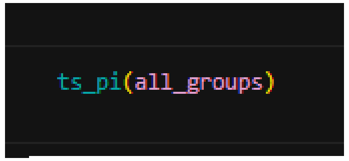
AR Security 12/6

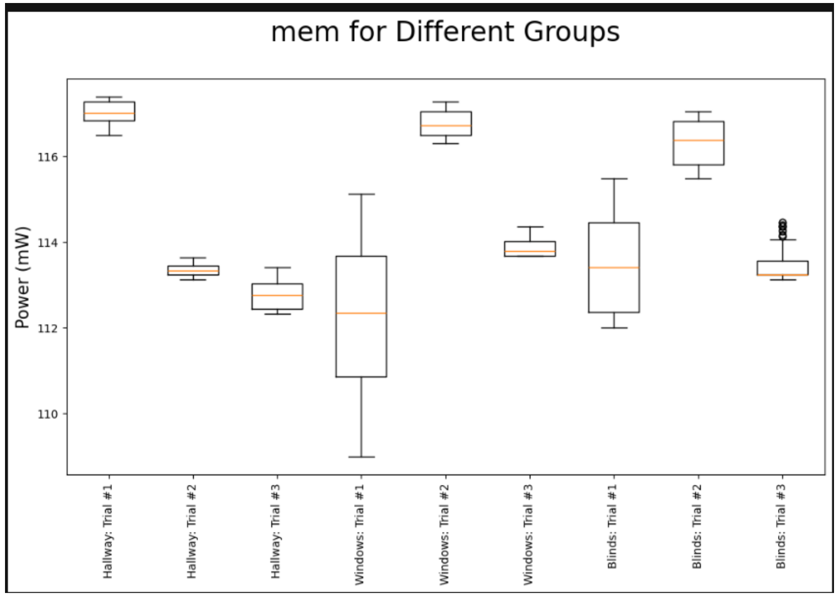
Allie Craddock & Casie Peng

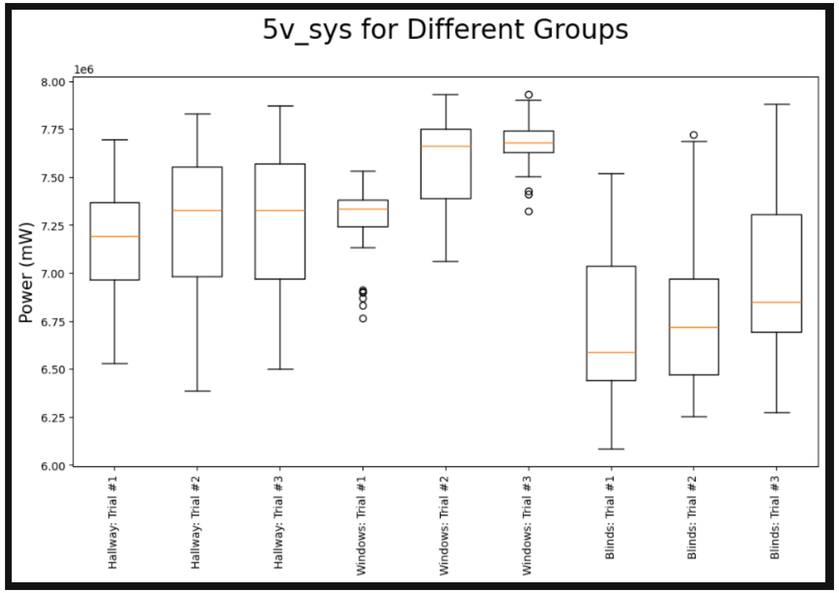
Data Analysis Code

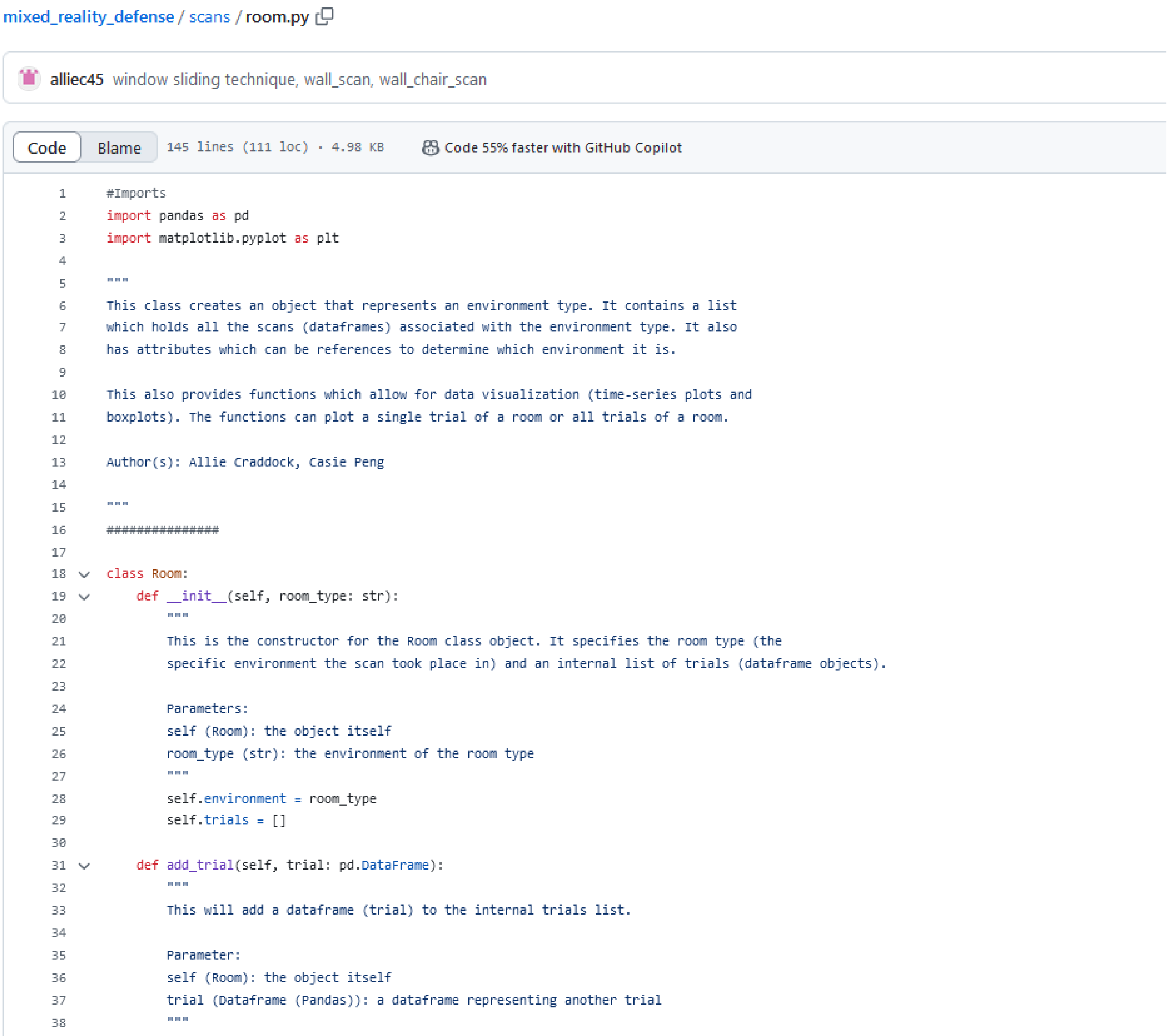
- Finished Generating Boxplots
- Finished method which generates all trials on subplots for all room types in a list, for all performance indicators
  - Main Jupyter Notebook got corrupted several times, had to fix that



```
ts_pi(all_groups)
```



