Genetic Algorithm based PI controller tuning with stability analysis for Linux ptp4l optimization

Marta Plantykow, Milena Olech, Maciek Machnikowski

Netdev 0x17

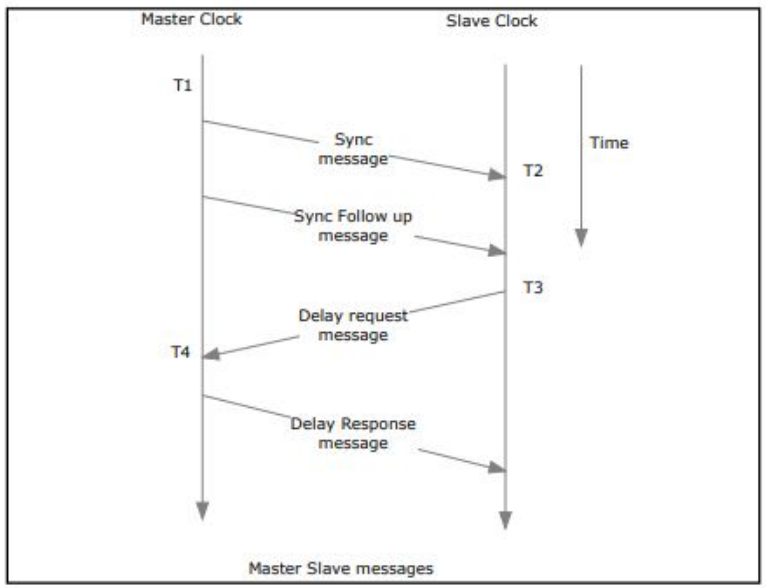
October 27, 2023

Agenda

Precision Time Protocol

IEEE 1588 standard defines a method for precise computer synchronization over Local Area Network - ***Precision Time Protocol*** (PTP).

PTP Synchronization messages, such as *Sync*, *FollowUp*, *DelayReq*, and *DelayResp* are exchanged between the provider of time (leader) and instance synchronizing to the provider (follower).



LinuxPTP

Linuxptp tool set provides an implementation of 1588 on Linux. It consist of five main tools:

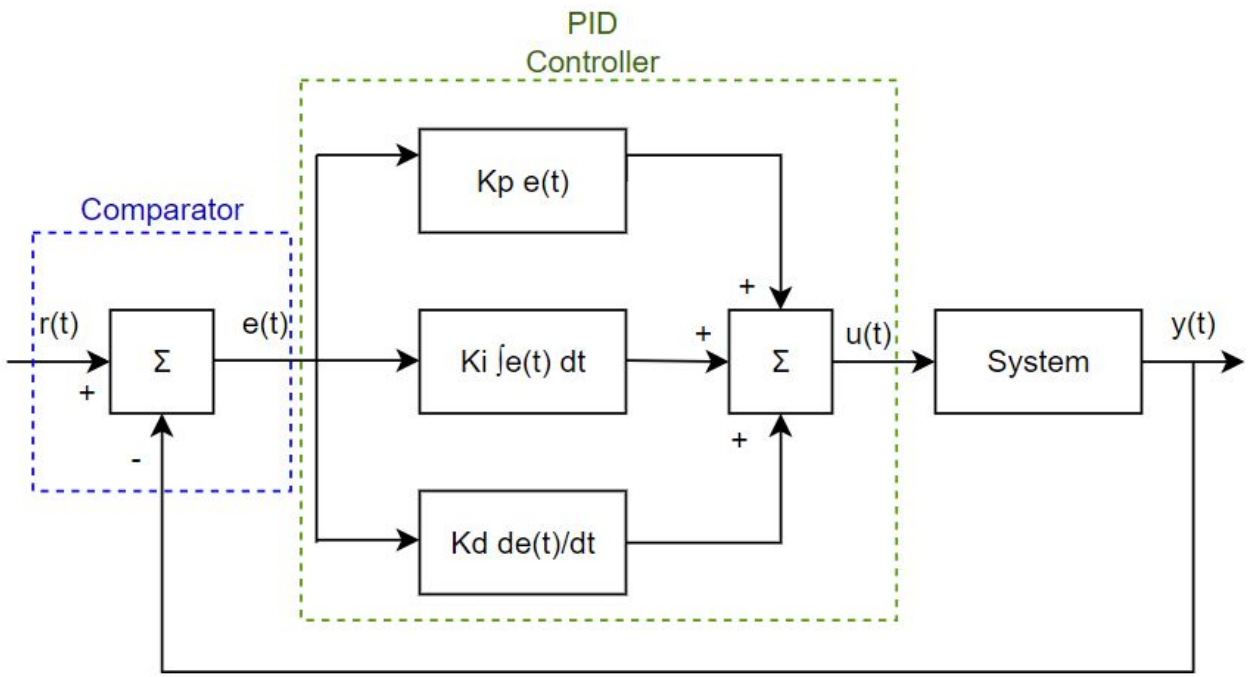
- phc2sys
- ptp4l
- ts2phc
- pmc
- phc_ctl

