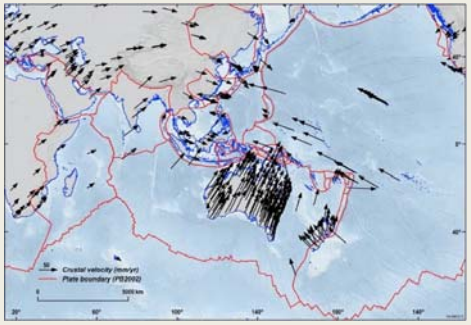
MQTT Protocol for Real-Time GNSS Data and Correction Distribution

Precise Positioning



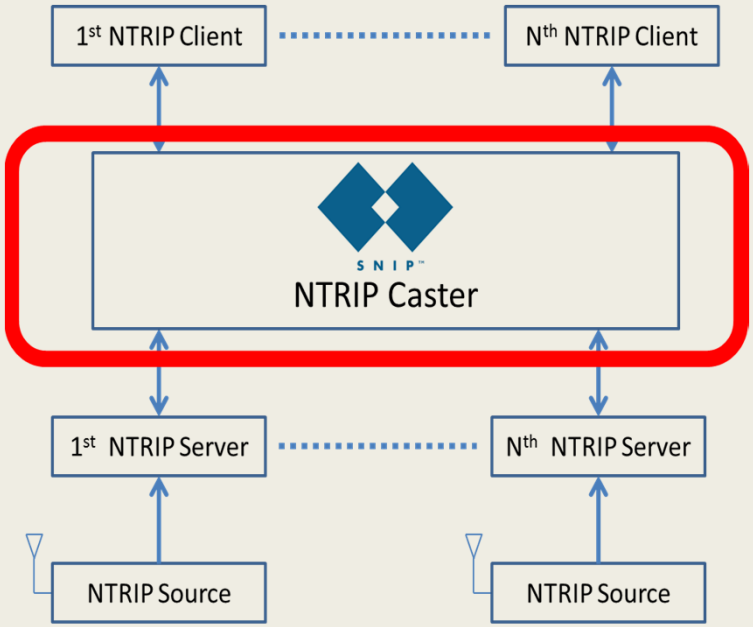
Precise Positioning

- Real-Time Kinematic (RTK) positioning
- Precise Point Positioning (PPP)
- Requires additional correction!

- Communication mediums
  - *VHF/UHF radio*
  - *WAN/LAN/WLAN*
  - *3G/4G cellular network*
- Communication Protocols
  - *TCP/IP server*
  - *NTRIP*

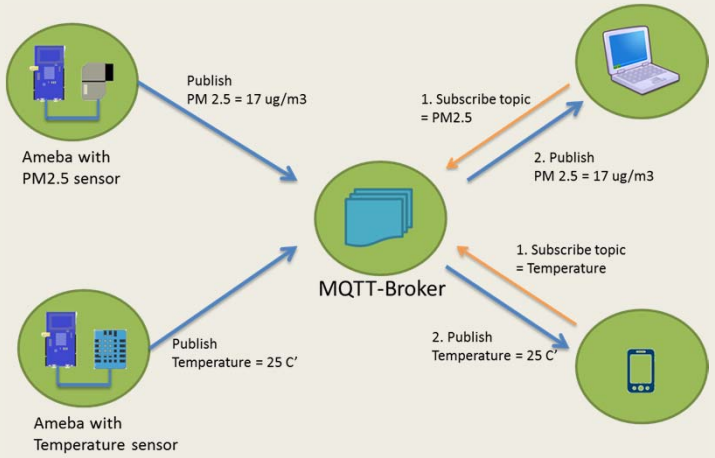
Precise Positioning

- Networked Transfer of RTCM via Internet Protocol (Ntrip)
  - *Radio Technical Commission for Maritime Services standard*
  - *Dissemination of GNSS observation and correction since 2004*
  - *HTTP/1.1 standard*
  - *Support hundreds of reference stations and up to thousand users*
- Not suitable for future applications



MQTT

- Message Queue Telemetry Transport (MQTT) protocol
  - *publish-subscribe model*
  - *broker distribute messages based on the topic of a message*
  - *Suitable for low-bandwidth, low-power, limited hardware resources*
  - *Advance features: QoS, security supports, high availability*
  - *Used by Facebook Messenger, Amazon Web Services, OpenStack and Microsoft Azure.*



Ntrip vs MQTT

- Server performance
  - *System loads (CPU, memory) under various operational scenarios*
- Positioning performances

Ntrip	MQTT	Positioning
• Str2str – NtripCaster and NtripClient • Python-based Ntrip client	• Mosquitto broker and clients • Paho MQTT python libraries	• Rtkrcv rtk positioining