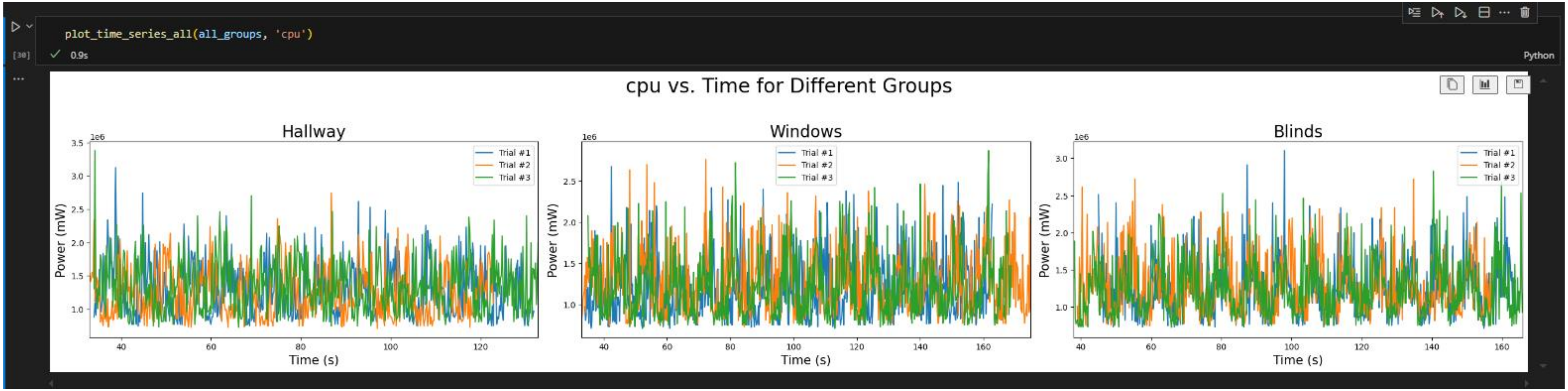



mixed_reality_defense / scans / room.py

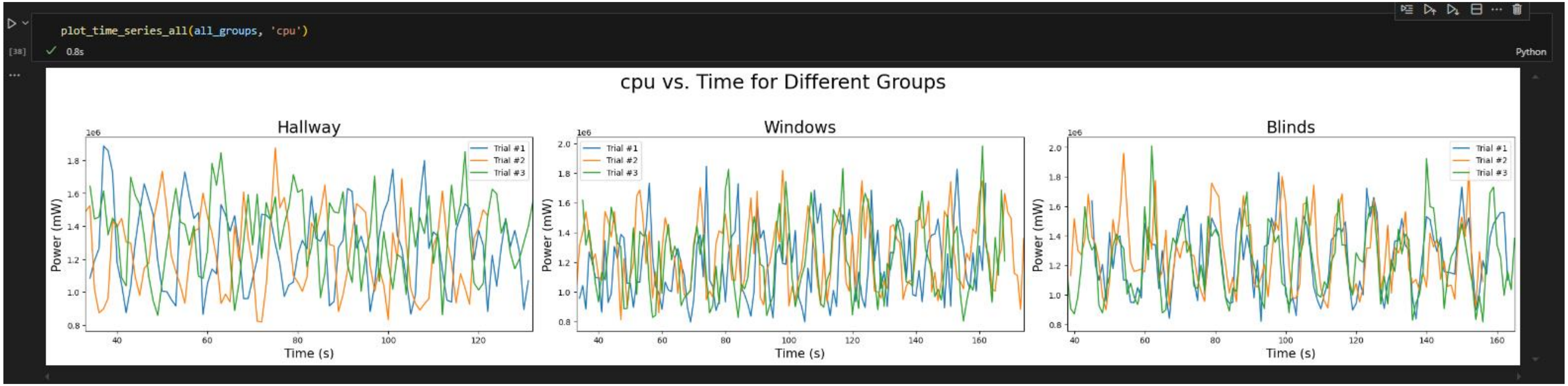
alliec45 window sliding technique, wall_scan, wall_chair_scan

145 lines (111 loc) · 4.98 KB

```python
#Imports
import pandas as pd
import matplotlib.pyplot as plt

"""
This class creates an object that represents an environment type. It contains a list
which holds all the scans (dataframes) associated with the environment type. It also
has attributes which can be references to determine which environment it is.

This also provides functions which allow for data visualization (time-series plots and
boxplots). The functions can plot a single trial of a room or all trials of a room.

Author(s): Allie Craddock, Casie Peng

"""
###############

class Room:
    def __init__(self, room_type: str):
        """
        This is the constructor for the Room class object. It specifies the room type (the
        specific environment the scan took place in) and an internal list of trials (dataframe objects).

        Parameters:
        self (Room): the object itself
        room_type (str): the environment of the room type
        """
        self.environment = room_type
        self.trials = []

    def add_trial(self, trial: pd.DataFrame):
        """
        This will add a dataframe (trial) to the internal trials list.

        Parameter:
        self (Room): the object itself
        trial (Dataframe (Pandas)): a dataframe representing another trial
        """
```



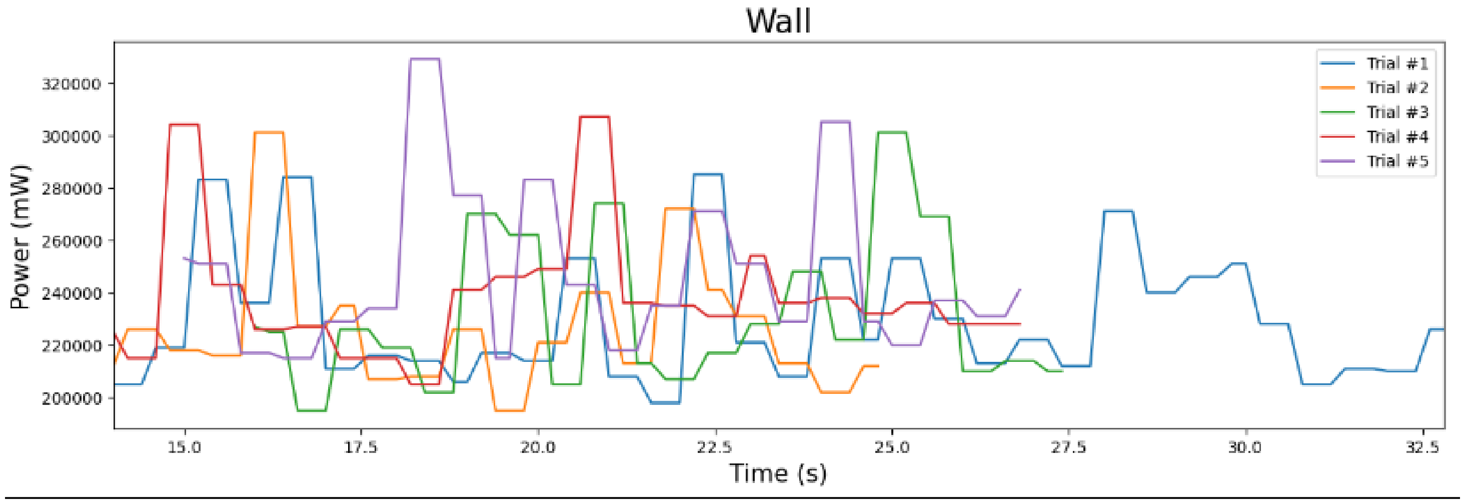
```
[30] plot_time_series_all(all_groups, 'cpu')
✓ 0.9s
```

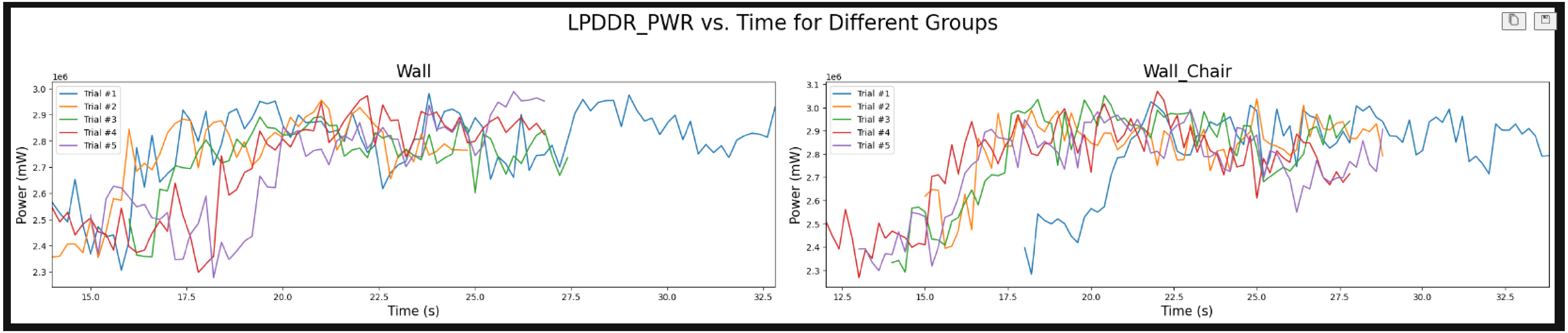


```
[38] plot_time_series_all(all_groups, 'cpu')
✓ 0.8s
```

