Cowbot Documentation: Mechanics.py

1 class `PersistentMechanics`

A container to store objects that must be preserved across frames.
Takes no arguments, and is called once in the initialization of CowBot. Shouldn't be used elsewhere.

2 class `Mechanic`

A class to hold a mechanic that needs to be preserved across frames.
Initialization takes no arguments, and is initialized once per mechanic when CowBot is initialized. As of now this shouldn't be used elsewhere, but that could change if the the overall structure of the code changes.

2.1 `self.check`

A Boolean that is `True` only on frames where the bot is performing the mechanic.

2.2 `self.action`

The actual mechanic object. Both current use cases are RLU objects (Aerial and AerialTurn).

3 class `AirDodge`

The mechanic to dodge in the air. Initialization takes two arguments: `direction` and `jumped_last_frame`.

3.1 `self.direction`

Vec3 relative to car coordinates. The intended direction to dodge.

3.2 `self.jumped_last_frame`

Boolean. True if the bot sent a "jump" input on the prior frame.

3.3 `self.input()`

Returns an empty input if `jumped_last_frame` is True. This ensures an attempted dodge doesn't just continue holding the jump button after takeoff. Otherwise returns a dodge in `direction` relative to the car: `jump` is set to 1, `yaw` is set to `direction.y`, and `pitch` is set to `-direction.x`. This accounts for the +x-direction being forward and the +y-direction being to the left.

4 class `JumpTurn`

A mechanic to jump and turn while in the air. Very coarse, used primarily for turning while sitting still. This use case will be deprecated as pathing improves. 50/50s are potentially another use case, turning in the air to dodge from a better angle.

4.1 `self.current_state`

The current state of the car, usually gotten from `game_into.me`. This class uses z-position and `wheel_contact`.

4.2 `self.jump_height`

Float. How high the car should be (in uu from the ground) before we stop holding jump. If set to zero, the bot inputs a one-frame jump.

4.3 `self.turn_direction`

+1 or -1. Whether to turn clockwise (+1) or counterclockwise (-1).

4.4 `self.input()`

Returns the controller input for the mechanic. If `jump_height` is zero, then `controller_input.jump` is set to one only while the car is on the ground. Otherwise jump is held until the car reaches `jump_height` (this can easily be improved later on, if I want to use it). Hold yaw to turn in the specified direction while in the air (basic control loop to reach a specified orientation might also be good here eventually).

5 class `QuickTurn`

A powerslide turn for turning a small amount quickly. Designed to be used in fast dodges. Possibly deprecated there in light of cancelled diagonal dodges. Maybe still useful for turning on the ball to shoot or dribble.

5.1 `self.direction`

+1 or -1. The direction to turn: +1 for clockwise (right), -1 for counterclockwise (left).

5.2 `self.boost`

0 or 1. Whether or not to use boost (1) or not use boost (0).

5.3 `self.input()`

Returns the controller input for the mechanic. Hold throttle always. Hold handbrake always. Boost if `self.boost == 1`. Set steer to `self.direction`.

6 class `CancelledFastDodge`

A cancelled forward-diagonal dodge to accelerate quickly. Used in kickoffs and eventually will be used for general purpose large-scale repositioning. We can choose left or right. Call this mechanic once the car is in the air and ready to dodge - it will dodge as long as it's high enough off the ground.

6.1 `self.double_jumped`

Boolean. True if the car is in the air and has already double jumped or dodged.

6.2 `self.dodge_direction`

+1 or -1. +1 for a right dodge, -1 for a left dodge.

6.3 `self.current_state`

`CarState` object. Usually will be set to `game_info.me`. This class actually only uses `current_state.jumped_last_frame`.

6.4 `self.input()`

Returns the controller input for the mechanic. Once the car is above 50uu, it will dodge forward+`direction`. Until then, it boosts. After the dodge, it continues boosting (check for max speed later) and pulls back on pitch to cancel the forward portion of the dodge.