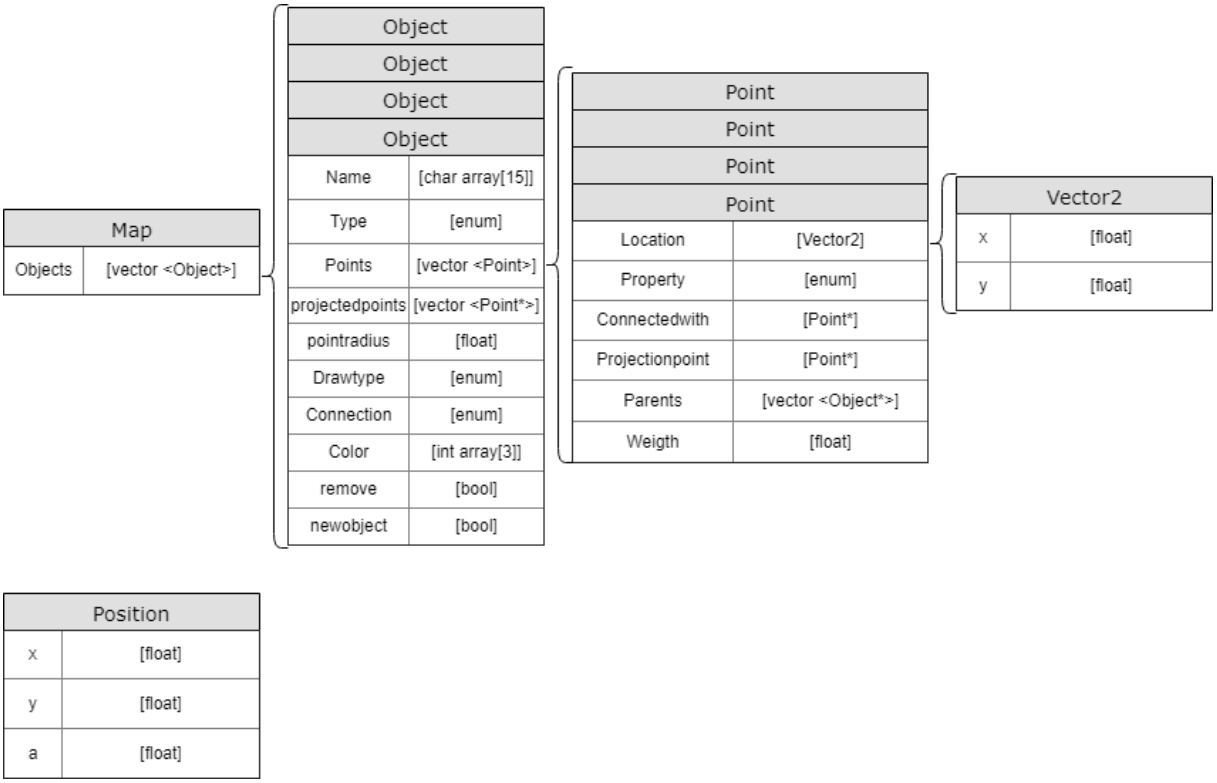
Data structures used by the perceptor



Map struct		
Variable	**Type**	**Usage**
Objects	[Vector <Object>]	To define the map. The map exists of multiple objects which can be a different type.
Function	**I/O**	**Description**
Transform	[Map| x| y| a] > [Map]	Transform all the object in the map to a new location and orientation. First rotate then translate.
Scale	[Map| x| y] > [Map]	Scale all the objects in the map with a factor x and y in the x and y direction respectively.
Print	[Map] > [-]	Print the names of all the objects in the console, used for debugging.
Printparents	[Map] > [-]	Print all the points and their parents from all the objects in the in the console, used for debugging.
Printprojectionlinks	[Map] > [-]	Print all the objects in the map with their points and projection points in the console, used for debugging.
removeobjects	[Map] > [Map]	Reorder the vector of objects so all the objects which have the remove flag set are at the

		end of the vector and then remove these objects from the vector.
setobjectsold	[Map] > [Map]	This function sets all the newobject variables to false which is used to distinguish between the data from the objects from the new scan and the objects already in the local map.

