Integration phase 2

Kiarash jamshidi

Exploiting video games to test AV

- The second one is adding the amount also to mutation. It will find the frontier for the operation as well. In every test, the road will be the same.

Fix

Generating road

Mutate

Operation

How many way integration type 2 has ?

- So far, we have two different way for finding the

First way

- In this part it attached the amount to the Member of the list

```
class BeamNGMember(Member):
    def __init__(self, control_nodes: Tuple4F, sample_nodes: Tuple4F, num_spline_nodes: int,
                 road_bbox: RoadBoundingBox, amount: int, type_operation: string):
```

How to understand the true measure of amount

- After passing the member in the NSGA2 class in the function pre_evaluate_members(invalid_ind) . We try to find the exact amount of the operation which the system is work properly with.

we came to beamng_nvidia_run

- In this part, same as the first type of the integration. i defined the first type of the operation in OPERATION CLASS and also the second type in thee BREWER class.
- In evaluate function at first we get the amount and the type of the operation from the first member.
- Then we go to evaluate the amount in run_simulation

What is the initial amount of operation

- In the Config, some variable defined for the different operations

```
self.FOG_DENSITY = 1
self.WET_FOAM = 40
self.NUMBER_OF_DROP_RAIN = 30000
self.WET_RIPPLE = 30000
self.NUMBER_BUMP = 400
self.ILLUMINATION_AMOUNT = 0.7
```

- This amounts are maximum for all the operation (cause of the failure)
- The failure means , with this amount for operation, the vehicle will go out of the boundary.

The program run multipole times

- With the road in the first member and also the amount for the operation the simulation will start.
- If the failure happen it will decrease the amount of the operation and will run it again
- If the success happens, it will stop the iteration. Then save the amount and the other information in the data folder.
- the amount has threshold , if amount arrive at threshold without success . It will raise the An Error.

sequenc of making new mutation

Main
main_beamng
BeamNGProblem
nsga2
beamng_problem
BeamNGIndividual
Member
BeamNGMember
beamng_nvidia_runner
operation
BeamNGBrewer
SimulationDataCollector
strat(type operation)
problem = initial(.., type_operation)
main(problem)
problem.pre_evaluate_members(invalid_ind)
pre_evaluate_members()
evaluate()
evaluate(members)
operation_func()
bring_up()
collect_current_data(), get_simulation_data()
end_iteration(), close()

The second way

- The only different is the mutant function will useful this Time
- At first in the member it will have a one road with the different amount of the operation

How it will work

- The beamng_nvidia_run has the same approach for the finding the right solution .
- It start from first member of the list and if the car get out it will start with the second member which has the same road and less amount of operation.
- It will iterate until it find the the success and save the data