Driverless Car: Class Diagrams

The system will consider two packages. Vehicles and Maps. The driverless car will have an internal system with component classes of Movement, Point and Direction. The maps package will have a Road class. Each road is made up of multiple Road Segments.

The Vehicle Class:

The vehicle as a class has the following attributes and methods:

Vehicle
reg: string make : string model: string drive: Movement
locate_road() detect_obstacle() get_speed_limit(road) get_stopping_distance() steer() accelerate() brake()

The reg, make and model will describe the vehicle, while the drive attribute will hold the data concerning the position and movement of the vehicle at a point in time.

The Movement Class:

Movement
location : Point direction: Direction speed: float acceleration: float time: integer

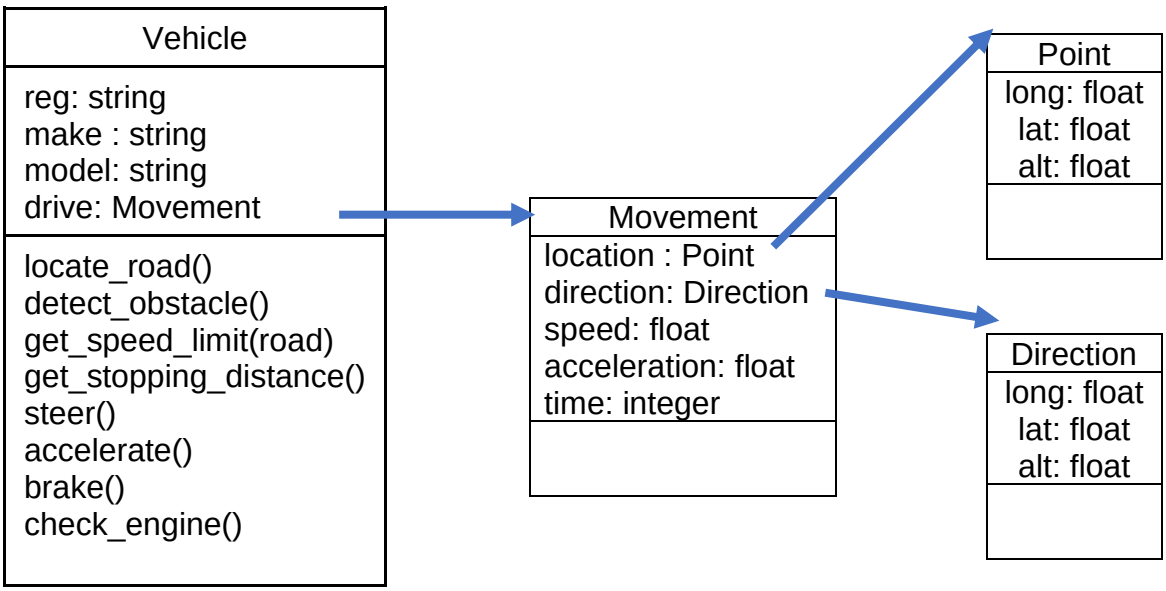
The Movement class introduces the Point and Direction classes.

The Point and Direction Classes:

Point
long: float lat: float alt: float

The Point class represents coordinates for longitude, latitude and altitude and would be used to locate the vehicle's potion on the map. The Direction class represents the 3-dimensional vector, used to show the direction of travel. While this is a vector rather than a coordinate, the attributes are the same as the Point class, meaning that the Direction class can inherit the Point class attributes.

Static View:



The Road Class:

In order for the autonomous vehicle to drive on the road, the vehicle must interact with the road network and each road will be an object of the Road class as shown below.

Road
name: string category : string segments: Road_Segment

Each Road is made up by multiple road segments, of the Road-Segment class. The road segment is defined by a start and end locations from which the distance and gradient are determined. Each road segment is modelled as a straight section of road.

Road_Segment
index: integer location_start : Point location_end:Point direction:Direction distance: int gradient: int category: string
get_distance() get_category(road) get_direction() get_gradient()

As each road is made up of many road segments, this is represented below.

The roads and road segments will be stored as part of a road network database. Route planning software can be used to determine a route which can be stored as a list of

road segments with the end of each segment representing a checkpoint in the journey. A route has the same attributes as a road as it is made up of road segments. As such a route could be considered a special type of road.

References

Reddy, P. P. (2019) Driverless Car: Software Modelling and Design using Python and Tensorflow.

Zhou, Z. Q. & Sun, L. (2019) Metamorphic testing of driverless cars. Commun. ACM 62(3): 61–67. DOI: https://doi.org/10.1145/3241979.

Joque, J. (2016) The Invention of the Object: Object Orientation and the Philosophical Development of Programming Languages. *Philosophy & Technology* (29):335 -356.

Rumbaugh, J., Jacobson, I. & Booch, G. (2005) *The Unified Moddeling Language Reference Manual. Second Edition.* Boston: Addison-Wesley

Phillips D. (2018) *Python 3 Object-Oriented Programming*. Third Edition. Birmingham: Packt Publishing Ltd.

Glinz, M. (2000) 'Problems and deficiencies of UML as a requirements specification language', *Tenth International Workshop on Software Specification and Design.* San Diego, 07 November 2000. San Diego: IEEE. 11-22.

Babaei, P et al. (2023) Perception System Architecture for Self-Driving Vehicles: A Cyber- Physical Systems Framework. Available from: https://www.researchsquare.com/article/rs-3777591/v1 [Accessed 25 April 2024]