Adept Lynx Triangle Drive: Configuration and Applications

Parker Conroy
Version 4: April 7 2014

Summary

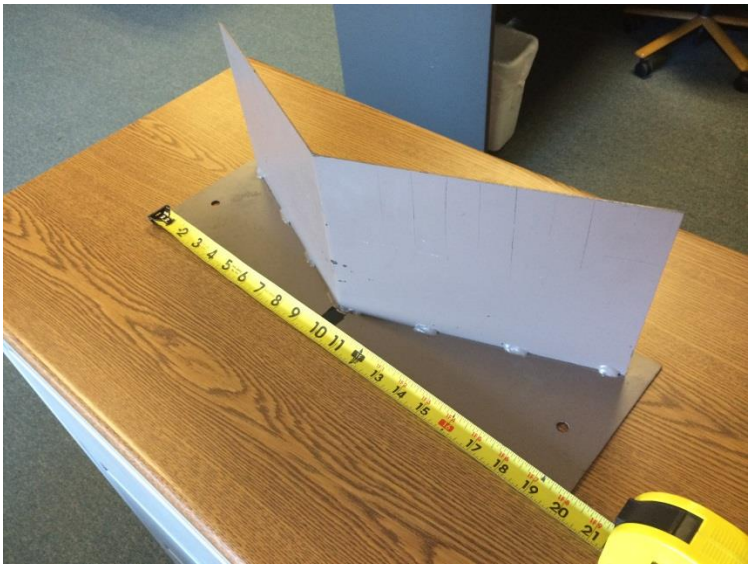
Triangle drive is a positioning method by which the Lynx maneuvers with increased accuracy relative to a physical triangle situated in the environment. This triangle is defined by its edge lengths and inclusive angle, giving the Lynx a specific feature in the environment to use for positioning. Such a triangle can be seen in the image above, where a conveyor-Lynx is positioning itself in front of a static conveyor. While a variety of use cases for this positioning method exist, all use the “TriangleDriveTo” or “TriangleDriveToAdvanced” robot tasks, which define the specifics for the movement.

Components

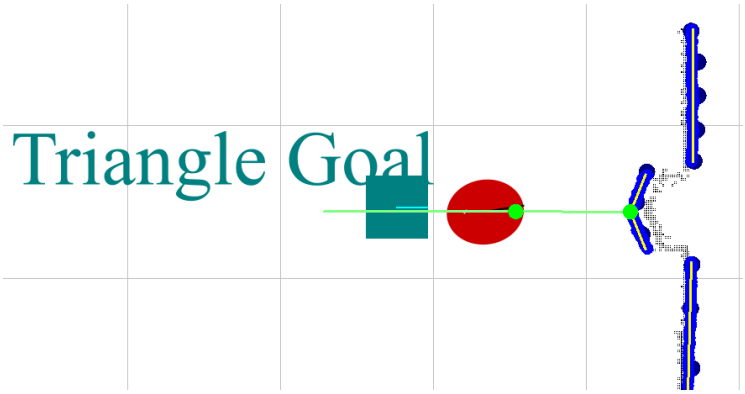
- The “TriangleDriveTo” task requires a physical triangle to be placed in the environment.
- Ideally the triangle should be unique to its immediate surroundings so that the Lynx does not mistake the surroundings as part of the triangle.
- The triangle should be kept in a clean, clutter free, area.
- The triangle should not have an angle near 90 degrees, lest it be confused with the corner of a room or other common objects.
- The triangle must also be of sufficient height to be seen by the Lynx’s horizontal, forward-facing laser. Ideally, the triangle should be three inches above and below the seven inch high laser so that it can be seen when the floor is not perfectly even

Building a custom triangle should begin with looking at the Lynx dock, which relies on triangle drive for automatic charging. Because of its need to be compact, the dock is as small of a triangle as should be used with the triangle drive methods. Generally, larger triangles result in more accurate movements. Seen below, the dock features a triangle measuring ~6.5 inches with an angle of -135 degrees. Adept advises against painting your triangle with low-luster paints, especially black.

Shown below is a larger triangle made of sheet metal which can be used in either an outgoing or inward facing vertex orientation.



When positioning using such a triangle, MobilePlanner can show the line extending out of the vertex as seen in this image. The green point in the image is the approach point. This feature "TriangleAdvancedNewApproch" can be enabled in MobilePlanner via the Map>Robot Data> Other Robot Data menu.