Object struct		
Variable	**Type**	**Usage**
Name	[char array [15]]	Visualization: Identification
Type	[enum]	Identification used to separate different objects [wall | door | test | origin | robot | dynamicobstacle | staticobstacle | safeDis | destination | projections | cabinet | start | path | node | nodes | roomwall]
points	[vector <Point>]	To define objects which consist of multiple points. Walls and doors consist of 2, other objects such as the origin, the robot, arrows and so on can consist of more points.
projectedpoints	[vector <Point*>]	Used to identify possible doors and rooms.
pointradius	[float]	Visualization: define how large to draw the points.
Drawtype	[enum]	Visualization: define if the points have to be connected with lines [points | lines]
connection	[enum]	Visualization: define if the first and last point have to be connected with a line [open | closed]
Color	[int array [3]]	Visualisation: specify the color of the object
remove	[bool]	Map remove function: determine if the object will be removed when the next removeobjects is called.
newobject	[bool]	Do distinguish between the objects that are already in the localmap and the objects from the new scan.
Function	**I/O**	**Description**

anglefrom	[Object| Point] > [float]	Calculate the angle of an object relative to the input point.
angle	[Object] > [float]	Calculate the angle [0 PI] of an object relative to 0. Works only when the object has 2 points e.g. wall or door.
origionalangle	[Object] > [float]	Calculate the angle [-PI PI] of an object relative to 0. Works only when the object has 2 points e.g. wall or door.
length	[Object] > [float]	Calculate the length of an object. Works only when the object has 2 points e.g. wall or door.
middle	[Object] > [Vector2]	Calculate the middle position of an object.
smallestrelativeangle	[Object| Object] > [float]	Calculate the smallest angle between 2 objects which both have 2 points.
averageperpendiculardistance	[Object| Object] > [float]	Calculate the average perpendicular distance between 2 objects which both have 2 points.
projection	[Object| point/Vector2] > [Vector2]	Calculate the projection of the input point / vector onto the object.
Constrained projection	[Object| Vector2] > [Vector2]	Calculate the projection of the input vector onto the object. Return zero if the projection is not on the object.
anglebetween	[Object| Object] > [float]	Calculate the angle between 2 objects measured from 1 direction.
gapdistance	[Object| Object] > [float]	Calculate the gap distance between 2 objects which both have 2 points. This is used when 2 objects are approximately parallel and have a small average perpendicular distance.
connectedto	[Object| Object] > [bool]	Determine if the two input objects have overlapping points and are thus connected to each other.
transform	[Object| x| y| a] > [Object]	This rotates and then translates all the points in this object with angle and an offset x and y.

scale	[Object| x| y] > [Object]	This scales all the points in this object with factor x and y in the x and y direction respectively.

Point struct		
Variable	**Type**	**Usage**
Location	[Vector2]	To describe the location of the point in a XY frame
Property	[enum]	To describe the property of a point [floating | convex | concave | connected | projected]
Connectedwith	[vector <Point*>]	To specify to which points it is connected
Projectionpoint	[Point*]	A reference to the projection of this point which is a new point.
Parents	[vector <Object*>]	References to all objects which have a point at this location
Weight	[float]	A weight to describe how certain a point is in the correct location
Function	**I/O**	**Description**
sameparentobject	[Point| Point] > [bool]	To check if two points have the same parent object.
connectedpoint	[Point| Point] > [bool]	To check if the two input points are connected to each other.

Vector2 struct		
Variable	**Type**	**Usage**
x	[float]	To store the x component of the vector.
y	[float]	To store the y component of the vector.
Function	**I/O**	**Description**
Operators | + | - |	[Vector2| Vector2] > [Vector2]	To add or subtract 2 vectors.
Operators | / | * |	[Vector2| float] > [Vector2]	To multiply or divide a vector with a number.
Operator | == |	[Vector2| Vector2] > [bool]	Determine if 2 vectors are the same.
Distance	[Vector2] > [float]	Calculate the Distance between 2 vectors.
Length	[Vector2] > [float]	Calculate the length of the input vector.
Angle	[Vector2] > [float]	Calculate the angle of the input vector relative to 0.

dot	[Vector2| Vector2] > [float]	Calculate the dot product of 2 vectors.
unit	[Vector2] > [Vector2]	Calculate the unit vector of a given vector.
transform	[Vector2| x| y| a] > [Vector2]	Rotate the input vector with angle a and then translate this vector with x and y.
scale	[Vector2| x| y] > [Vector2]	Scale the input vector in the x and y direction

Position struct		
Variable	**Type**	**Usage**
x	[float]	To store the x component of the position.
y	[float]	To store the y component of the position.
a	[float]	To store the angle component to the position.
Function	**I/O**	**Description**
Operators | + | - |	[Position] > [Position]	To do simple calculations with a position variable.