ptp4l

- Command-line tool
- Generates PTP traffic and controls the PHC by setting its time and frequency.
- Can be used to synchronize multiple clocks and can be considered both a source - leader clock - and the consumer - follower clock - of the time packets
- If the connection between the leader and the follower is established, the application provides the output with thorough precision tracking

PID controller

1. Proportional-Integral-Derivative controller
2. Tuning the PID controller affects the system response
3. PID tunes the process of manipulating the K_p, K_i, and K_d



Stability verification

Stability is the most important control system specification. There are three stability criteria based on the response characteristic:

- Every system for which each bounded input yields a bounded output is a ***stable system***.
- Every system for which each bounded input yields unbounded output is an ***unstable system***.
- Every system for which some bounded input yields an unbounded output is ***marginally stable system***.

PTP stability

linuxptp suite implements IEEE 1588 using the Proportional-Integral (PI) controller in the clock's control system.

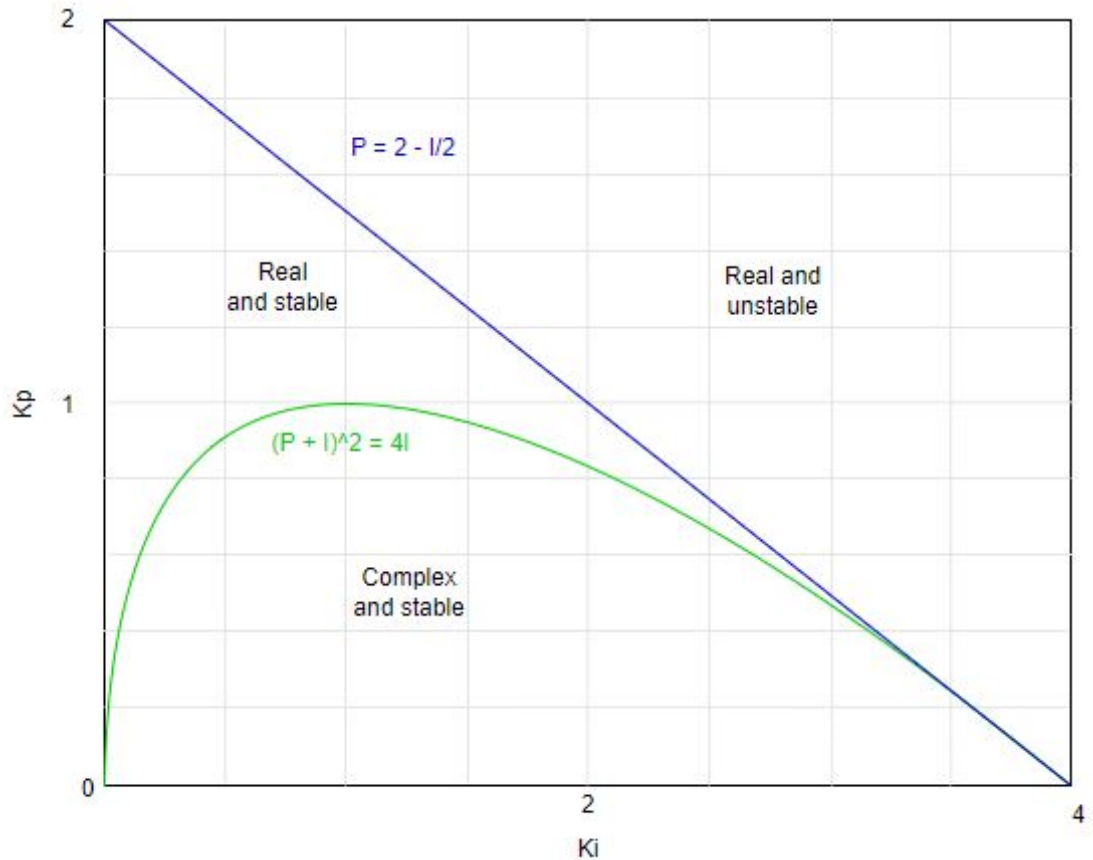
Software implementation of PI controller in linuxptp ***limits*** in the range of ***parameters*** K_p and K_i: $K_p \leq 2$, $K_i \leq 1$.