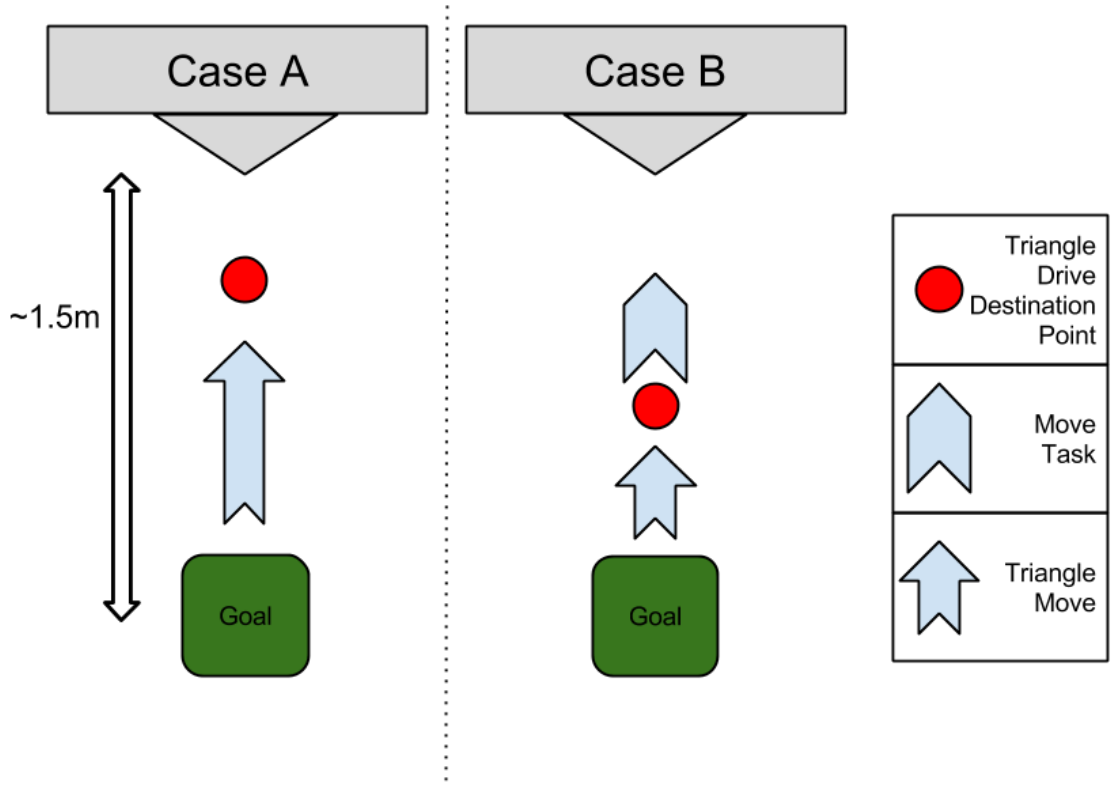


Use cases

Both of the cases given below result in the Lynx robot arriving at the same position, with 600mm between the vertex of the triangle and the center of the Lynx robot. What differs is how the robot will complete the task. In Case A, the Lynx will perform a few back and forth heading changes along its path to the desired point. This is the more general case in which the TriangleDriveTo task will be used. This method is often more accurate as well, due to the constant error corrections the Lynx performs as it reaches the desired point.

Often tight tolerances in the environment make it so that the robot has minimal extra room to navigate to its desired point. Case B shows an alternative method that further specifies the movement of the

Lynx. By navigating to the desired point before driving close to the triangle, the heading changes are kept at a further distance from potential obstacles. A “Move” task is then used to drive the robot straight forward into the desired position.



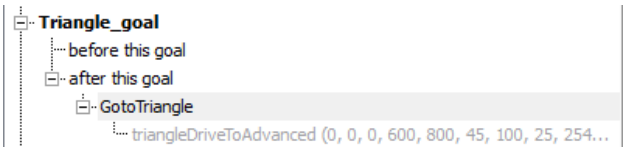
Case A

The robot drives to the goal (the green rectangle) and adjusts its heading to within the tolerance set in its Path Planning settings. Generally the goal in which you call the triangle drive task should be placed 1 to 1.5 meters away from the vertex of the triangle. The Lynx begins the triangle drive task by using its laser scanner to detect the triangle mounted on the wall, and starts to drive toward the destination point which is specified as a distance from the vertex of the triangle. This point is shown as a red circle in the graphic.

