Notes on Adv.Mod.Op - Project

STEP 1:

1. Constant speed is not realistic. Many cases, such as changing direction or accelerating, shall be taken into consideration

1. “Average speed meant to account for high-speed travel, low-speed turns, and acceleration/deceleration times”

2. Battery Size: Assuming it as a constant is not realistic. It decreases throughout time depending on the usage. Possible Solution. For n times is constant and then starts to decrease.

3. “Compute\_Time”: It should depend on the length of the travel….”Random.uniform(0,1)” is added independently of the length…it should proportional though. SOLUTION: It might depend on the path the tugger has to do

1. “#random.uniform(0,1) to account for possible stops due to the presence of obstacles on the vehicle path (e.g., people, unit loads not placed correctly, ..)”

4. Compute\_energy: Energy does not depend solely on time but also on other variables such as loading weight or speed (acceleration, deceleration)

5. Class Train:

1. \_\_init\_\_

1. The position of the tuggers at the beginning of the simulation cannot be equal…tuggers cannot lay on each other. Could influence the consumption the day after, for instance. That is why it should be considered, for both modelling purposes and reality issues.
2. Next stop If we take into account acceleration and deceleration, if no loads in the next stop have to be charged there is no point in stopping there (saving time, energy, etc.). Thus it is not mandatory to stop at line 1 everytime

2. check\_charge: “NA” [Domanda: Non ha senso che la selezioni randomly…]

3. move

1. **“if self.model.schedule\_lines.agents[self.next\_line].UL\_in\_buffer >= 1”** is wrong. It does not take into consideration the possibility of the lines to create a new UL.
2. **Self.load < self.capacity** -> It does not take into consideration the loading’s limit of the tugger. ULs weights differ!
3. **self.remaining\_energy -= compute\_energy(loading\_time**) -> Energy consumption depends on the activity (loading in this case) and not solely on time.
4. **self.load += 1** -> Why can’t the tugger take on more ULs than one if it has space?

4.charging:

* Do batteries charge linearly? No, charging time depends on battery load.

5.step:

* Modelsystem is in integers whereas tasktime is in floats…we are loosing time.

6. ChargingStation:

a) Need to convert the steps in a smaller time unit to avoid loss

6. General: MOVEMENT IS INSTANTANEOUS. CAN WE ACHIEVE A CONTINUOUS MOVEMENT ON THE GRID?

STEP 2:

1) If the tugger does not have to pass through line 1, it could move diagonally. Thus, the rectilinear distance would be wrong.

Problems and (POSSIBLE) SOLUTIONS:

|  |  |
| --- | --- |
| PROBLEMS | SOLUTIONS |
| Tugger train loads the units few seconds before they have been created (for the first round). | Changing system time from 0 to 1 and rearranged step() function in the FactoryModel: Before it was updating the line’s time before the model’s one! |
| The tugger does not move, it says fixed at the 1st station, i.e., station[0] | Within the Class Train, the function move() was not updating the Tugger’s position, thus the tugger was moving only between station[0] and the warehouse. With the code added, the positions are updated everytime. |
| Distance is miss-calculated between station 3 and 4 | How to compute the distance has been modified in order to consider the special case regarding the movement between the third and fourth stations. Note: max\_x has been added as parameter to create a more flexible code |
| Speed not constant, it depends on the weight of the tugger | Creare funzione in utils |
|  |  |
|  |  |
|  |  |
|  |  |