Measuring the network RTT in a moving train: Fun project

***Abstract*—What to do on the train if the journey takes hours? Of course, measure the packet latency to the selected server in the Internet and plotting the data after interpolation on the map. Fun? Yes, it is.**

I. Introduction

I was always interested in collecting data about latency and network coverage while traveling on a train. And today I have got the opportunity to do so. I took the trip from Helsinki to Rovaniemi by train. The trip is about 9-10 hours long. So I have decided to take the opportunity to collect such data.

The purpose of this document is not to provide a complete map trace of the trip, but rather to show how to use GPS2IP iOS application and a few simple scripts to collect the needed data.

As a side note, I should say that the network is amazing in Finland (compared to other countries I have been to).

II. Data collection methodology

We have used the iPhone GPS2IP application to pick up the GPS coordinates from the internal GPS receiver to get the current position (latitude and longitude) in GPRMC format. The application was sending the coordinates to the MacBook laptop over a UDP socket. The Python script was registering the coordinates and the timestamps, as well as putting the data into a file. At the same time, the script was computing average RTT (in milliseconds) by invoking an external bash script.

Here is the script I was using to do the job:

```
from threading import Thread

import socket

from time import sleep
from time import time

UDP_IP = "172.20.10.2"
UDP_PORT = 10000

fd = open("working.log", "w+")

def gps():
        sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        sock.bind((UDP_IP, UDP_PORT))
        while True:
                data = sock.recvfrom(1024)
                fd.write(f"{time()} {data[0].decode('ASCII').strip()}\n")
                fd.flush()
                sleep(4)

import subprocess

def icmp():
        while True:
                rtt = subprocess.check_output(["bash", "ping-host.sh"])
                fd.write(f"{time()} {rtt.decode('ASCII').strip()}\n")
                fd.flush()
                sleep(4)

t1 = Thread(target=gps, args=(), daemon=True)
t1.start()

t2 = Thread(target=icmp, args=(), daemon=True)
t2.start()

while True:
        sleep(10)
```

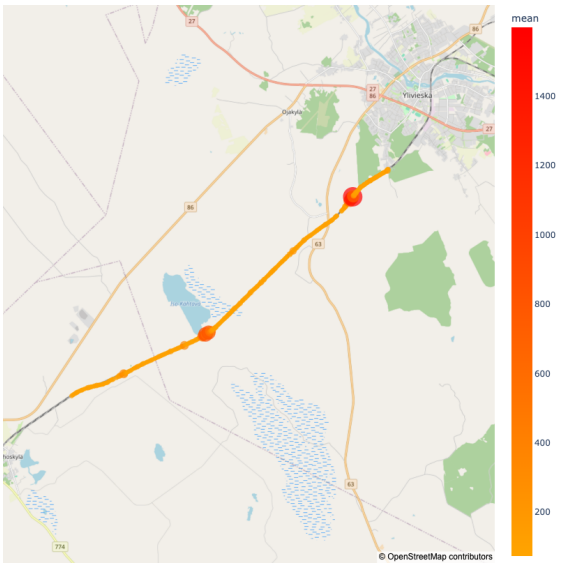

Fig. 1: Trace of the RTT values

III. Data processing and basic results

For the demonstration, we have collected a rather small trace (30-40 minutes long) for the part of the journey from Tampere to Rovaniemi. At first, we combined two measurements using the closest timestamps and created a single log file (since the GPS and RTT values were stored in two different files). We then used the plotly library to show the RTT values on Open Street Map. The results of our effort are shown in Figure 1. The source code of the stuff is here [1].

References

[1] Fun stuff in the train. https://github.com/dmitriykuptsov/train-rtt.