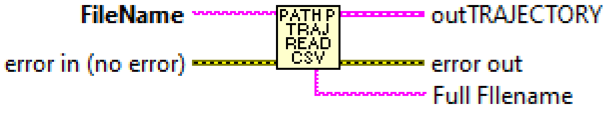
PathPlanner_Trajectory_New



PathPlanner_Trajectory_New_States



PathPlanner_Trajectory_ReadCSVFile



Create a trajectory from a CSV file. This can be used on a PC or the RoboRIO. Normally the CSV file is created as output from one of the trajectory utility programs. The file could also be created manually or by a custom written program.

Parameters:

- FileName -- Name of the CSV file to read. See file name notes for additional information.

- Error In -- Input error cluster (optional)

Returns:

- outTrajectory - Trajectory data structure cluster

- Error out - returned error cluster

Notes on use:

-- This routine writes informational messges to the console and to the driver station log.

Notes on file naming:

-- The file name must include the extention. ".csv" is not automatically appended to the name.

-- The file name can be a simple file or an absolute path. If a simple file name is used the default path on the RoboRIO is: "home:\lvuser\natinst\LabVIEW Data". On a Windows PC the default path is the LabVIEW default directory. Normally this is: ❽%HOMEDRIVE%%HOMEPATH%\Documents\LabView Data".

-- Filenames on the RoboRIO, which runs Linux, are case sensitive.

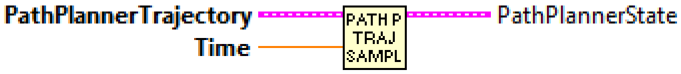
Notes on file contents:

-- Blank lines are ignored.

-- Lines that begin with either #, !, or ' in the first character are considered comments and are ignored.

-- Other lines are interpretted as comma separated data

PathPlanner_Trajectory_Sample



Sample the path at a point in time

Inputs:

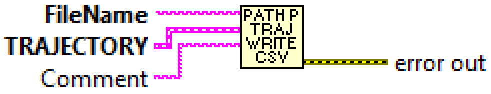
-- PathPlannerTraectory -- PathPlanner Trajectory data cluster

-- time -- The time to sample

Outputs:

- PathPlannerState -- The state at the given point in time

PathPlanner_Trajectory_WriteCSVFile



Create a CSV file from a trajectory. This can be used on a PC or the RoboRIO.

Parameters:

- FileName -- Name of the CSV file to read. See file name notes for additional information.

- Trajectory - Trajectory data structure cluster

- Comment - string - Optional comment to place in CSV file.

Returns:

- Error out - returned error cluster

Notes on file naming:

-- The file name must include the extention. ".csv" is not automatically appended to the name.

-- The file name can be a simple file or an absolute path. If a simple file name is used the default path on the RoboRIO is: "home:\lvuser\natinst\LabVIEW Data". On a Windows PC the default path is the LabVIEW default directory. Normally this is: ❽%HOMEDRIVE%%HOMEPATH%\Documents\LabView Data".

-- Filenames on the RoboRIO, which runs Linux, are case sensitive.

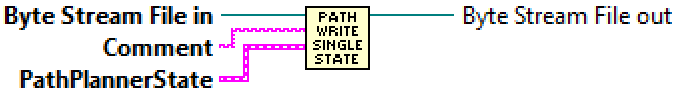
Notes on file contents:

-- Blank lines are ignored.

-- Lines that begin with either #, !, or ' in the first character are considered comments and are ignored.

-- Other lines are interpretted as comma separated data

PathPlanner_Trajectory_WriteCSVFileIndividualState



Internal subVI used by Util_Trajectory_WriteFile (and others). This writes one trajectory state to a file.

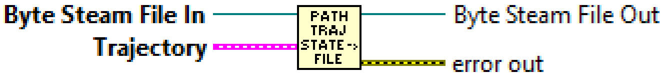
Parameters:

- Byte stream in - file stream
- comment - comment for this line
- TrajectoryState - The state to write

Returns:

- Byte Stream Out - file stream

PathPlanner_Trajectory_WriteCSVFileStates



Write trajectory states to a file. This is an internal routine

Parameters:

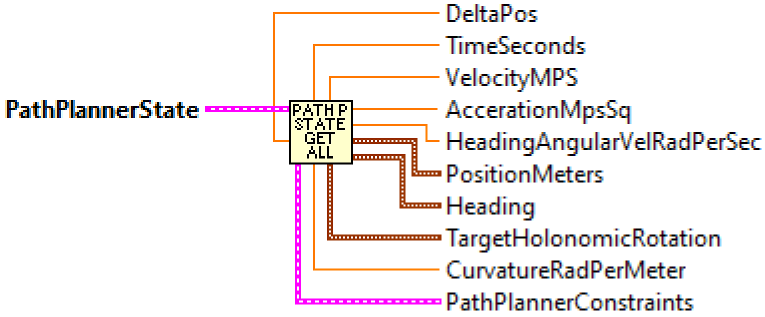
- ByteStreamIn - File stream
- Trajectory - Data structure containing trajectory

Returns:

- ByteStreamOut - File stream
- Error out - returned error cluster

TrajectoryState

PathPlanner_TrajectoryState_GetAll



Gets elements of trajectory state

Inputs

- PathPlannerTrajectoryState -- cluster -- State data structure

Outputs:

-- timeSeconds - double - The time at this state in seconds (default = 0;)