The default values for linuxptp are $K_p = 0.7$ and $K_i = 0.3$

A two-dimensional plane representing the relationship between parameters K_p and K_i can be divided into ***three sections***, each representing ***different*** type of ***stability***.

PTP stability

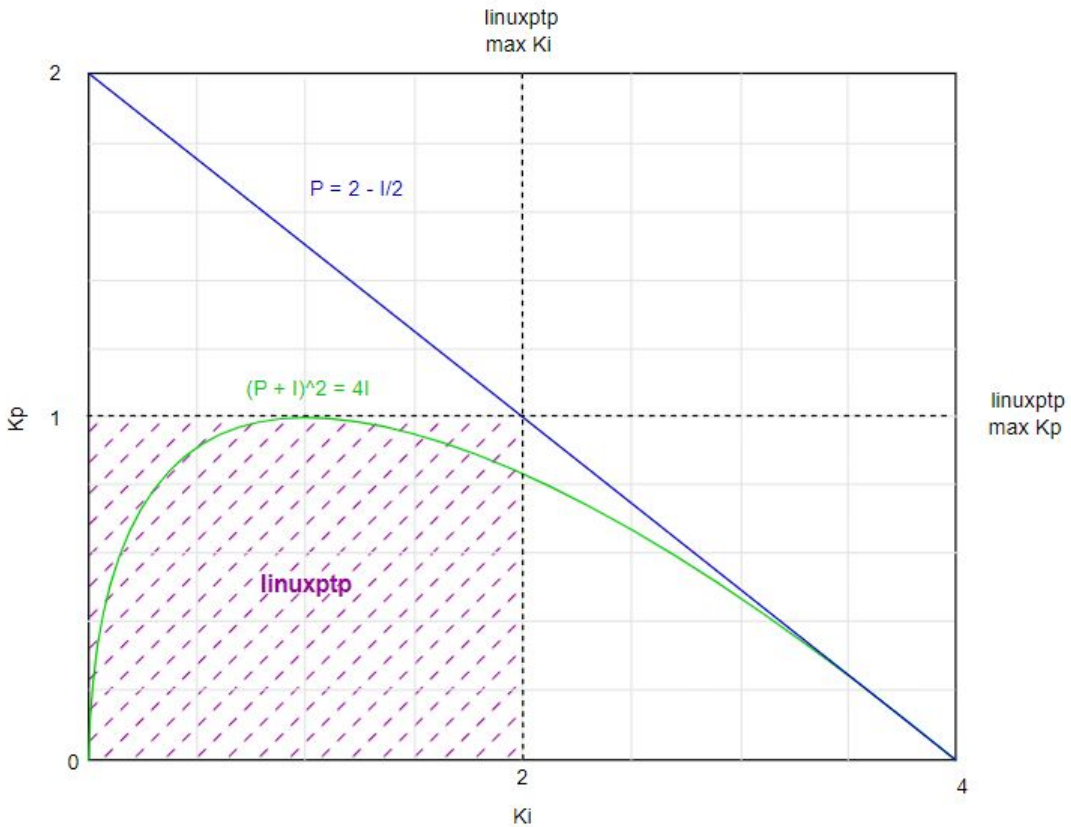
Stability regions for Kp and Ki parameters



Measurement, Control, and Communication Using IEEE 1588 (Eidson, 2006)

PTP stability

Linuxptp stability regions for Kp and Ki parameters



Data evaluation methods

PTP4l synchronization accuracy may be scored using metrics, where expected values are compared with observations:

- expected values $P_i, i = (1, ..., n)$
- observations $O_i, i = (1, ..., n)$

To evaluate data, a RMSE metric defined by following formula has been used:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} |e_i|^2} \tag{1}$$

where e_i is the difference between observation and expected value.