Ntrip vs MQTT

Nectar S1

NTRIP Caster

Nectar S2

MQTT broker

Nectar S3

Data stream manipulation and distribution

- Ntrip Server
- MQTT publisher

Nectar S4

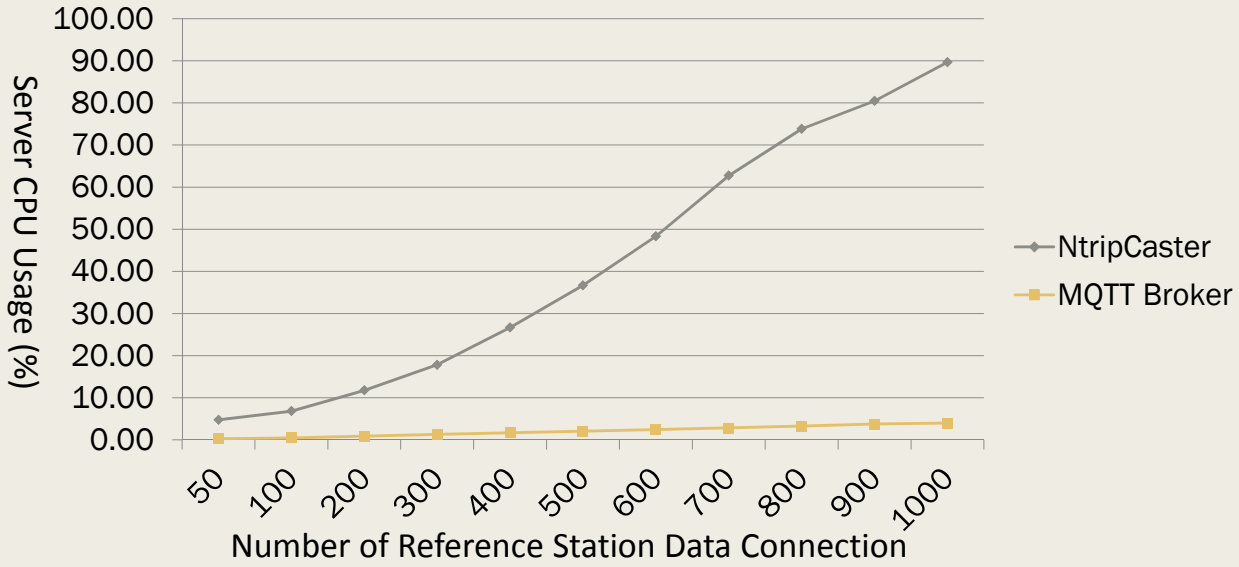
Data stream connection and management & Positioning

Sever Config

- 1 CPU @ 2.3GHz
- 4 GB ram
- 40 GB HD disk

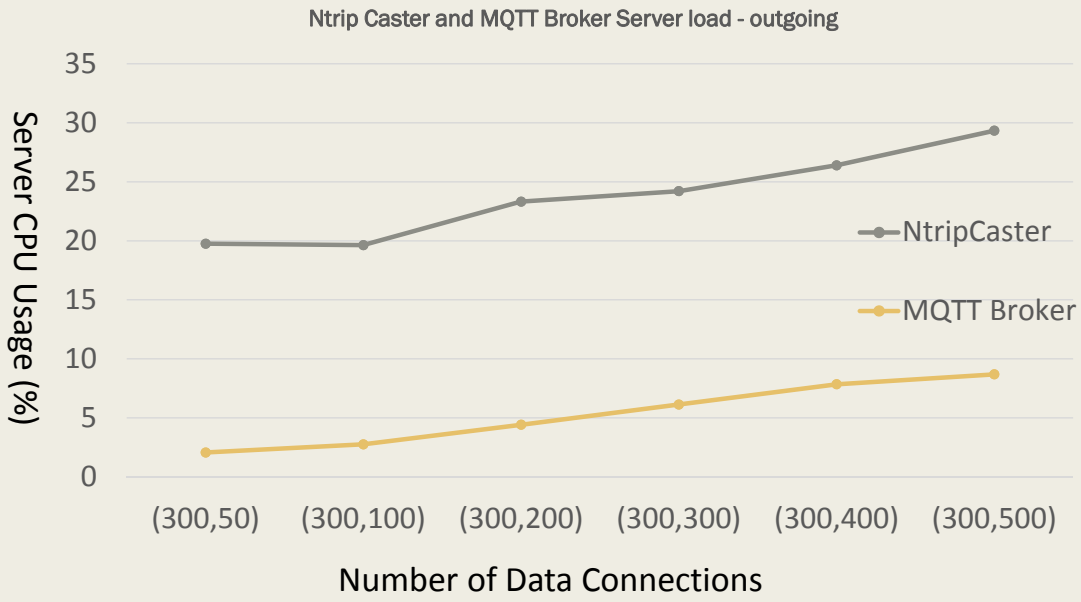
System Load Comparison

- MQTT broker requires less resources than Ntrip with similar incoming data streams
  - *NtripCaster: 4.7%11.76%, 36.68%, 89.67% for 50/200/500/1000 incoming data streams.*
  - *MQTT server load were 0.28%, 0.85%, 2.06%, 4.00% for the same scenarios.*
  - *no significant use of memory*