-- velocityMps - double - The velocity at this state in m/s (default = 0)

-- accelerationMpsSq - double - The acceleration at this state in m/s^2 (default = 0)

-- headingAngularVelocityRps - double - The time at this state in seconds (default = 0)

-- positionMeters - translation2d - The position at this state in meters (default = 0,0)

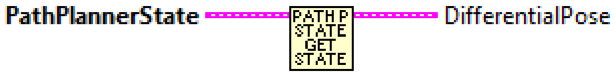
-- heading - rotation2d - The heading (direction of travel) at this state (default = 0)

-- targetHolonomicRotation - rotation2d - The target holonomic rotation at this state (default = 0)

-- curvatureRadPerMeter - double - The curvature at this state in rad/m (default = 0)

-- constraints -- cluster -- constraints to apply at this state (default - none)

PathPlanner_TrajectoryState_GetDifferentialPose

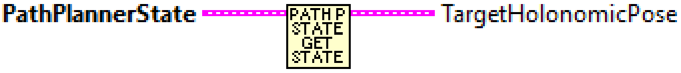


Inputs:

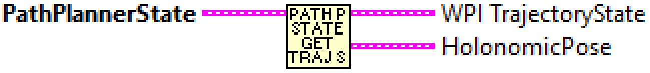
-- PathPlannerState -- Path Planner trajectory state

Outputs:

PathPlanner_TrajectoryState_GetTargetHolonomicPose



PathPlanner_TrajectoryState_GetWPITrajectoryState



Get Trajectory Library / WPILIB trajectory state from a PathPlanner Trajectory State

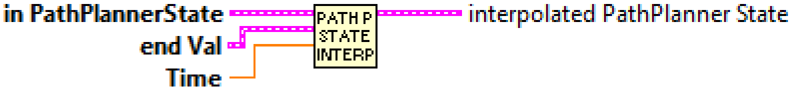
Inputs:

-- PathPlannerState -- Path Planner trajectory state

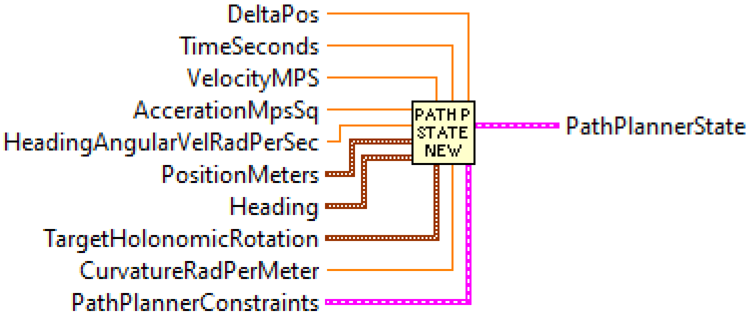
Outputs:

-- TrajectoryState -- LabVIEW trajectory library / WPILib trajectory state.

PathPlanner_TrajectoryState_Interpolate



PathPlanner_TrajectoryState_New



Create a trajectory state

Inputs:

-- timeSeconds - double - The time at this state in seconds (default = 0;)

-- velocityMps - double - The velocity at this state in m/s (default = 0)

-- accelerationMpsSq - double - The acceleration at this state in m/s^2 (default = 0)

-- headingAngularVelocityRps - double - The time at this state in seconds (default = 0)

-- positionMeters - translation2d - The position at this state in meters (default = 0,0)

-- heading - rotation2d - The heading (direction of travel) at this state (default = 0)

-- targetHolonomicRotation - rotation2d - The target holonomic rotation at this state (default = 0)

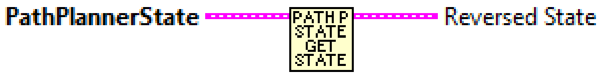
-- curvatureRadPerMeter - double - The curvature at this state in rad/m (default = 0)

-- constraints -- cluster -- constraints to apply at this state (default - none)

Outputs

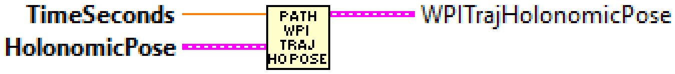
- PathPlannerTrajectoryState -- cluster -- Newly created state

PathPlanner_TrajectoryState_Reverse

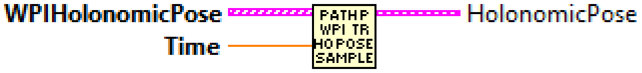


WPITrajHolPose

PathPlanner_WPITrajHolPose_New



PathPlanner_WPITrajHolPose_Sample



Sample the path at a point in time

Inputs:

-- PathPlannerTraectory -- PathPlanner Trajectory data cluster

-- time -- The time to sample

Outputs:

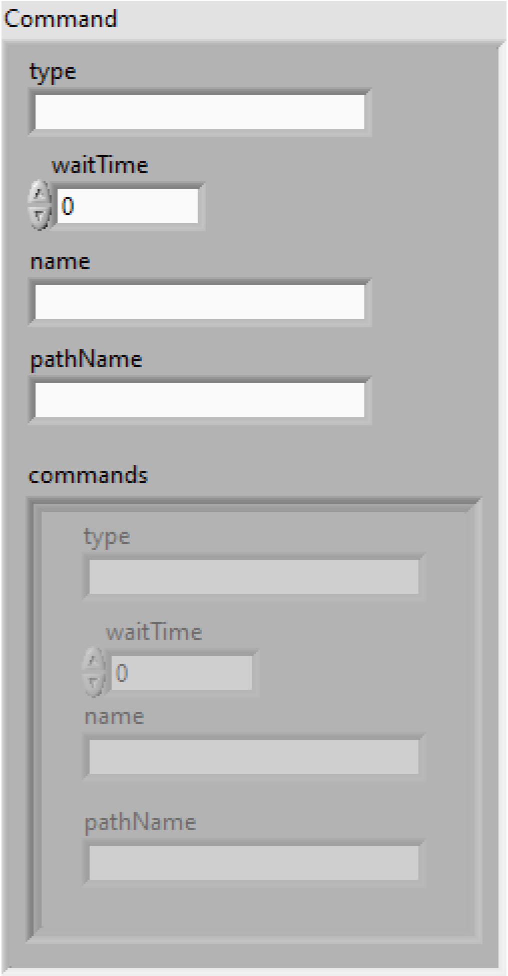
- PathPlannerState -- The state at the given point in time

Type Definitions

TypeDef

TypeDef-PathPlannerCommand

PATH P
CMD



Command

type

waitTime

0

name

pathName

commands

type

waitTime

0

name

pathName

TypeDef-PathPlannerConstraintsZone

PATH P
CNSTR
ZONE

A zone on a path with different kinematic constraints

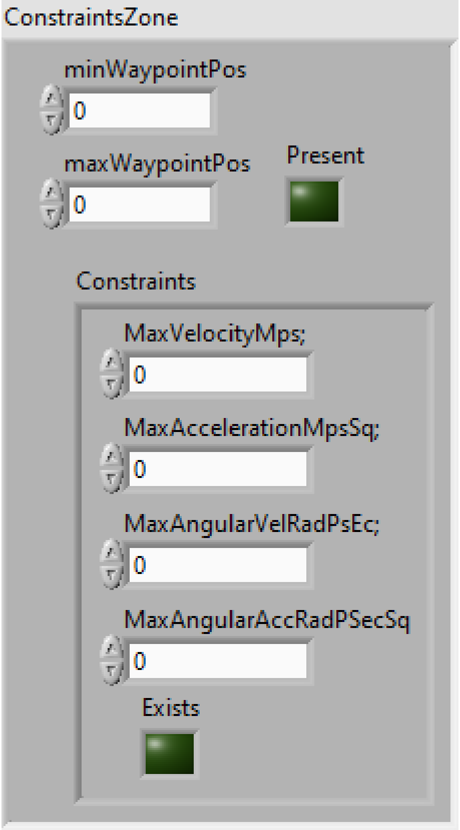
Contains:

-- MinWayPointPos - double - Starting distance on path to apply constraint

-- MaxWayPointPos - double - Ending distance on path to apply constraint

-- Constraint - cluster - Constraint to apply

-- Present - boolean - flag indicting this cluster is not null



ConstraintsZone

minWaypointPos 0

maxWaypointPos 0

Present

Constraints

MaxVelocityMps; 0

MaxAccelerationMpsSq; 0

MaxAngularVelRadPsEc; 0

MaxAngularAccRadPSecSq 0

Exists

TypeDef-PathPlannerEventMarker

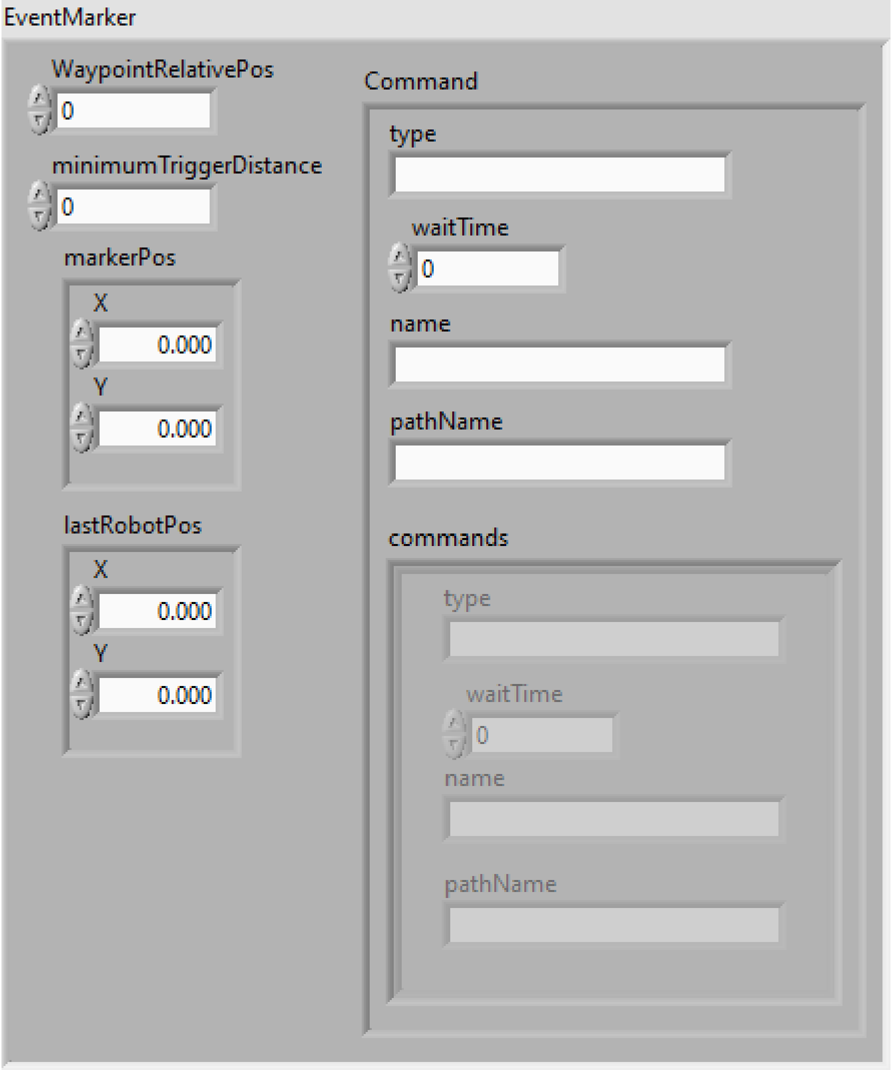
PATH P EVENT MARK

Describes the goal end state of the robot when finishing a path

contains:

double velocity

Rotation2d rotation



EventMarker

WaypointRelativePos
0

minimumTriggerDistance
0

markerPos
X 0.000
Y 0.000

lastRobotPos
X 0.000
Y 0.000

Command
type

waitTime
0

name

pathName

commands
type

waitTime
0

name

pathName

TypeDef-PathPlannerGoalEndState

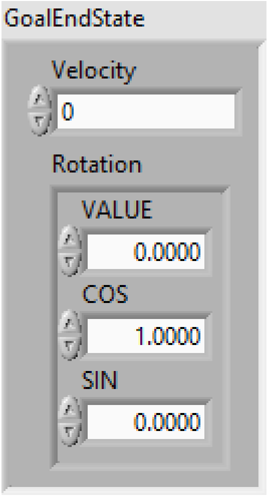
PATH P
END
STATE

Describes the goal end state of the robot when finishing a path

contains:

-- velocity - double

-- rotation - Rotation2d



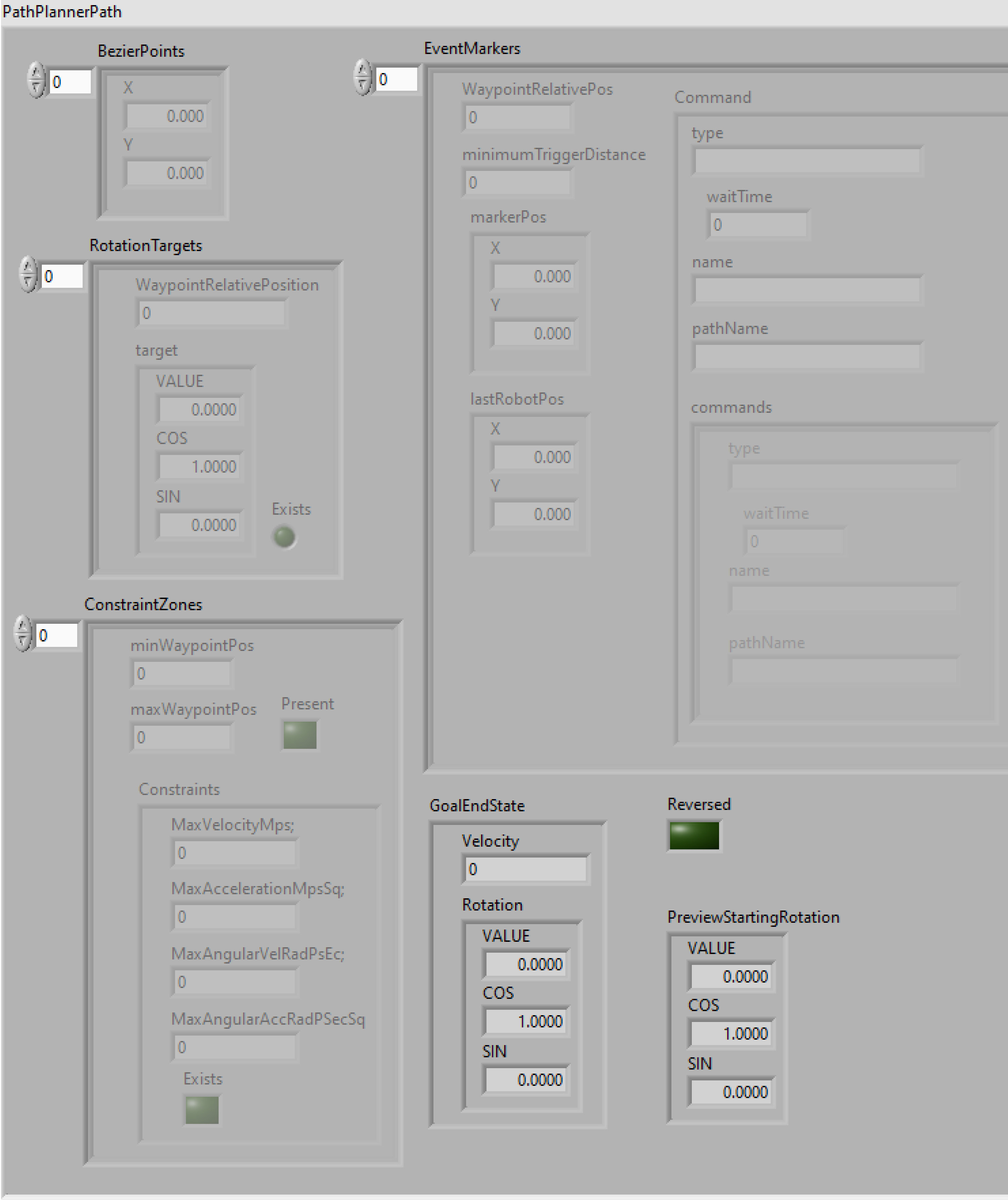
GoalEndState

Velocity: 0

Rotation

- VALUE: 0.0000
- COS: 1.0000
- SIN: 0.0000

TypeDef-PathPlannerPath



PathPlannerPath

BezierPoints
0
X
0.000
Y
0.000

RotationTargets
0
WaypointRelativePosition
0
target
VALUE
0.0000
COS
1.0000
SIN
0.0000
Exists

ConstraintZones
0
minWaypointPos
0
maxWaypointPos
0
Present
Constraints
MaxVelocityMps;
0
MaxAccelerationMpsSq;
0
MaxAngularVelRadPsEc;
0
MaxAngularAccRadPSecSq
0
Exists

EventMarkers
0
WaypointRelativePos
0
minimumTriggerDistance
0
markerPos
X
0.000
Y
0.000
lastRobotPos
X
0.000
Y
0.000
Command
type

waitTime
0
name

pathName

commands
type

waitTime
0
name

pathName

GoalEndState
Velocity
0
Rotation
VALUE
0.0000
COS
1.0000
SIN
0.0000

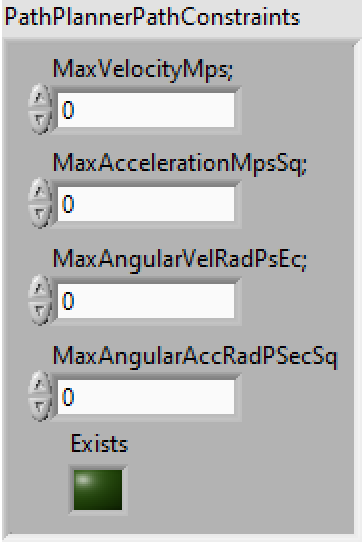
Reversed

PreviewStartingRotation
VALUE
0.0000
COS
1.0000
SIN
0.0000

TypeDef-PathPlannerPathConstraints

PATH P
CONST

Kinematic path following constraints



PathPlannerPathConstraints

MaxVelocityMps; 0

MaxAccelerationMpsSq; 0

MaxAngularVelRadPsEc; 0

MaxAngularAccRadPSecSq 0

TypeDef-PathPlannerPathPoint

PATH P
POINT

A point along a pathplanner path

Contains:

-- position - translation2d - The position of this point

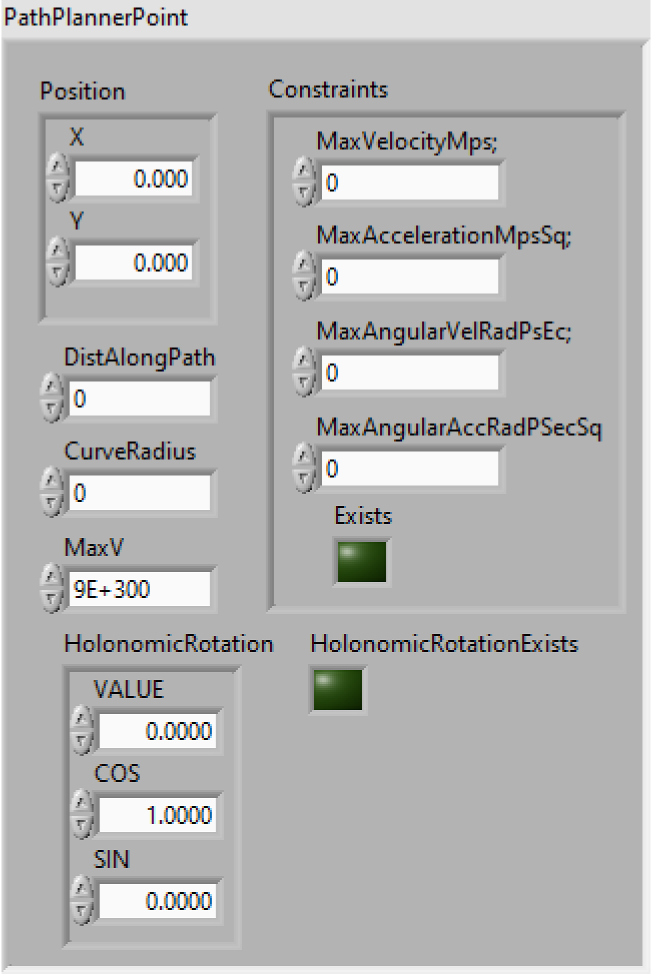
-- distanceAlongPath - double - The distance of this point along the path, in meters

-- CurveRadius - double - The curve radius at this point

-- MaxV - double - The max velocity at this point

-- holonomicRotation - Rotation2d - The target rotation at this point

-- constraints - cluster - The constraints applied to this point



PathPlannerPoint

Position

X 0.000

Y 0.000

DistAlongPath 0

CurveRadius 0

MaxV 9E+300

HolonomicRotation

VALUE 0.0000

COS 1.0000

SIN 0.0000

Constraints

MaxVelocityMps; 0

MaxAccelerationMpsSq; 0

MaxAngularVelRadPsEc; 0

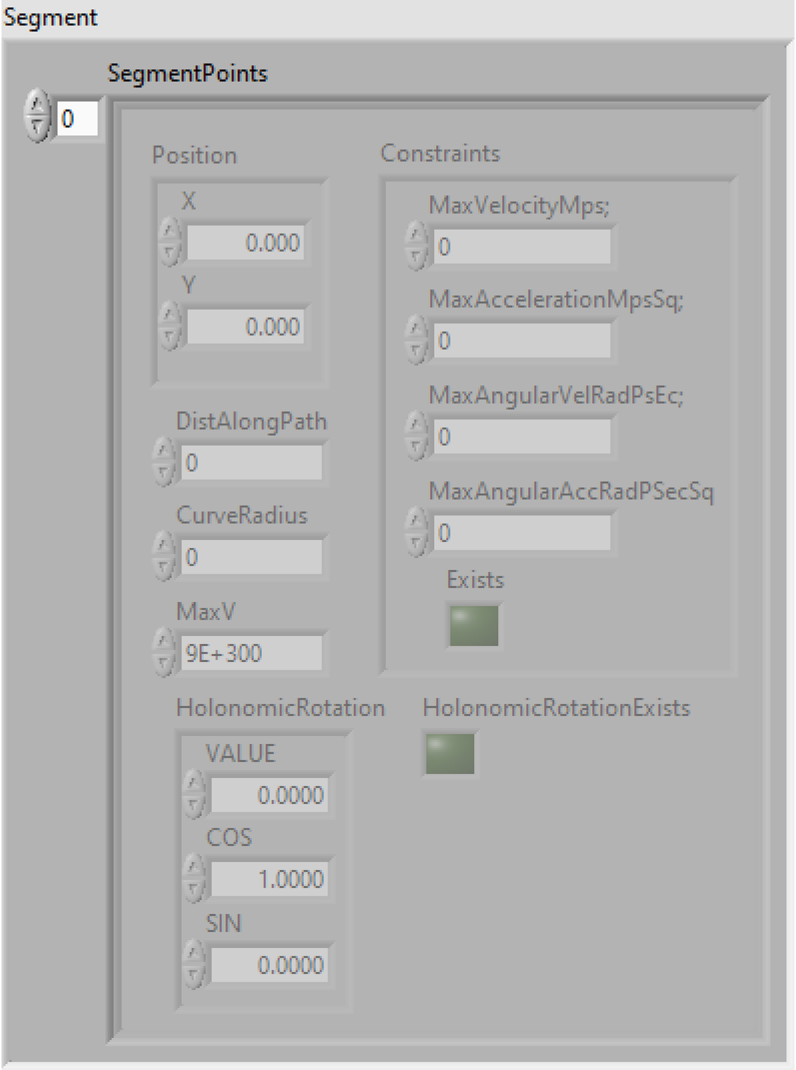
MaxAngularAccRadPSecSq 0

Exists

HolonomicRotationExists

TypeDef-PathPlannerPathSegment

PATH P
SEG-
MENT



Segment

SegmentPoints: 0

Position
X: 0.000
Y: 0.000

DistAlongPath: 0

CurveRadius: 0

MaxV: 9E+300

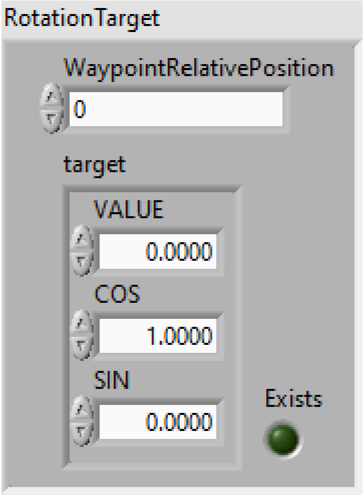
Constraints
MaxVelocityMps;: 0
MaxAccelerationMpsSq;: 0
MaxAngularVelRadPsEc;: 0
MaxAngularAccRadPSecSq: 0
Exists

HolonomicRotation
VALUE: 0.0000
COS: 1.0000
SIN: 0.0000

HolonomicRotationExists

TypeDef-PathPlannerRotationTarget

PATH P
ROT
TARG



RotationTarget

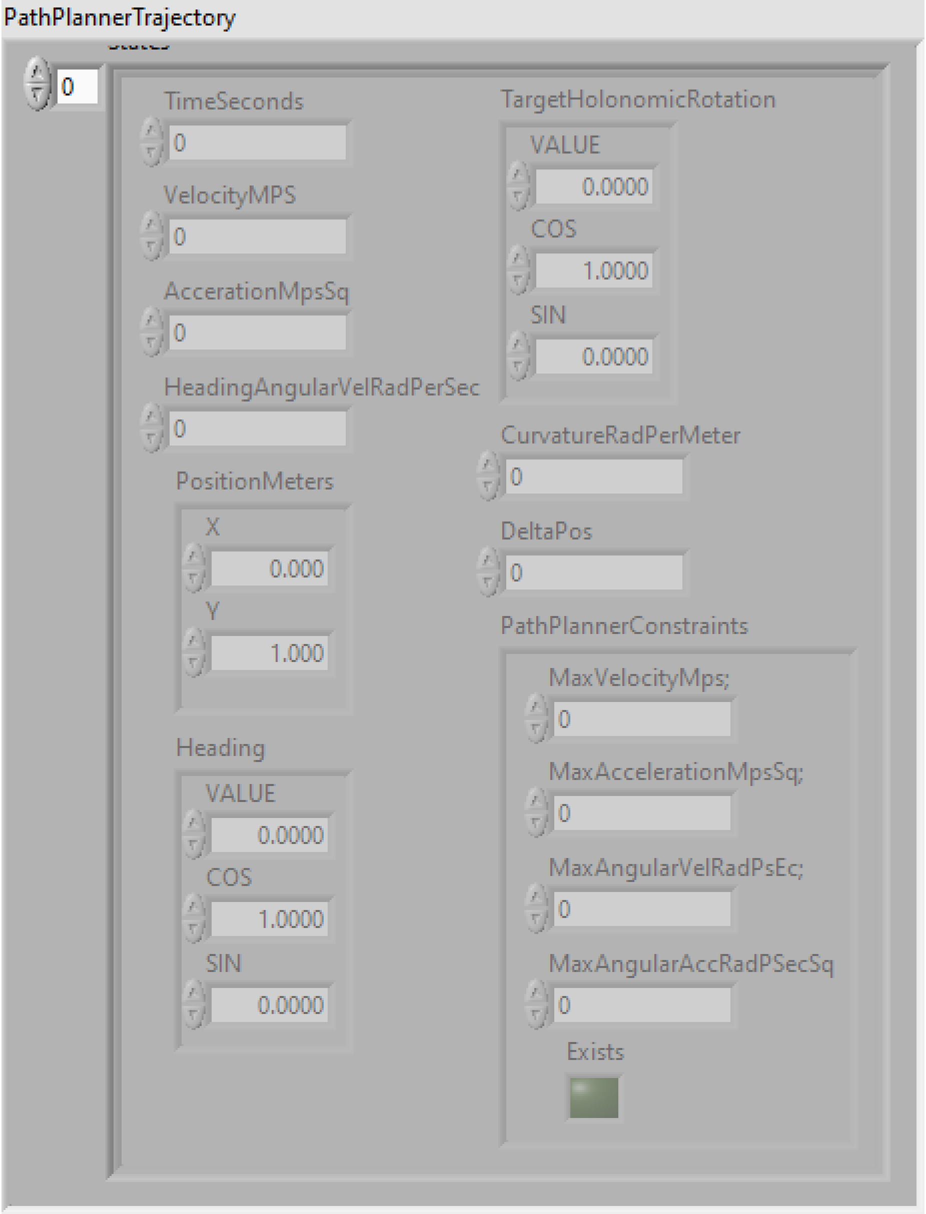
WaypointRelativePosition: 0

target
VALUE: 0.0000
COS: 1.0000
SIN: 0.0000

Exists

TypeDef-PathPlannerTrajectory

PATH P
TRAJ



PathPlannerTrajectory

0

TimeSeconds: 0

VelocityMPS: 0

AccerationMpsSq: 0

HeadingAngularVelRadPerSec: 0

PositionMeters
- X: 0.000
- Y: 1.000

Heading
- VALUE: 0.0000
- COS: 1.0000
- SIN: 0.0000

TargetHolonomicRotation
- VALUE: 0.0000
- COS: 1.0000
- SIN: 0.0000

CurvatureRadPerMeter: 0

DeltaPos: 0

PathPlannerConstraints
- MaxVelocityMps;: 0
- MaxAccelerationMpsSq;: 0
- MaxAngularVelRadPsEc;: 0
- MaxAngularAccRadPSecSq: 0
- Exists

TypeDef-PathPlannerTrajectoryState

PATH P
TRAJ
STATE

PathPlanner Trajectory State

-- The time at this state in seconds

timeSeconds;

-- The velocity at this state in m/s

velocityMps

-- The acceleration at this state in m/s^2

accelerationMpsSq

-- ????