As the robot drives forward it continues to scan for the triangle and correct its position. This movement is not perfectly straight and may include heading changes, alternating between driving left or right until the robot has arrived at the destination point.

Starting the triangle drive tasks with the robot closely aligned with the triangle will naturally require less correction. As such, it is best to place the goal collinear with the line extending out of the triangle vertex so that the corrections will be minimal.

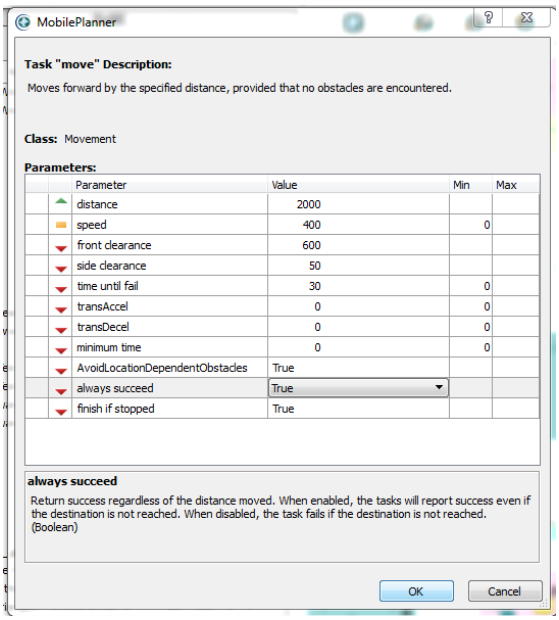
The TriangleDriveToAdvanced task should be first placed in a macro, as this makes it easier to use in multiple locations. That marco, named here “GoToTriangle” is put after the goal, here referenced as Triangle_goal. This goal should be setup as to orient the robot in the direction of the triangle.



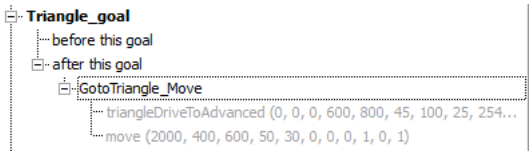
Case B

In the second case, the movement is broken up into two tasks, the triangle drive and a move. The FinalDistanceFromVertex parameter of the TriangleDriveToAdvanced task is changed so that the destination of the triangle drive is well before the final desired robot location. After the initial triangle drive, the Move task is used to drive the robot directly forward for a specified distance by varying the frontClearance. This and other parameters are further explained in the programming section of this document.

Quite often, as with this case, it is more appropriate to be concerned with the final distance from the wall, or triangle, than the distance from the starting point. The solution shown sets the distance parameter to something higher than what is needed (2000, or 2 meters in the example) then sets the frontClearance parameter to equal the desired distance from the wall (600 mm in the example). Next, set the parameter always succeed to true. Caution should be used in that the only object that will move in front of the robot is the triangle itself, as the task is now designed to always succeed. Always succeeding implies that even if the robot does not travel the entire distance specified by the distance parameter, it will still complete the task without error. The result of using this strategy is that the robot will drive straight forward until it detects that a laser reading has entered the front clearance, at which point it will stop and the task will succeed.



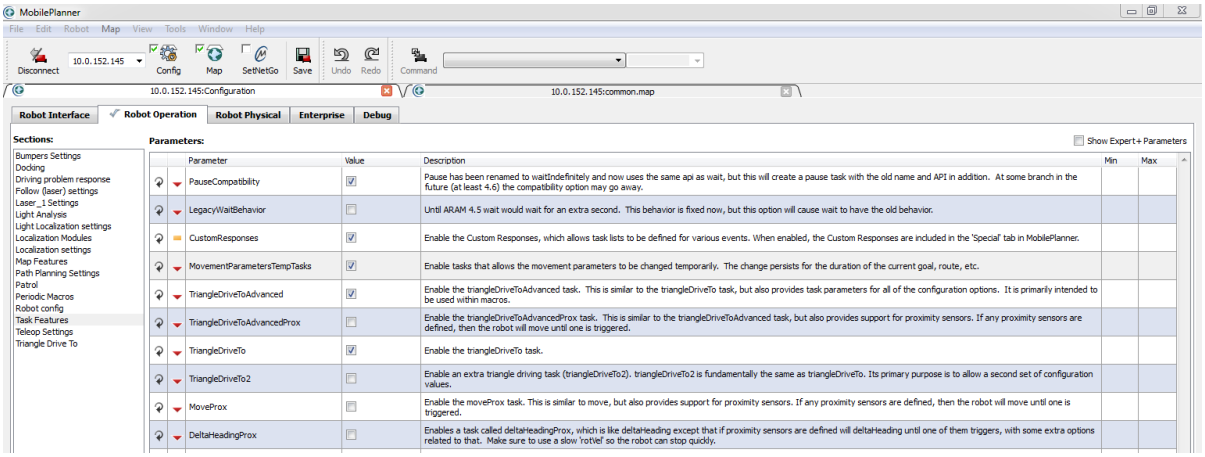
Here is what this movement looks like when the tasks are placed in a macro, “GoToTriangle_Move”. This macro is placed after a goal in MobilePlanner:



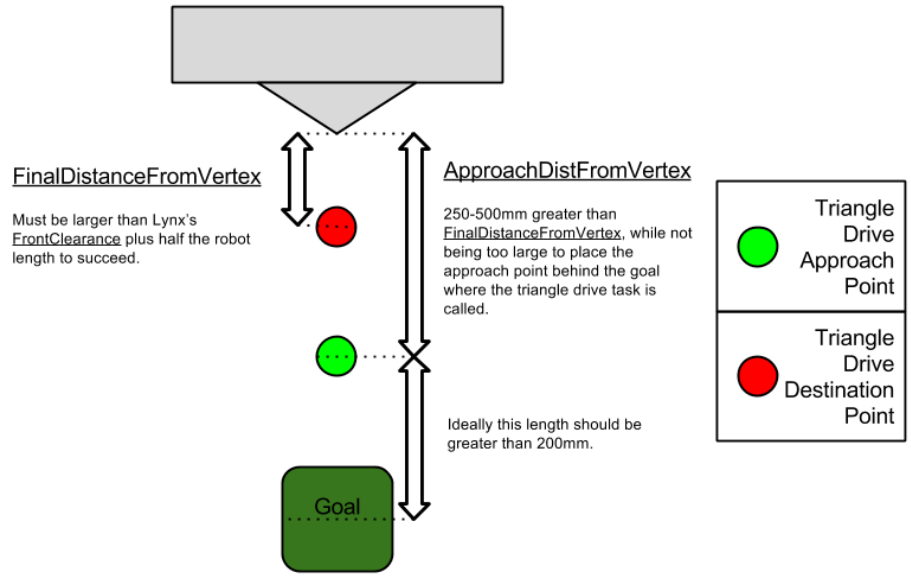
Programming Instructions

The TriangleDriveTo and TriangleDriveToAdvanced tasks are enabled by default. As seen in the image below, they can be enabled or disabled from MobilePlanner; navigate to Robot Configuration>Robot Operation > Task Features. Here are separate check boxes for the two tasks, TriangleDriveTo and

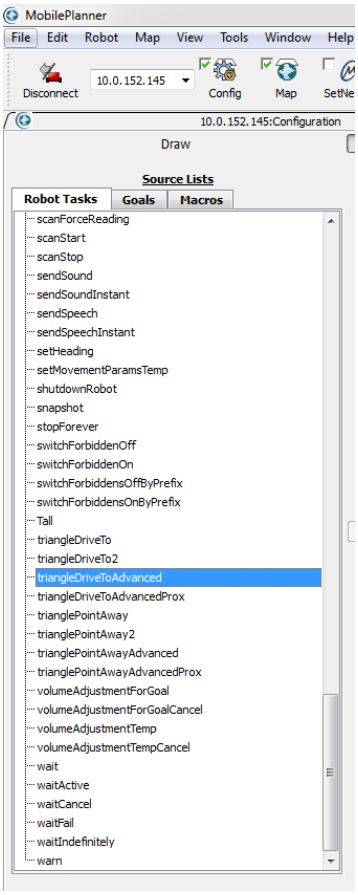
TriangleDriveToAdvanced.



Configuring the task



Once these items are enabled, the two triangle drive methods can be found within the map tab under Build, in Source Lists, on the Robot Tasks tab and then tasks.

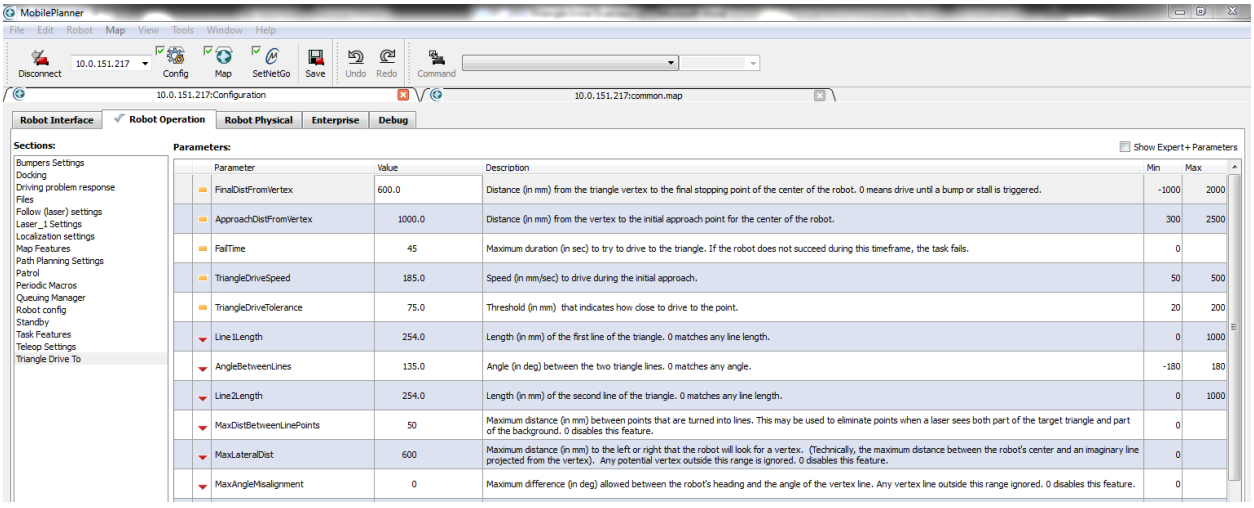


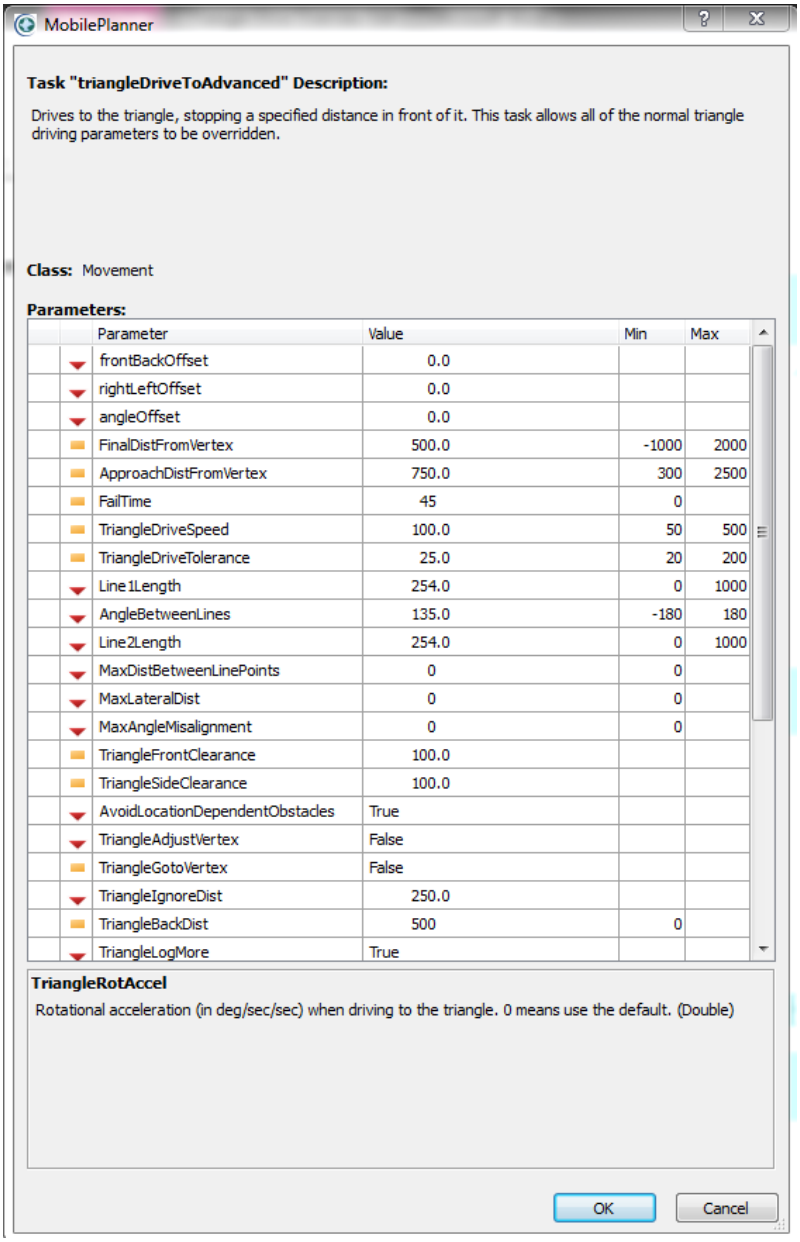
These tasks can be added to a macro or to a goal in the Editable List pane of the Build menu. The TriangleDriveTo task is coupled with the configuration found in the robot configuration, in the Robot Operation tab under the section called 'Triangle Drive To'. The advantage to using the triangleDriveTo task is that the user may change the behavior of the task wherever it is used in the robot's map. Alternatively, the TriangleDriveToAdvanced task's parameters are not dictated by the 'Triangle Drive To' configuration section. Instead, the advanced task is configured separately in the map each time it is used, allowing for different settings for different locations, for example. Each of the configurable parameters are explained in the help text at the bottom of the task. Some of the more complex parameters are explained in detail below:

- MaxLateralDist (at a value other than 0) varies the allowable distance between the robot's center point and the line in space which is an extension of the vertex of the triangle. This will prevent finding erroneous triangles, but if set too restrictively may also prevent you from finding the triangle of interest. A good default for this parameter is 1000 millimeters.
- MaxAngleMisalignment (when set to a value other than 0) will disallow triangles where the angular difference between the line extended from the vertex and the heading of the robot is larger than the set value. A good default for this parameter is 10 degrees.
- ApproachDistFromVertex is the distance between the vertex of the triangle to the initial approch point for the robot. This is a point where the robot will manuver to allign with the triangle before it drives to the final desired point. When performing a triangle drive task, this

point is shown in green in MobilePlanner. This parameter must be 250-500mm greater than the value for FinalDistanceFromVertex, while not being too large as to place the point beyond the goal where the TriangleDriveTo task is called.

- TriangleRotVelMax and TriangleTransVelMax can be changed to yield slight improvements in accuracy at the cost of speed. Maximum rotational speed can also be modified to increase accuracy. While these values have no default minimum, a value of 100 mm/s and 20 degrees/s will make the task as accurate as possible but very slow. Extremely low values for rotvelmax, rotaccel, and rotdecel will result in undesirable movements.
- FinalDistanceFromVertex speaks to the final location of the robot after completing the triangle drive movement. This value is very dependent on the application in which the triangle is used, as it changes the final position of the robot relative to the triangle vertex. Your front clearance parameters must be large enough to allow the robot to move this close to the triangle object. This parameter is in millimeters.
- AngleBetweenLines is the angle of the triangle's vertex. Outward facing triangles, where the vertex points towards the robot are measured as positive values, where inward facing triangles are negative.




MobilePlanner
Task "triangleDriveToAdvanced" Description:
Drives to the triangle, stopping a specified distance in front of it. This task allows all of the normal triangle driving parameters to be overridden.
Class: Movement
Parameters:
Parameter
Value
Min
Max
frontBackOffset 0.0
rightLeftOffset 0.0
angleOffset 0.0
FinalDistFromVertex 500.0 -1000 2000
ApproachDistFromVertex 750.0 300 2500
FailTime 45 0
TriangleDriveSpeed 100.0 50 500
TriangleDriveTolerance 25.0 20 200
Line1Length 254.0 0 1000
AngleBetweenLines 135.0 -180 180
Line2Length 254.0 0 1000
MaxDistBetweenLinePoints 0 0
MaxLateralDist 0 0
MaxAngleMisalignment 0 0
TriangleFrontClearance 100.0
TriangleSideClearance 100.0
AvoidLocationDependentObstacles True
TriangleAdjustVertex False
TriangleGotoVertex False
TriangleIgnoreDist 250.0
TriangleBackDist 500 0
TriangleLogMore True
TriangleRotAccel
Rotational acceleration (in deg/sec/sec) when driving to the triangle. 0 means use the default. (Double)
OK
Cancel