Cowbot Documentation: Miscellaneous.py

TODO: Move Vec3 related functions to CowbotVector?

1 cap_magnitude(x, magnitude = 1)

Returns `sgn(x)*max(abs(x),magnitude)`. The output and both arguments are floats.

2 rotate_to_range(theta, interval)

2.1 theta

Float.

2.2 interval

List or tuple of two floats. `interval[0]` should be strictly less than `interval[1]`.
This function is primarily used to find an angle equivalent to `theta` but is between $-\pi$ and π. Add and subtract multiples of the length of `interval` to `theta` until `theta` is between `interval[0]` and `interval[1]`. Return the final theta.

3 car_coordinates_2d(current_state, direction)

Throws out the z-component and rotates the x and y components to the car-basis. Returns a `Vec3` with zero z-component.

3.1 current_state

`CarState`. The state of the car we're working with.

3.2 direction

`Vec3`. The direction vector from the car to the target.

4 angles_are_close(angle1, angle2, epsilon)

Returns a Boolean, True if `angle1` (float) and `angle2` (float) are within `epsilon` (float) of each other.

5 left_or_right(current_state, target_pos)

Check if the car should turn left or right to face towards the target. Returns $+1$ for right and -1 for left.

5.1 current_state

`CarState`. The current state of the car.

5.2 target_pos

`Vec3`. The point on the field we are trying to point towards.

6 rot_to_mat3(rot

Takes `rot`, an `Orientation` object, and returns the corresponding RLU `mat3` object.

7 pyr_to_matrix(pyr)

TODO: Change all "pyr" notation to "ypr" to match the convention used by RL.
Takes an Euler angle orientation (pitch, yaw, roll) and returns the orientation matrix `[front, left, up]`.

8 Vec3_to_Vector3(vector)

Takes a `Vec3` (CowBot) and returns the corresponding `Vector3` (framework).

9 Vec3_to_vec3(vector)

Takes a `Vec3` (CowBot) and returns the corresponding `vec3` (RLU).

10 vec3_to_Vec3(vector)

Takes a `vec3` (RLU) and returns the corresponding `Vec3` (CowBot).

11 is_in_map(location)

Takes `location` (Vec3) and returns a Boolean, True if `location` is inside the game map. Rudimentary for now, can definitly be improved over time.

12 angle_to(target, start, initial_angle)

Takes a target location `target` (`Vec3`), a starting location `start` (`Vec3`), and a starting yaw `initial_angle` (float between $-\pi$ and π).
Returns the angle between the initial yaw and the angle needed to face the target.

13 min_radius(speed)

Returns the minimum radius (float) possible given an input speed (float). Comes from Chip's notes on ground handling. Data was taken for an Octane, turns in a plank body will likely be slightly wider.

14 max_speed(radius)

Returns the maximum speed (float) possible given an input radius (float). Comes from Chip's notes on ground handling. Data was taken for an Octane, turns in a plank body will likely be slightly wider.