Data Evaluation

The main purpose of ptp4l synchronization is having the leader and the follower PHC timers as closely as possible. The ideal scenario is to have the offset between the leader and the follower equal to 0.

Data evaluation process measures the distance from an offset returned from ptp4l run on a follower side (observations) to 0 (expected value).

Framework

The proposed framework's is capable of ***optimizing the time synchronization*** using ***the linuxptp toolbox*** by optimizing ***the Proportional – Integral*** servo's parameters using ***the Genetic Algorithm***.

```
maciek@rpi-client:~/PTP-Optimization $ sudo ./main.py --i eth0 --t 20
Measuring result with default settings...
MoTTY X11 proxy: Authorisation not recognised

In case you are trying to start a graphical application with "sudo", read
https://blog.mobatek.net/post/how-to-keep-X11-display-after-su-or-sudo/

RMSE: 287.757
Default k_p: 0.7 default k_i: 0.3 Score: 287.757

Creating initial population...
Initial population created!
****************************************************************
EPOCH NUMBER 0
****************************************************************
Epoch 0: creature 0, k_p 0.628, k_i 1.588 RMSE: 197.865
Epoch 0: creature 1, k_p 0.089, k_i 1.898 RMSE: 130.404
Epoch 0: creature 2, k_p 0.066, k_i 0.325 RMSE: 1045.083
Epoch 0: creature 3, k_p 0.520, k_i 1.607 RMSE: 6249.194
Epoch 0: creature 4, k_p 0.917, k_i 0.551 RMSE: 496213.886
Epoch 0: creature 5, k_p 0.304, k_i 1.007 RMSE: 688782.09
Epoch 0: creature 6, k_p 0.443, k_i 1.174 RMSE: 693832.332
Epoch 0: creature 7, k_p 0.743, k_i 1.368 RMSE: 745966.806
Epoch 0: creature 8, k_p 0.017, k_i 1.583 RMSE: 1290211.961
Epoch 0: creature 9, k_p 0.235, k_i 0.608 RMSE: 744883.0
Epoch 0: Best score: 130.404 default 287.757
Result better by 54.7%

Crossing parents...
New generation creation - crossed creatures added!
Replicating parents...
New generation creation - replicated creatures added!
Adding new random parents
New generation creation - random creatures added!
Mutants are coming...
Mutation finished!
****************************************************************
Progress: 100.0%
****************************************************************
maciek@rpi-client:~/PTP-Optimization $
[0] 0:bash*
```

Framework - Introduced changes

Improvements applied to the PTP-Optimization framework:

1. support for a new application - ptp4l
2. advanced stability verification
3. extended logging

Framework - Stability verification

The current framework's implementation allows configuring parameters for stability verification.

Three options are available:

- **Complex** - reduces the possible search space for optimal values to the "Complex and Stable"
- **Real** - reduces the possible search space for optimal values to the "Real and Stable"
- **False** - reduces the possible search space for optimal values to the area limited by variables specified in the config file

Test 1

The purpose of the first test was to investigate the test repeatability and find an optimal time of execution for further tests.

Test scenario:

- 60 minutes of running ptp4l application
- Default K_p and K_i values
- Repeat each run 10 times
- Collect data about metric change in several time intervals
- Disabled stability verification

Test 1 Results

	RMSE						
Run	30s	90s	150s	300s	**600*s***	1200s	3600s
1	44.85	26.76	21.47	16.53	12.82	10.90	9.65
2	51.88	30.58	24.05	17.96	14.13	12.51	10.49
3	41.13	24.97	20.03	15.65	12.56	10.85	10.16
4	7.22	12.55	11.52	12.05	10.94	10.07	9.51
5	16.07	11.62	10.60	10.45	9.45	9.48	9.88
6	53.51	31.74	25.13	18.73	14.56	11.85	10.26
7	7.53	9.37	8.78	9.06	8.84	9.65	9.41
8	49.24	28.98	22.97	17.26	13.53	11.25	9.58
9	17.69	13.00	11.73	10.15	9.56	9.69	9.25
10	28.64	18.03	14.96	11.85	10.24	9.56	8.74
Ave.	31.78	21.06	17.12	13.95	11.66	10.58	9.69
Stdev	17.56	8.03	5.92	3.44	1.99	1.00	0.49