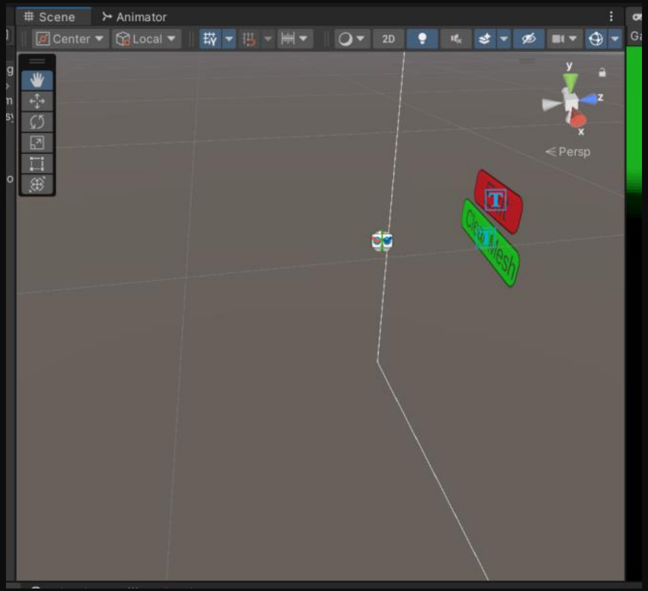
Short Form Scans

- Wall
- Wall + Chair (more surface area)
- Window (in progress)
- Concerned about variability making it hard to determine just how much environmental factors are impacting specific performance indicators





Unity

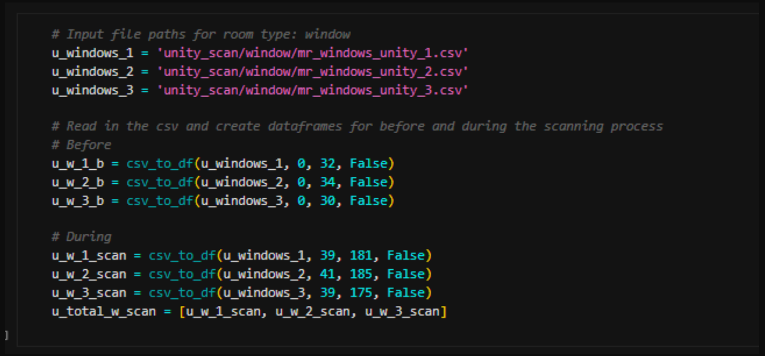


Completed

- Finished main scans
  - Able to compare data now
- Unity data uploaded to the VSCode to be analyzed
- Created buttons
  - Code isn't working…

To Do

- Zip Unity project into the VSCode
- Get buttons working
- Finish smaller scans to compare
- Expand boundary of mapping



```python
# Input file paths for room type: window
u_windows_1 = 'unity_scan/window/mr_windows_unity_1.csv'
u_windows_2 = 'unity_scan/window/mr_windows_unity_2.csv'
u_windows_3 = 'unity_scan/window/mr_windows_unity_3.csv'

# Read in the csv and create dataframes for before and during the scanning process
# Before
u_w_1_b = csv_to_df(u_windows_1, 0, 32, False)
u_w_2_b = csv_to_df(u_windows_2, 0, 34, False)
u_w_3_b = csv_to_df(u_windows_3, 0, 30, False)

# During
u_w_1_scan = csv_to_df(u_windows_1, 39, 181, False)
u_w_2_scan = csv_to_df(u_windows_2, 41, 185, False)
u_w_3_scan = csv_to_df(u_windows_3, 39, 175, False)
u_total_w_scan = [u_w_1_scan, u_w_2_scan, u_w_3_scan]
```

Current Questions