Cowbot Documentation: Pathing.Pathing.py

TODO: Remove the arguments in “follow_x(current_state)”, since they are methods of the path object anyway - they can access `self.current_state` directly.
This module is the parent to all of the path-type classes.

1 class `GroundPath`

1.1 `self.length`

Float. The length of the path.

1.2 `self.time_to_traverse`

Float. The estimated time needed to follow the path.

1.3 `self.waypoints`

A list of `Vec3` waypoints. Not currently used.

1.4 `self.current_state`

`CarState`. The current state of our car.

1.5 `self.input()`

Returns the final controller input to follow a `GroundPath`. `input()` achieves this by calling the appropriate “follow_x” method.

1.6 `self.follow_arc(current_state)`

Called by `input()` to follow a circular arc.

1.7 `self.follow_line(current_state)`

Called by `input()` to follow a line segment.

1.8 `self.follow_curve(current_state)`

Called by `input()` to follow a general curve. Currently does nothing, plans are to give a curve as a sequence of points and this methods turns between them.

1.9 `self.follow_waypoint(current_state)`

Called by `input()` to drive towards a waypoint. This just uses `GroundTurn`, and should only be used for fairly small angles.

2 class `PathPiece`

TODO: Direction should be +1 for CW and -1 for CCW.

2.1 `self.shape`

String. The tag that tells `GroundPath.input()` which type of path we’re trying to follow. Currently supports “Line”, “Arc”, “Waypoint”, and “Curve”.

2.2 `self.start`

`Vec3`. The starting position of the path.

2.3 `self.end`

`Vec3`. The ending position of the path.

2.4 `self.start_tangent`

Nonzero `Vec3`. The tangent vector to the path at `self.start`.

2.5 `self.end_tangent`

Nonzero `Vec3`. The tangent vector to the path at `self.end`.

2.6 `self.direction`

+1 or -1. The direction to turn around a circular arc - +1 for CW, -1 for CCW.

2.7 `self.waypoint`

`Vec3`. The next point to drive towards in a path specified via waypoints.