headingAngularVelocityRps

-- The position at this state in meters

positionMeters

-- The heading (direction of travel) at this state

heading

-- The target holonomic rotation at this state

targetHolonomicRotation

-- The curvature at this state in rad/m

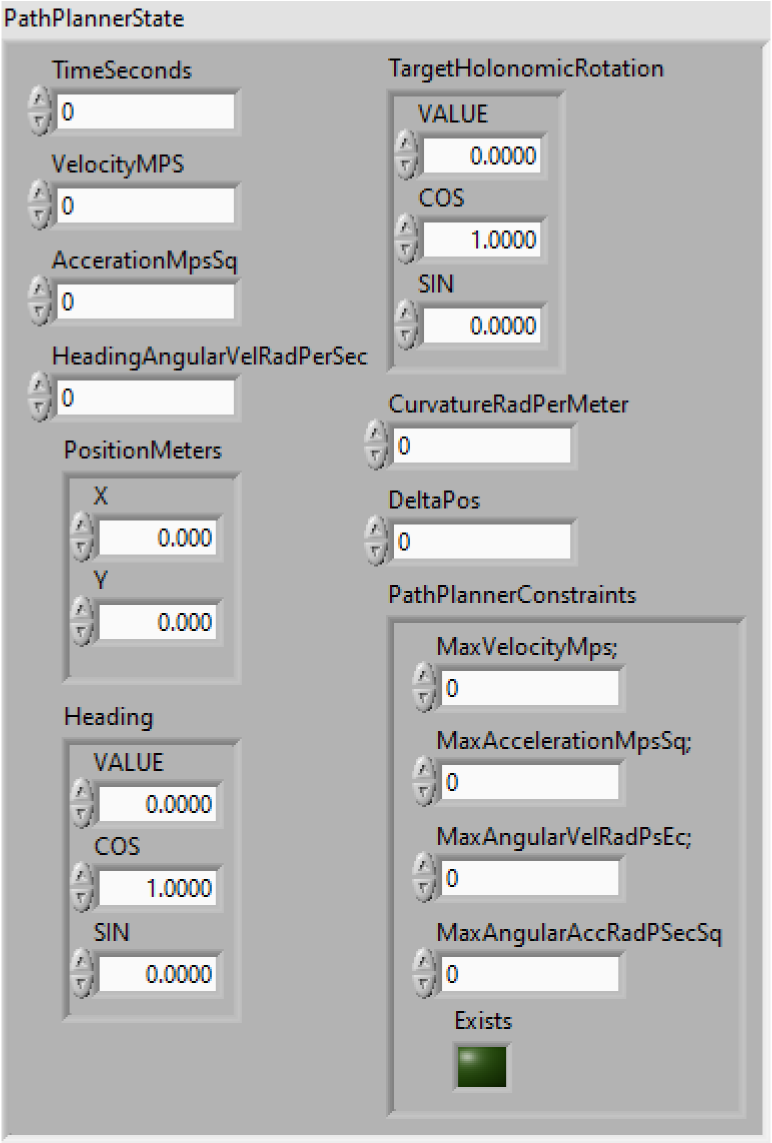
curvatureRadPerMeter

-- The constraints to apply at this state

constraints

-- Values only used during generation

deltaPos



PathPlannerState

TimeSeconds
0

VelocityMPS
0

AccerationMpsSq
0

HeadingAngularVelRadPerSec
0

PositionMeters
X
0.000
Y
0.000

Heading
VALUE
0.0000
COS
1.0000
SIN
0.0000

TargetHolonomicRotation
VALUE
0.0000
COS
1.0000
SIN
0.0000

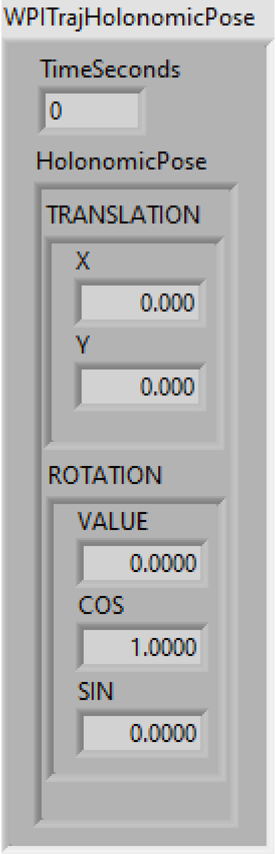
CurvatureRadPerMeter
0

DeltaPos
0

PathPlannerConstraints
MaxVelocityMps;
0
MaxAccelerationMpsSq;
0
MaxAngularVelRadPsEc;
0
MaxAngularAccRadPSecSq
0
Exists

TypeDef-PathPlannerWPITrajHolonomicPose

WPI
TRAJ
HOL
POSE



WPITrajHolonomicPose

TimeSeconds
0

HolonomicPose

TRANSLATION

X
0.000

Y
0.000

ROTATION

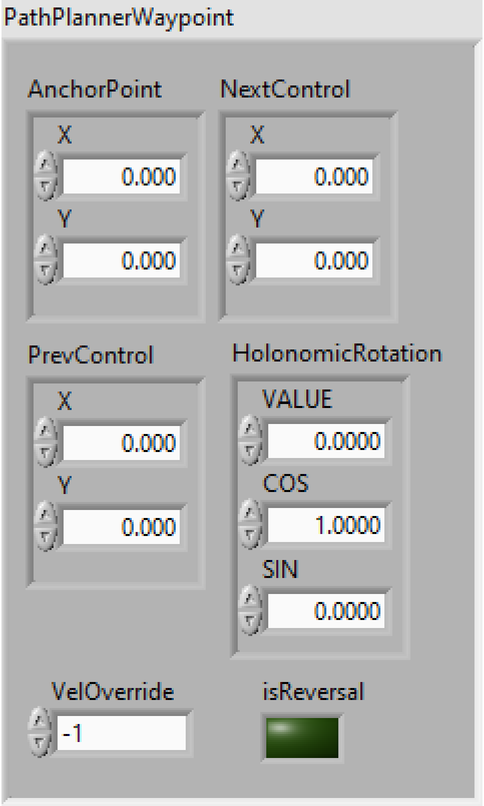
VALUE
0.0000

COS
1.0000

SIN
0.0000

TypeDef-PathPlannerWaypoint

PATH P
WAY-
POINT



PathPlannerWaypoint

AnchorPoint
X 0.000
Y 0.000

NextControl
X 0.000
Y 0.000

PrevControl
X 0.000
Y 0.000

HolonomicRotation
VALUE 0.0000
COS 1.0000
SIN 0.0000

VelOverride -1

isReversal 🟩