mpc_analysis

October 20, 2024

[ ]:
```
import pandas as pd
from utils_mpc import make_plot, rmse
```

[ ]:
```
data_path = 'all_mpc/'
desired_data_dict = {'1': 'mpc_CH.log',
                     '2': 'mpc_CH_2x.log',
                     '3': 'mpc_CV.log',
                     '4': 'mpc_CV_2x.log',
                     '5': 'mpc_F8H.log',
                     '6': 'mpc_F8H_2x.log',
                     '7': 'mpc_F8VS.log',
                     '8': 'mpc_F8VS_2x.log',
                     '9': 'mpc_F8VT.log',
                     '10': 'mpc_F8VT_2x.log',
                     '11': 'mpc_HELIX.log',
                     '12': 'mpc_HELIX_2x.log',
                     '13': 'mpc_sawtooth.log',
                     '14': 'mpc_sawtooth_2x.log',
                     '15': 'mpc_triangle.log',
                     '16': 'mpc_triangle_2x.log',
                     '17': 'mpc_CHS.log',
                      }
```

[ ]:
```
for i in range(1,18):
    data_file = desired_data_dict[str(i)]
    log_file_path = data_path + data_file #'vertcirc_1_OG.log' #helix_spin_0.
 ↪log #helix_spin_1_OG.log #helix_spin_jax.log #vertcirc_1_NEW.log
    df_log = pd.read_csv(log_file_path, header=0, dtype={'metadata': str})
    print(f"{i} rmse: {rmse(df_log)}")
```

```
1 rmse: 0.7455420816771026
2 rmse: 0.9303294279330608
3 rmse: 0.380768626488362
4 rmse: 0.4872637622179187
5 rmse: 0.5052098008027658
6 rmse: 0.4055230302806952
7 rmse: 0.41752940290678536
8 rmse: 0.4339281528695203
```

```
9 rmse: 0.4793536683497293
10 rmse: 0.4091149611782499
11 rmse: 0.8227820503837961
12 rmse: 0.9458675104176244
13 rmse: 0.38387945621444414
14 rmse: 0.4063009493537855
15 rmse: 0.20061772136606157
16 rmse: 0.27882806926246373
17 rmse: 0.7271584392173842
```

[ ]:
```
data_file = desired_data_dict['1']
log_file_path = data_path + data_file #'vertcirc_1_OG.log' #helix_spin_0.log
 ↪#helix_spin_1_OG.log #helix_spin_jax.log #vertcirc_1_NEW.log
# Reading the .log file into a pandas DataFrame
df_log = pd.read_csv(log_file_path, header=0, dtype={'metadata': str})

# Display the first few rows to confirm successful import
df_log.head()
```

[ ]:

	time	x	y	z	yaw	throttle	roll_rate	pitch_rate	yaw_rate	x_ref	y_ref	z_ref	yaw_ref	mpc_time	ctrl_callback_time_history	metadata
0	0.000004	-0.097932	-0.042978	-0.242654	0.028840	-0.622345	0.311682	-0.736683	-0.184362	0.0	0.0	-0.6	0.0	0.022432	0.027110	Hardware
1	0.028268	-0.097932	-0.042978	-0.242654	0.028840	-0.622345	0.311682	-0.736683	-0.184362	0.0	0.0	-0.6	0.0	0.001094	0.004769	1x Speed
2	0.035926	-0.097884	-0.043092	-0.242548	0.028861	-0.622530	0.308951	-0.736645	-0.184490	0.0	0.0	-0.6	0.0	0.008902	0.013041	Horizon:3.0
3	0.050433	-0.097884	-0.043092	-0.242548	0.028861	-0.622530	0.308951	-0.736645	-0.184490	0.0	0.0	-0.6	0.0	0.001113	0.004853	Num Steps:20
4	0.059910	-0.097847	-0.043200	-0.242458	0.028861	-0.622102	0.312656	-0.738795	-0.184490	0.0	0.0	-0.6	0.0	0.007195	0.011186	No Pyjoules

[ ]:
```
actual_values = df_log[['x', 'y', 'z', 'yaw']].to_numpy()
reference_values = df_log[['x_ref', 'y_ref', 'z_ref', 'yaw_ref']].to_numpy()
```

[ ]:
```
print(f"This data comes from: {df_log['metadata'][0]}")
print(f"Speed: {df_log['metadata'][1]}")
print(f"{df_log['metadata'][2]}")
print(f"{df_log['metadata'][3]}")
```