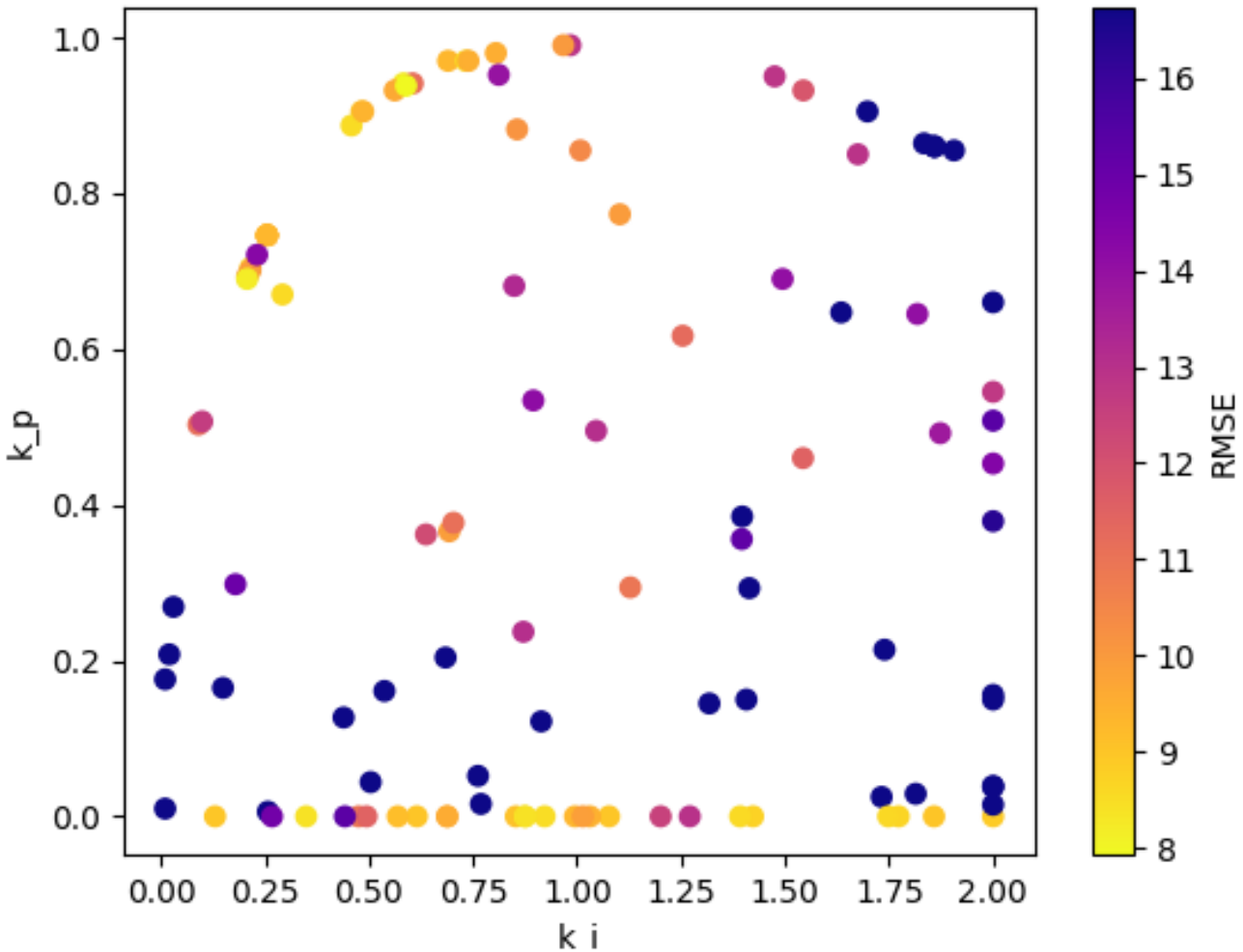
Test 2

The purpose of the third test was to investigate the impact of stability verification on the results.

