Driving Simulator

Car Classes Documentation

**AbstractCar**

Description

The AbstractCar class inherits the pygame Sprite class and acts as a base for PlayerCar and BotCar classes to inherit. It is not meant to be instantiated on its own.

Methods

\_\_init\_\_

Parameters: startPos (tuple containing x and y position), startAngle (starting angle of car, default 0)

Description: Creates the image, rect, and mask for car. An angle of 0 degrees points right, 90 points down, 180 points left, 270 points up, etc. Also contains adjustable variables for the car’s length, max speed, turn speed, and acceleration.

setImage

Description: Transforms the car’s image from the original based on length and angle of the car. This ensures the image doesn’t get warped from repeated transformations.

stop

Description: Stops the car and sets the stopped flag to True. This function is meant to be called when the car crashes and can no longer move.

accelerate

Description: Increases the car’s speed until it’s max speed (defined in \_\_init\_\_ method).

decelerate

Description: Decreases the car’s speed until reversing at half it’s max speed (defined in \_\_init\_\_ method).

brake

Description: Brings the car’s speed to 0, accelerating or decelerating as necessary (not using the accelerate nor decelerate methods)

turn

Parameters: dir (string “left” or “right”), scale (number to multiply turn speed by, for example 0.5 to turn at half speed, default 1)

Description: Turns the car left or right based on dir parameter, and updates image, mask, and rect.

move

Description: Updates the car’s position based on its current angle and velocity. Also keeps track of subX and subY, variables to keep track of decimal for position since pygame only uses integers for coordinates. Using only integer pixel positions isn’t accurate enough for movement. Currently has no adjustments for how quickly the game is running, so it should ideally be used at a near constant framerate.

getRect

Description: Returns the pygame rect object for the car’s sprite.

checkCollision

Description: Checks if the car has collided with anything in its collideGroups list. (This list can be added to using the setCollide method). If the car’s mask collides with any masks within any of the groups, it calls the stop method and changes the car’s image to an explosion and plays an explosion sound.

setCollide

Parameters: groups (a list of pygame groups that the car should collide with)

Description: Sets collideGroups, a list of pygame groups that the checkCollision method checks to see if the car has collided with anything.

isStopped

Description: Returns True if the car has collided with anything in the groups defined using setCollide, causing it to crash and stop, otherwise returns False.

getAngle

Description: Returns the car’s angle (in degrees, from [0, 360).

getSpeed

Description: Returns the car’s velocity

getPos

Description: Returns the position of the car’s center (A tuple in the format (x, y)).

setPos

Parameters: pos (a tuple containing the x and y positions)

Description: Moves the car’s center to the pos parameter.

setAngle

Parameters: angle

Description: Sets the car’s angle to the angle parameter and updates the car’s image.

**PlayerCar**

\_\_init\_\_

Parameters: startPos (tuple containing x and y position), startAngle (starting angle of car, default 0)

Description: Initializes AbstractCar class and sets the sound file and volume for car crashes.

player\_input

Description: Calls appropriate movement methods based on player inputs from pygame.

autoTurn

Description: Turns the car toward the closest of the 8 directions (up, down, left, right, and 4 in between)

update

Description: Checks for collisions, updates movement based on input, and moves the car accordingly.

**BotCar**

\_\_init\_\_

Parameters: startPos (tuple containing x and y position), startAngle (starting angle of car, default 0)

Description: Initializes AbstractCar class and sets the sound file and volume for car crashes. Also contains adjustable member variable for targetBuffer (distance in pixels the car’s center must be from a target to move on to next target).

bot\_move

Description: Moves the car toward the next target by adjusting angle and speed (using turn, accelerate, decelerate, and stop methods)

setTarget

Parameters: target (tuple containing two elements, the first element being a tuple of x and y coordinates, the second element being True or False to determine if the car should stop at the target or not). For example, to stop at (300, 400), the target parameter is ((300, 400), True).

Description: Sets the bot car’s next target to the passed target (overwriting the current target).

addTargets

Parameters: targets (list of targets, where each target is format ((x, y), bool) as described above for setTarget).

Description: Adds a passed list of targets to an internally stored list of targets to reach. The bot car will attempt to reach each target in order, moving on to the next one when within the buffer distance defined in \_\_init\_\_.

newTarget

Description: Used when determining whether the car should move on to its next target or not. Returns True when within buffer distance defined in \_\_init\_\_, returns False otherwise.

queuedTargets

Description: Returns the number of targets the bot car has left to hit (not including the one it is currently targeting)

checkStopSign

Parameters: stopSigns

Description:

update:

Description: Checks for collisions, updates movement based on target, moves the car, and updates the target.