System Load Comparison

- User connections have less stress on the server load.
  - *For a commercial CORS network of 300 reference stations*
  - *NtripCaster load 19.77% to 29.33% for 50 and 500 user connections*
  - *MQTT broker load 2.07% to 8.69% for the same scenario*

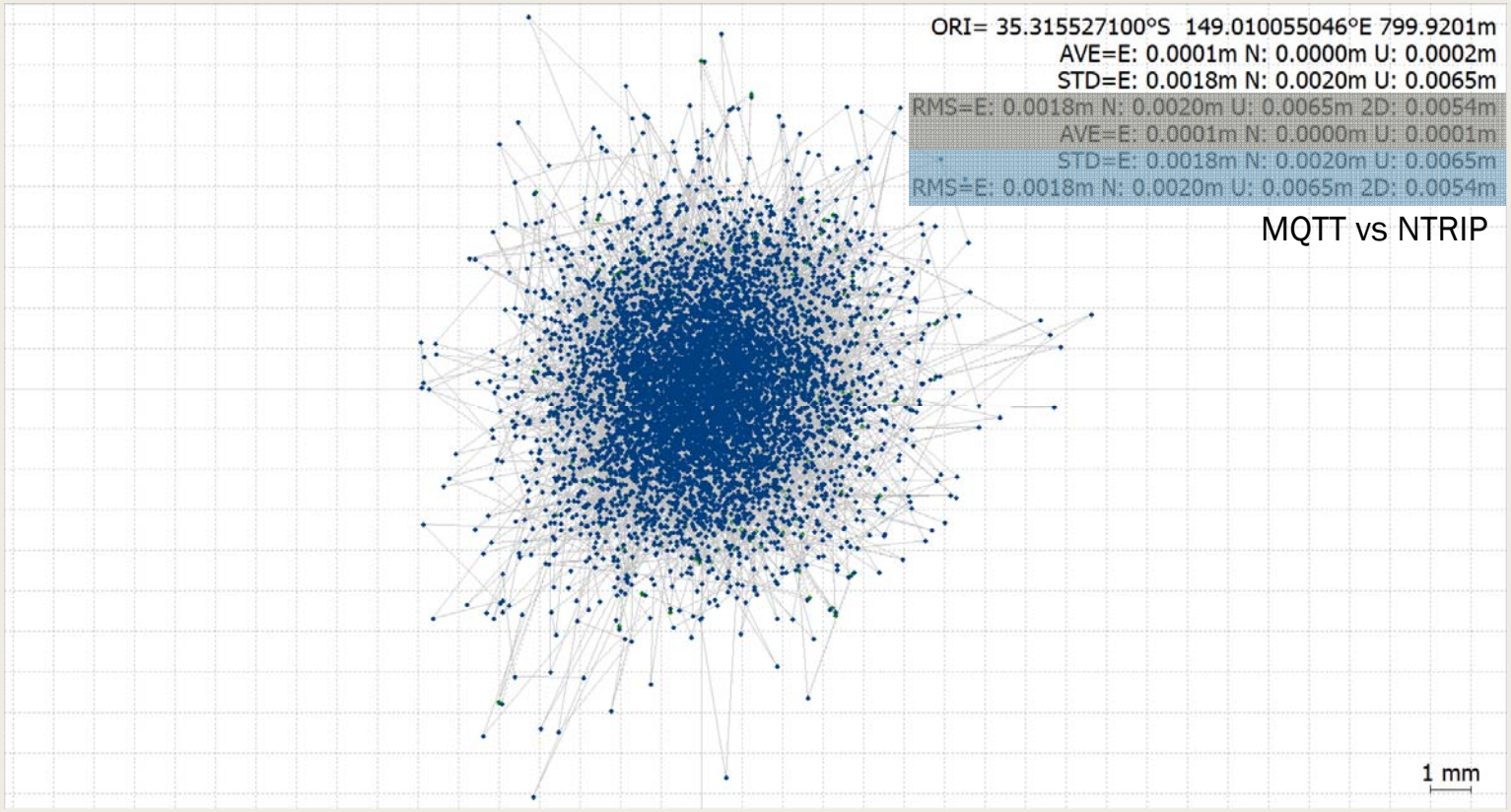


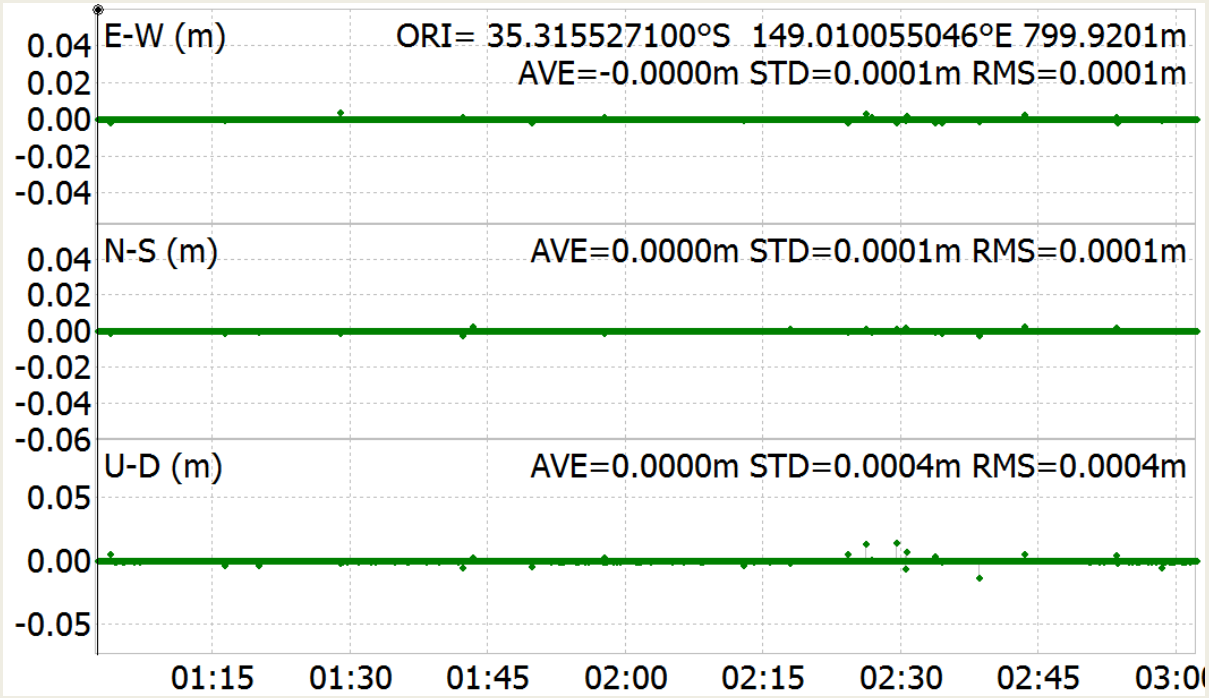
Positioning Validation

- Positioning configuration
  - *STR17 as rover via Ntrip*
  - *STR24 as base via Ntrip or MQTT*
  - *RTK positioning solution*
  - *Two rtkrcv module running simultaneously*

- Positioning via MQTT protocol
  - *Python MQTT client subscript to STR24 data stream*
  - *Data streamed to local TCP server (localhost:10011)*
  - *RTKRCV connected to the TCP server to get data streamed from MQTT broker*

Positioning Validation





Differences between NTRIP/MQTT positioning solution

- Insignificant RMS positioning variation between NTRIP/MQTT solution
  - *0.1mm in the horizontal and 0.4mm in vertical*
  - *due to network latency/packet losses.*
  - *23 instances of latency difference, 15 of which MQTT > NTRIP*

Conclusion and Future Works

- MQTT broker is more efficient than Ntrip Caster in handling large load
  - *Less computing resources needed*
  - *Horizontal scalability (emqtt / HiveMQ, etc)*
  - *Advance features such as QoS, SSL/TSL, topic subscription*
- Different topic level publish and subscription

MQTT/GNSS/CORS/STR24/RTCMv3/GPS/1019	(GPS NAV)
MQTT/GNSS/CORS/STR24/RTCMv3/GAL/1045	(GAL NAV)
MQTT/GNSS/CORS/STR24/RTCMv3/GPS/1077	(GPS OBS)
MQTT/GNSS/CORS/STR24/RTCMv3/GAL/1097	(GAL OBS)
Subscript: MQTT/GNSS/CORS/STR24/RTCMv3/GPS/#	(all topic under STR24 level)

Conclusion and Future Works

- Future testing and validation
  - *Latency*
  - *Bandwidth optimization*
  - *Security*
- MQTT v5 features
- Field vehicle testing