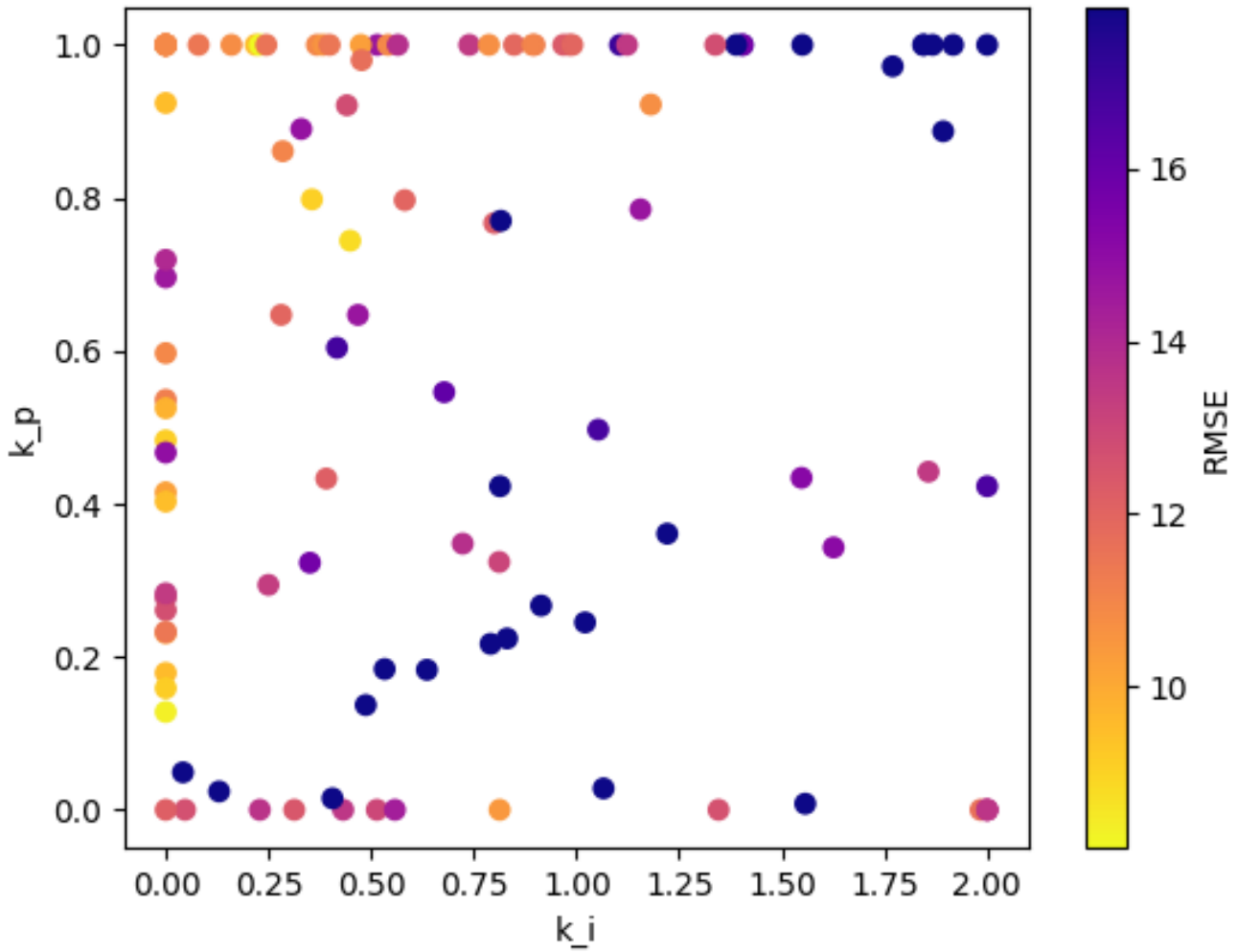
Test scenario:

- linuxptp built-in boundaries equal $K_p \leq 1$ and $K_i \leq 2$
- Case1: "Complex and stable" region, additionally limited by the linuxptp stability limits
- Case2: The full stability region limited by built-in linuxptp boundaries