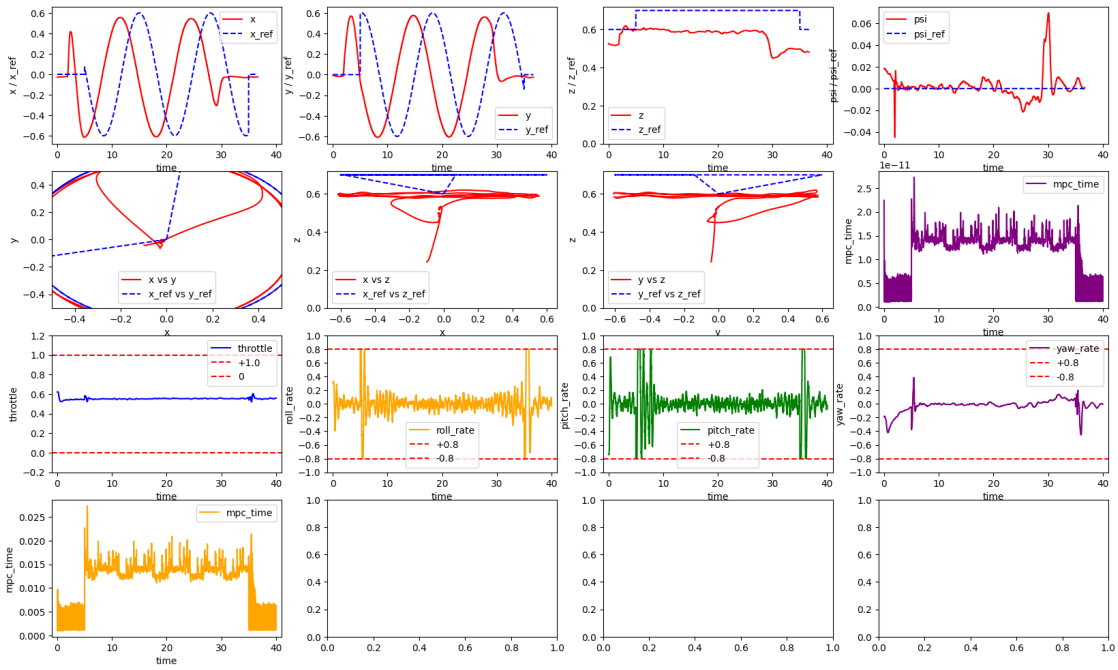
```
print(f"{df_log['metadata'][4]}")
print(f"Mean computation time: {df_log['mpc_time'].mean()}")
print(f"Mean callback time: {df_log['ctrl_callback_time_history'].mean()}")
print(f"RMSE: {rmse(df_log)}")
```

```
This data comes from: Hardware
Speed: 1x Speed
Horizon:3.0
Num Steps:20
No Pyjoules
Mean computation time: 0.01011410766363767
Mean callback time: 0.014968894049198144
RMSE: 0.7455420816771026
```

1 Plot it!

[ ]:
```
# Reuse the plotting function defined earlier for the uploaded .log data
make_plot(df_log)
```

```
This data comes from: Hardware
Speed: 1x Speed
Mean computation time: 0.01011410766363767
Mean callback time: 0.014968894049198144
RMSE: 0.7455420816771026
```



[ ]:

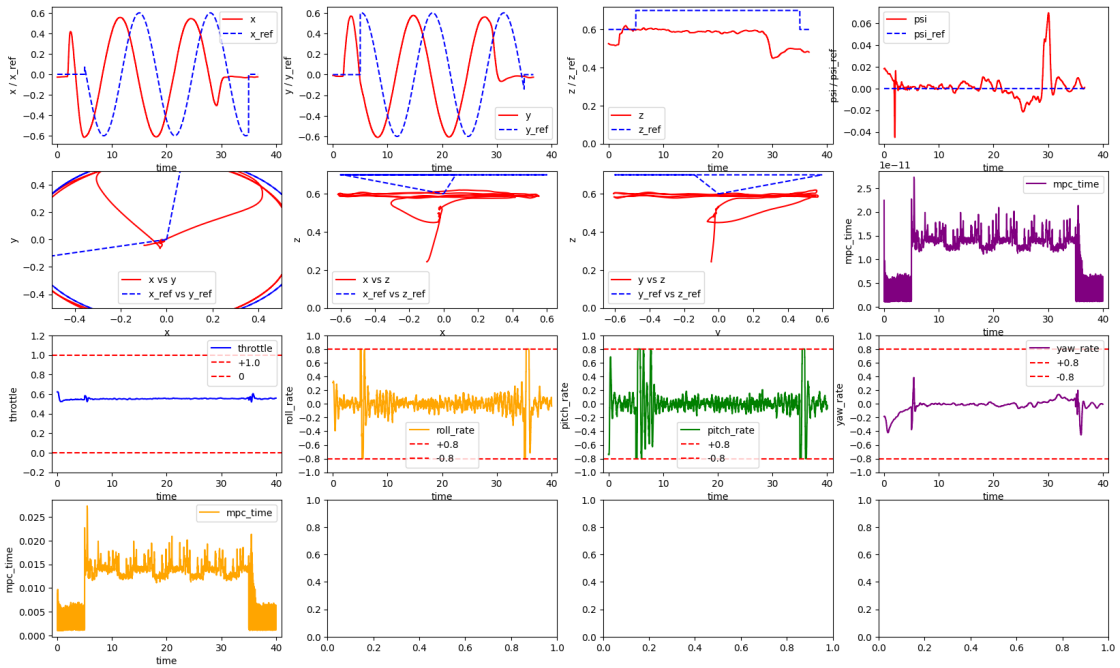
[ ]:

