2024

# Novelty Report

U7701752

YANRUI MA

# Battery and Charging Station

## 1. Description

I implemented a battery function for the robot and a charging station function node.

We could open the battery function and charging station during the stimulation by operating battery_simulation.launch.py.

The logic is simple, we assume our robot is powered by a battery that will run out during the operation. We could guide the robot to a charging station or ignore it. If we unfortunately run out of power, we will stop immediately. We assume that there would be a staff to rescue us to charge us. But it needs extra time for staff to come here.

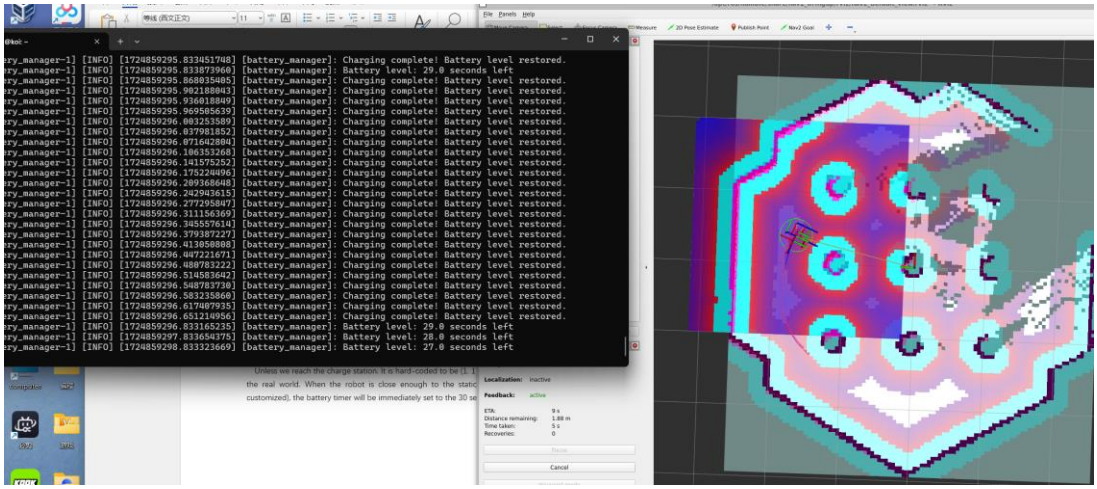
It is actually very close to the real-world situation in my mind. But to make it easier to implement, I have made a lot of compromises.

The battery is a timer, we could set a time for the robot, which defaults to 30 seconds, to be the battery time. After every 30 seconds, the robot needs to stop and call staff to help.

We set it to be 5 seconds (this can be customized) for staff to rescue the robot. After the rescue, the battery time will be set to 30 seconds again.

```
[battery_manager-1] [INFO] [1724859007.833826890] [battery_manager]: Battery level: 15.0 seconds left
[battery_manager-1] [INFO] [1724859008.833304310] [battery_manager]: Battery level: 14.0 seconds left
[battery_manager-1] [INFO] [1724859009.833563732] [battery_manager]: Battery level: 13.0 seconds left
[battery_manager-1] [INFO] [1724859010.833316854] [battery_manager]: Battery level: 12.0 seconds left
[battery_manager-1] [INFO] [1724859011.834150608] [battery_manager]: Battery level: 11.0 seconds left
[battery_manager-1] [INFO] [1724859012.833278108] [battery_manager]: Battery level: 10.0 seconds left
[battery_manager-1] [INFO] [1724859013.833651893] [battery_manager]: Battery level: 9.0 seconds left
[battery_manager-1] [INFO] [1724859014.833170852] [battery_manager]: Battery level: 8.0 seconds left
[battery_manager-1] [INFO] [1724859015.833205508] [battery_manager]: Battery level: 7.0 seconds left
[battery_manager-1] [INFO] [1724859016.833426262] [battery_manager]: Battery level: 6.0 seconds left
[battery_manager-1] [INFO] [1724859017.833590761] [battery_manager]: Battery level: 5.0 seconds left
[battery_manager-1] [INFO] [1724859018.833082429] [battery_manager]: Battery level: 4.0 seconds left
[battery_manager-1] [INFO] [1724859019.833109381] [battery_manager]: Battery level: 3.0 seconds left
[battery_manager-1] [INFO] [1724859020.833678623] [battery_manager]: Battery level: 2.0 seconds left
[battery_manager-1] [INFO] [1724859021.833169945] [battery_manager]: Battery level: 1.0 seconds left
[battery_manager-1] [INFO] [1724859022.833199632] [battery_manager]: Battery level: 0.0 seconds left
[battery_manager-1] [INFO] [1724859023.833245940] [battery_manager]: Battery depleted! Robot stopped and charging for 5
seconds...
[battery_manager-1] [INFO] [1724859028.839051275] [battery_manager]: Charging complete! Battery level restored.
[battery_manager-1] [INFO] [1724859028.840134596] [battery_manager]: Battery level: 29.0 seconds left
[battery_manager-1] [INFO] [1724859029.835110385] [battery_manager]: Battery level: 28.0 seconds left
[battery_manager-1] [INFO] [1724859030.833112337] [battery_manager]: Battery level: 27.0 seconds left
```

Unless we reach the charge station. It is hard-coded to be (1. 1). This position can be customized. There is a cylinder in the real world that I put there to pretend to be a charge station. However, the model itself has no connection to the code. When the robot gets close enough to the station (distance could be customized, it is 0.5 currently), the battery timer will be immediately set to 30 seconds.



## 2. How it connects to the real world

I think it is very common situation for an industry robot to consider the power management problem. I also searched about this and found the idea of an automatic charging station which inspired me to make a novelty like this. Although I made a lot of comparisons, the idea is very close to the real world. I have heard a lot about EVs running out of power on the road which should be a serious problem.

## 3. Feasibility

The innovation is based entirely on existing robot navigation systems and position-awareness capabilities and does not require the introduction of new sensors or movements. The robot decides whether or not to restore power by detecting its distance from the charging station as it approaches the station. The battery management node decides when to stop and when to recharge by listening to the robot's position information and navigation goals. This innovation leverages functionality that already exists in the robot and can therefore be easily implemented in the current ROS 2 and Gazebo environments.

## 4. Baseline test and solution

This part is simple for me. We used Nav2 as our baseline method which doesn't have any battery management function. So, it always goes straight to the goal until runs out of power. The assignment requires a 25 percent drop, but my function has high customization capability. If we set the staff helping time to 10 seconds, and the battery time to 30 seconds, the efficiency drop will be 33.3%.

In the future, I want it to be an optimization problem. If the robot finds it may not be able to get to its goal before running out of time, it will head charge station first.

Also, I may extend the punishment for no power to make this decision more likely to be a

good decision. Currently, staff rescue time is too short maybe even shorter than the time needed to go to the charge station.

## Conclusion

Through the design and implementation of this innovation point, we have successfully simulated the behavior of a robot after power depletion, demonstrating the importance of battery management in the real world. This innovation point performs significantly in reducing the efficiency of the robot's task completion and lays the foundation for further optimization and functionality extensions in the future.