Test 2 Results - Case1



Test 2 Results - Case2



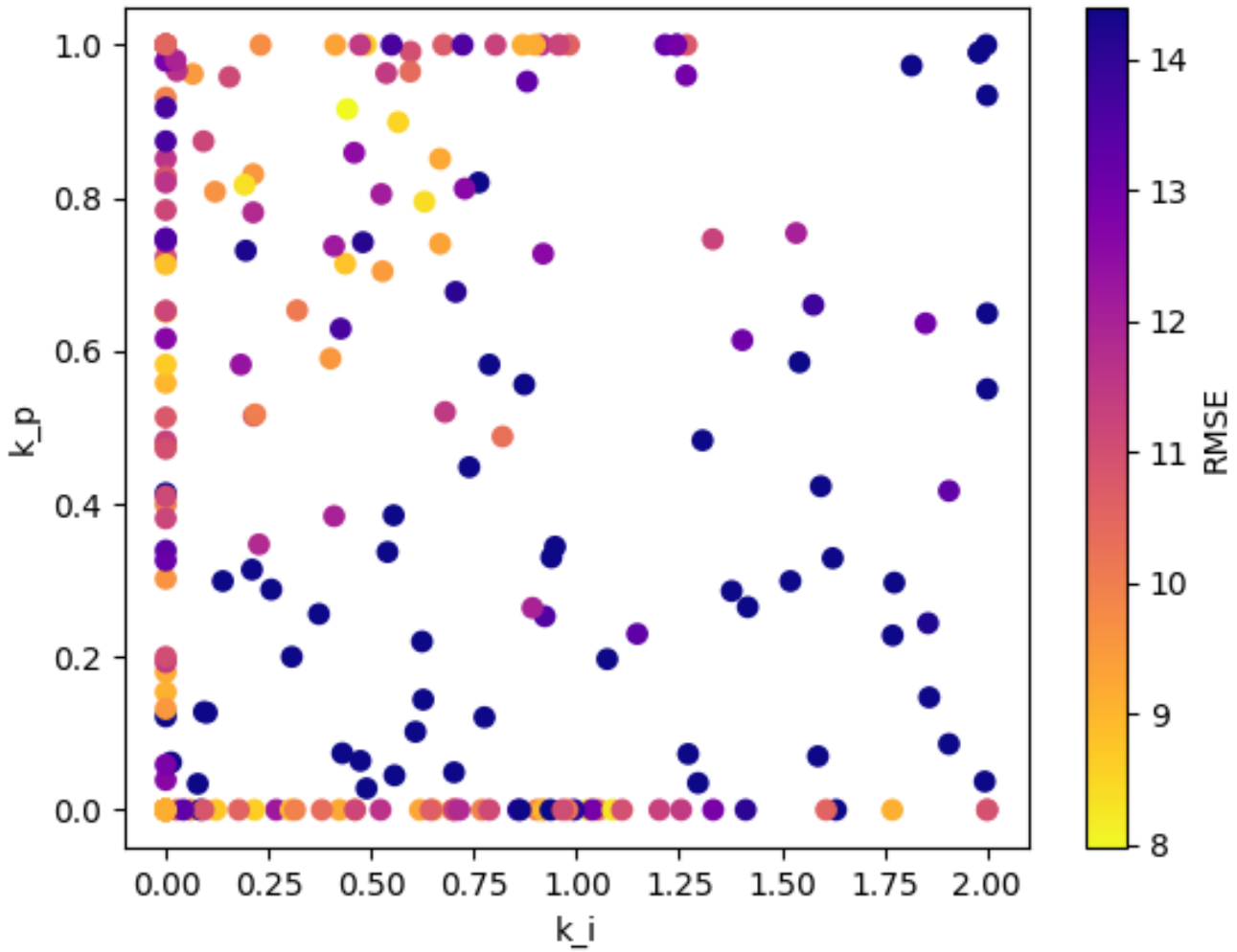
Test 3

The purpose of the third test was to investigate the influence of increasing the number of epochs on results.

Test scenario:

- Number of epochs increased to 10.
- Disabled stability verification

Test 3 Results



Conclusions

1. Greater K_p and lower K_i brings better results than default values
2. "Complex and stable" area excludes a crucial portion of good results
3. Increasing the number of epochs does not significantly improve the framework's results
4. With K_p and K_i values generated by GA, the framework reaches better scores than with default values

Key takeaways

- linuxptp implements IEEE 1588 using the Proportional-Integral (PI) controller that can be optimized to achieve better synchronization in a shorter time.
- PTP optimization is enhanced with the study of stability regions.
- The new framework version supports the ptp4l application and stability verification.
- In all examined cases, the framework demonstrated its capability to achieve results better than the default configuration.

Q&A