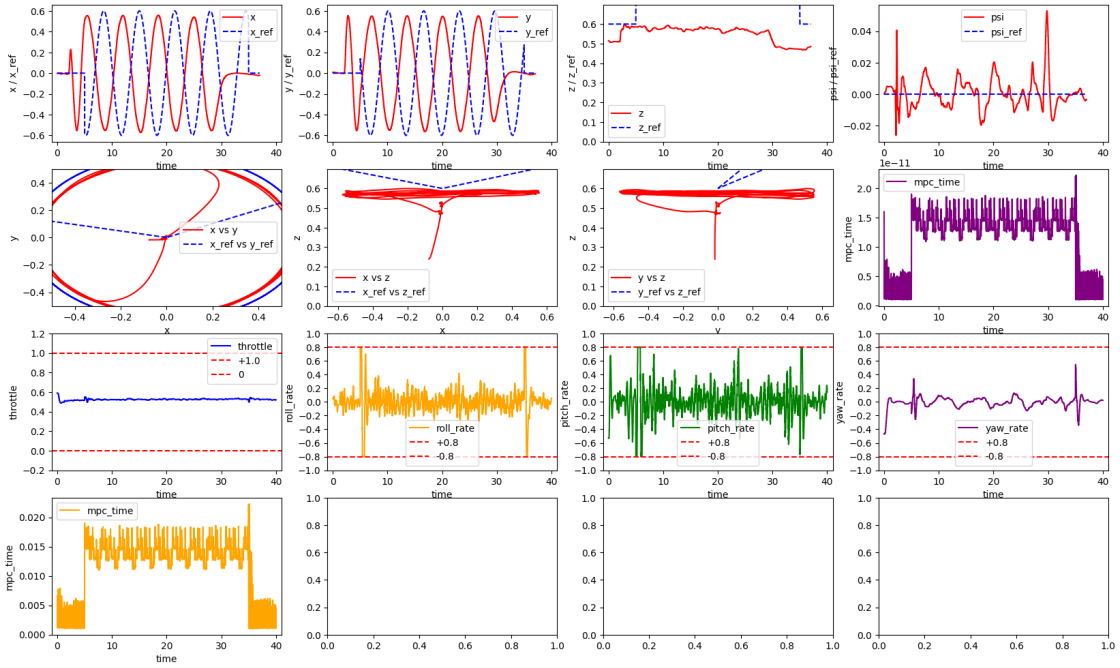
[ ]:
```
for i in range(1, 18):
    data_file = desired_data_dict[str(i)]
    log_file_path = data_path + data_file
    df_log = pd.read_csv(log_file_path, header=0, dtype={'metadata': str})

    # Ensure the entire text before '=' takes up 20 characters
    text_before_equals = f"{desired_data_dict[str(i)][:-4]} rmse"
    print(f"{i}) {text_before_equals:<20} = {rmse(df_log):.6f}")
```

```
1) mpc_CH rmse          = 0.745542
2) mpc_CH_2x rmse       = 0.930329
3) mpc_CV rmse          = 0.380769
4) mpc_CV_2x rmse       = 0.487264
5) mpc_F8H rmse         = 0.505210
6) mpc_F8H_2x rmse      = 0.405523
7) mpc_F8VS rmse        = 0.417529
8) mpc_F8VS_2x rmse     = 0.433928
9) mpc_F8VT rmse        = 0.479354
10) mpc_F8VT_2x rmse     = 0.409115
11) mpc_HELIX rmse       = 0.822782
12) mpc_HELIX_2x rmse    = 0.945868
13) mpc_sawtooth rmse    = 0.383879
14) mpc_sawtooth_2x rmse = 0.406301
15) mpc_triangle rmse    = 0.200618
16) mpc_triangle_2x rmse = 0.278828
17) mpc_CHS rmse         = 0.727158
```

[ ]:
```
for i in range(1, 3):
    data_file = desired_data_dict[str(i)]
    log_file_path = data_path + data_file
    df_log = pd.read_csv(log_file_path, header=0, dtype={'metadata': str})
    make_plot(df_log)
```

```
This data comes from: Hardware
Speed: 1x Speed
Mean computation time: 0.01011410766363767
Mean callback time: 0.014968894049198144
RMSE: 0.7455420816771026
This data comes from: Hardware
Speed: 2x Speed
Mean computation time: 0.009839157948548716
Mean callback time: 0.01395336463925086
RMSE: 0.9303294279330608
```





[ ]:
```
for i in range(1,3):
    print(i)
```

```
1
```

2

[ ]: