# GNSS RTK Application Note

**GNSS Module Series**

Version: 1.0

Date: 2024-08-06

Status: Released

# Legal Notices

# Use and Disclosure Restrictions

## License Agreements

## Copyright

# About the Document

| Document Information | |
|---|---|
| **Title** | **GNSS RTK Application Note** |
| **Subtitle** | GNSS Module Series |
| **Document Type** | Application Note |
| **Document Status** | Released |

## Revision History

| Version | Date | Description |
|---|---|---|
| - | 2022-04-15 | Creation of the document |
| 1.0 | 2024-08-06 | First official release |

# Contents

## Table Index

## Figure Index

# 1 Introduction

Positioning by satellite navigation systems is more or less accompanied by a certain level of random inaccuracy, which is roughly 5 to 15 meters. Although the accuracy can meet most positioning application scenarios, ordinary satellite navigation system positioning cannot meet high-precision positioning application scenarios, such as decimeter-level and centimeter-level applications. At this time, the differential global satellite navigation system (DGNSS) technology can be used to eliminate common random errors, thereby improving the accuracy of the receiver's position, speed and time. As a kind of DGNSS technology, RTK can be used to solve real-time dynamic position information, and the positioning accuracy can reach the decimeter or even centimeter level.

This document focuses on the use of the RTK positioning function of Quectel's high-precision positioning module.

**Table 1: Applicable Modules**

| Module Series | Model |
|---|---|
| LC29H | L C29H (BA) |
| | L C29H (DA) |
| | L C29H (EA) |
| LG69T | LG69T (AM) |
| | LG69T (AP) |
| LG290P | LG290P (03) |
| QLM29H | QLM29HBAA-GM |

# 2 RTK Overview

RTK (Real Time Kinematic) is a real-time differential GNSS technology based on carrier phase measurements, which achieves high-precision positioning (centimeter-level accuracy). As the GNSS signal travels from satellites to the receiver, it experiences delays and distortions caused by ionosphere (ionospheric effect) and atmosphere. The ionospheric effect significantly slows the signal and also can disturb it on the way. These factors, along with other influences, such as clouds or obstacles, can affect the signal travel time and increase the position errors.

Compared to traditional GNSS, the RTK technology based on carrier measurements can eliminate the major errors with a reference station transmitting the corrections to a receiver needed to get the accurate position. Besides, both the RTK and traditional GNSS require the measurement of the distance from the satellite to the receiver. Unlike traditional GNSS, which relies on PRN code-based ranging, RTK utilizes carrier-based ranging for higher accuracy because the carrier signal wavelength (e.g., 19 cm for the L1 signal) is much shorter than the PRN code wavelength (e.g., around 300 m for L1). The distance to a satellite is calculated by multiplying the carrier wavelength by the number of complete carrier cycles counted between the satellite and the receiver and then adding the phase difference. The key to achieving centimeter-level accuracy in RTK is to resolve the unknown number of carrier cycle ambiguities into integers.

Depending on the number and status of visible satellite signals, RTK can obtain position solutions with varying accuracy levels, i.e., “RTK single-point solution”, “RTK float solution” and “RTK fixed solution”. In RTK singl-point solution, the 3D coordinates calculated by the receiver without any correction information are less stable and the error is larger. In RTK fixed solution, the ambiguity is an integer. In RTK float solution, the ambiguity is allowed to be a decimal or floating-point number. General, the accuracy in RTK float solution is from 10 cm to 50 cm or even worse, and an accuracy of less than 10 cm can be achieved in RTK fixed solution.

An RTK system consists of three main parts:

- **Base Station:** A stationary reference receiver with a precisely known location that transmits correction data to the rover.
- **Communication Link:** A reliable method (e.g., radio, cellular) for transmitting correction data from the base station to the rover.
- **Rover:** The mobile receiver that utilizes the base station correction data to achieve high-precision positioning.

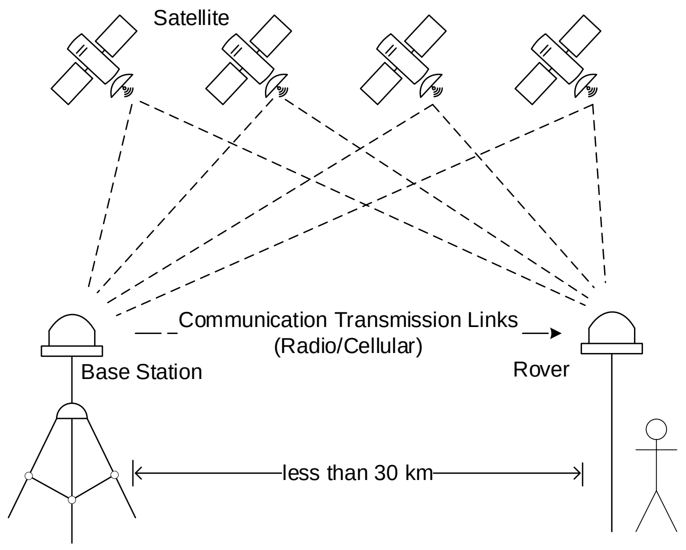


**Figure 1: RTK Working Principle**

RTK working principles:

- Place the base station at a precisely surveyed location with a clear view of the sky to receive GNSS signals continuously.
- The base station sends its coordinates and correction data to the rover through a reliable data link (network, radio, etc.).
- The rover receives the data from the base and synchronously observes and collects the carrier phase data of GNSS satellites.
- The rover calculates a precise position with the carrier phase data received from GNSS satellites and correction data received from base station.

Normally, RTK can be divided into: traditional RTK and network RTK:

- **Traditional RTK:** The transmission link of differential correction data in traditional RTK is usually provided by wireless radio stations, and the distance between base stations and rover is typically limited to 20–30 km due to significant decrease in the positioning accuracy at larger distances.

- **Network RTK:** It is a real-time positioning technology that combines data from multiple base stations for error calculation and correction, and provides spatial location services to users. It requires data sharing between system centers and multiple reference stations through the network. Network RTK technology overcomes the shortcomings of traditional RTK technology, such as strong correlation between positioning accuracy and distance, and frequent installation of reference stations during operation.

## 2.1. Traditional RTK Technology and Network RTK Technology

In the traditional RTK technology service model, the transmission link of the correction data is usually provided by a radio station, and the distance between the base station and the rover is generally within 30 km. Because the measurement deviation is related to the distance, the orbit deviation, ionosphere deviation and troposphere deviation will be different. If the distance exceeds this, the positioning accuracy of the rover will drop significantly.

In order to overcome the limitations of the traditional RTK technology with a single rover, network RTK technology came into being. Network RTK technology is a real-time positioning technology that combines data from multiple base stations to calculate and correct errors and provide spatial location services to users. It requires the system center to share data from multiple base stations through the network. Network RTK technology overcomes the shortcomings of traditional RTK technology, such as small operating range, strong correlation between positioning accuracy and distance, and the need to frequently set up base stations during operation.

Currently, the most common method of network RTK is VRS (Virtual Reference Station). In this case, the rover sends the position to the network data processing center in the standard NMEA0183 GGA message mode. The network data processing center will create a VRS near the rover and calculate a set of corrections, and then send them back to the rover, which then calculates a high-precision position.

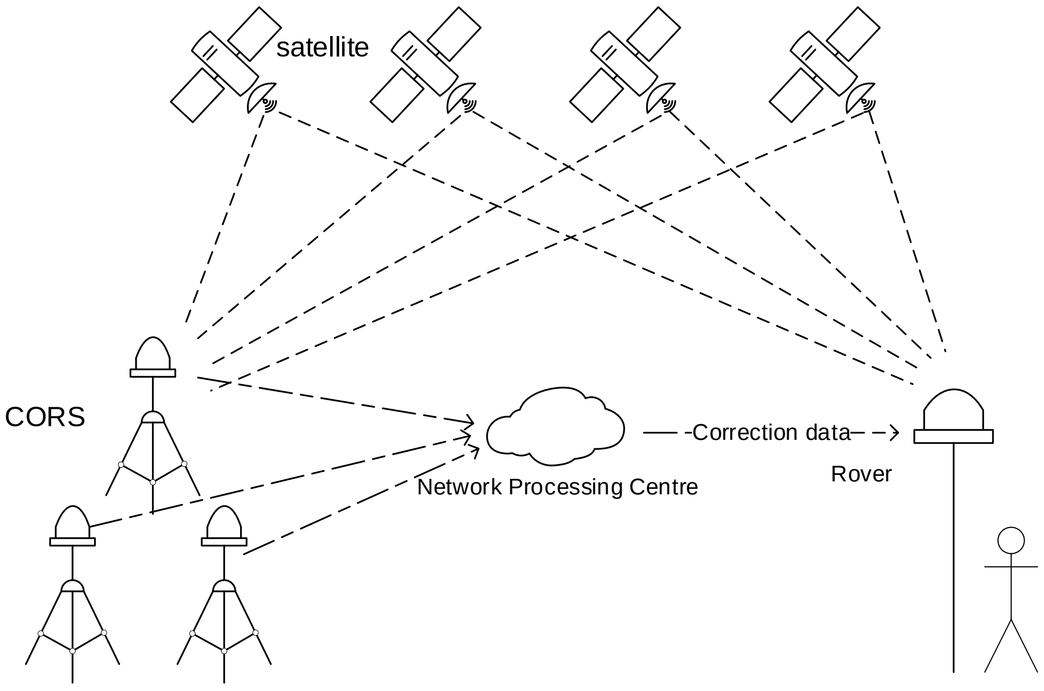


**Figure 2: Network RTK Working Principle**

## 2.2. Correction Data Format

Radio Technical Commission for Maritime (RTCM) standards are used internationally for Differential Global Navigation Satellite Systems (DGNSS), and like conventional DGNSS, the standards also apply to RTK systems, particularly those developed by RTCM Specialized Committee 104. For information, see https://www.rtcm.org.

The two correction data format standard protocols used by the RTK scheme:

- RTCM 10402.3: Recommended standards for differential GNSS service. It is used throughout the world for differential satellite navigation systems, both at sea and over land.
- RTCM 10403.3: Differential GNSS services of Version 3 + Amendment 3 (August 19, 2022). A more effective alternative to RTCM 10402.3.

RTCM 10403.x is more efficient than RTCM 10402.x, but version 3.x messages are not compatible with version 2.x.

# 3 RTK and NTRIP

High-precision RTK positioning requires constant injection of base station correction data into the GNSS rover, which is typically accomplished via the Internet using the NTRIP protocol.

NTRIP is an application-level protocol that supports streaming of Global Navigation Satellite System (GNSS) data over the Internet. NTRIP is a general-purpose stateless protocol based on the Hypertext Transfer Protocol (HTTP). Use NTRIP protocol to send GNSS data streams, mainly including data streams in RTCM data format and GGA data streams in NMEA format, as well as GNSS data streams in other formats.

NTRIP supports the following ways to access the Internet:

- Mobile IP networks such as GSM, GPRS, EDGE, UMTS, LTE and 5G
- WIFI network
- Wired IP Network

**NOTE**

It is not required to use the NTRIP protocol while developing and debugging the RTK function. To observe the positioning effect of the module entering the RTK solution, you can directly utilize the TCP/UDP protocol or other methods to inject the relevant base station's correction data into the high-precision GNSS module.

## 3.1. Composition of NTRIP System

NTRIP protocol currently has two versions: NTRIP v 1.0 and NTRIP v2.0. The main difference is the data transmission format. The v2.0 version follows the standard HTTP protocol specification. For detailed differences, see the RTCM 104 series specification documents. Considering that most of the current actual applications are v1.0 versions of the protocol, the following introduction defaults to NTRIP v 1.0 version.

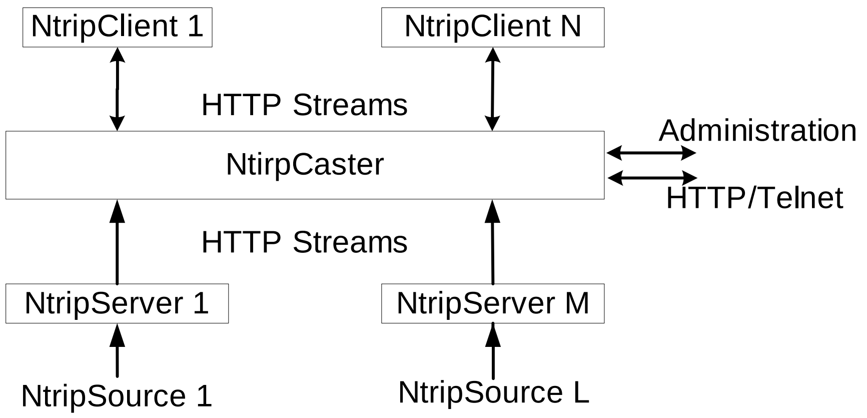


**Figure 3: NTRIP System Components**

- **NtripSource**: generates data streams at a specific location.
- **NtripServer**: transfers the data streams from a source to the NtripCaster.
- **NtripCaster**: data distribution center for receiving and sending GNSS data streams.
- **NtripClient**: where the NtripCaster sends the correction data of the specific location to after logging into NtripCaster.

NtripSource and NtripServer are generally integrated into a GNSS base station, which generates raw observation data of the current specific location and then sends it to NtripCaster through the internet network.

NtripCaster is generally a server that is responsible for receiving and forwarding GNSS data. Usually NtripCaster has two working modes:

- According to different mounting points, the original observation data of the corresponding base station is directly sent to the rover.
- First, multiple actual base stations are virtualized into a virtual reference station (VRS) that best matches the rover through an algorithm, and then the correction data is sent to the rover.

NtripClient is usually a GNSS rover. After logging into NtripCaster, it sends its own coordinates to NtripCaster. NtripCaster selects or generates raw observation data of a specific location and then sends it to NtripClient. In this way, RTK high-precision positioning can be achieved on the GNSS rover.

## 3.2. Communication Between NtripServer and NtripCaster

### 3.2.1. Communication Interaction Process

Before using a TCP/IP connection to transfer GNSS data to NtripCaster, NtripServer must send an Ntrip message to obtain access rights and specify a mount point. The detailed format is as follows:

```
SOURCE <password> /<mountpoint> <CR><LF>

Source-Agent: QNTRIP< product|comment ><CR><LF>

STR: <STR-string><CR><LF>

<CR><LF>

<data>
```

<password>: password for NtripCaster
<mountpoint>: Caster mountpoint for the Source.
<STR-string>: Data fields 3 to n of source-table record type “STR”, optional
<product|comment>: Information about the source agent

Example:
NtripServer sends request information:

```
SOURCE pwd123/QGNSS_MP

Source-Agent: NTRIP NtripServerCMD/1.0<CR><LF>
```

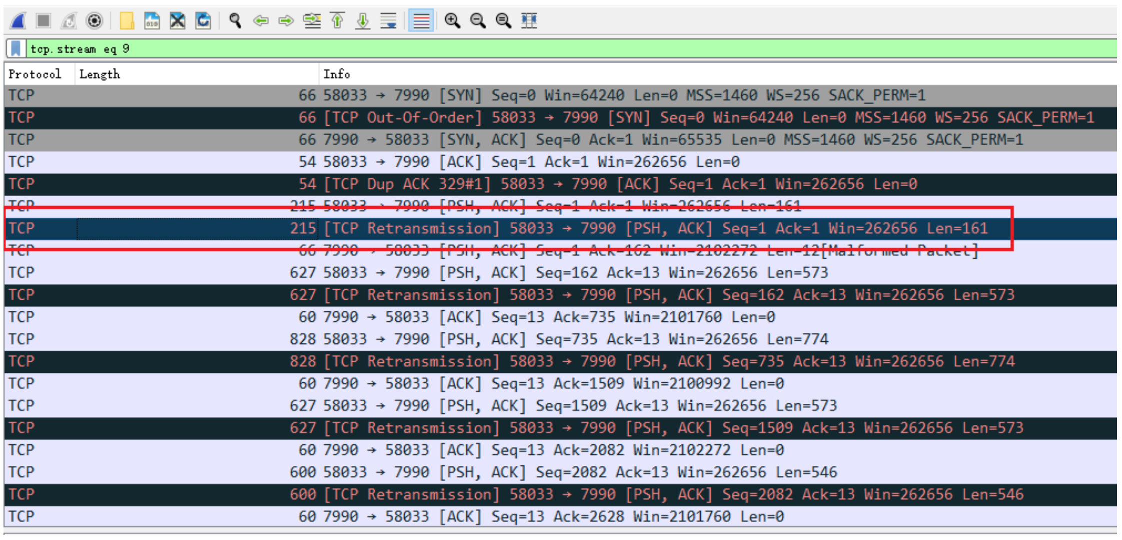
If the password is correct, NtripCaster will respond:

```
ICY 200 OK
```

NtripCaster accepts the data after sending the message " ICY 200 OK ". NtripCaster will only accept data from cryptographically authenticated sources.

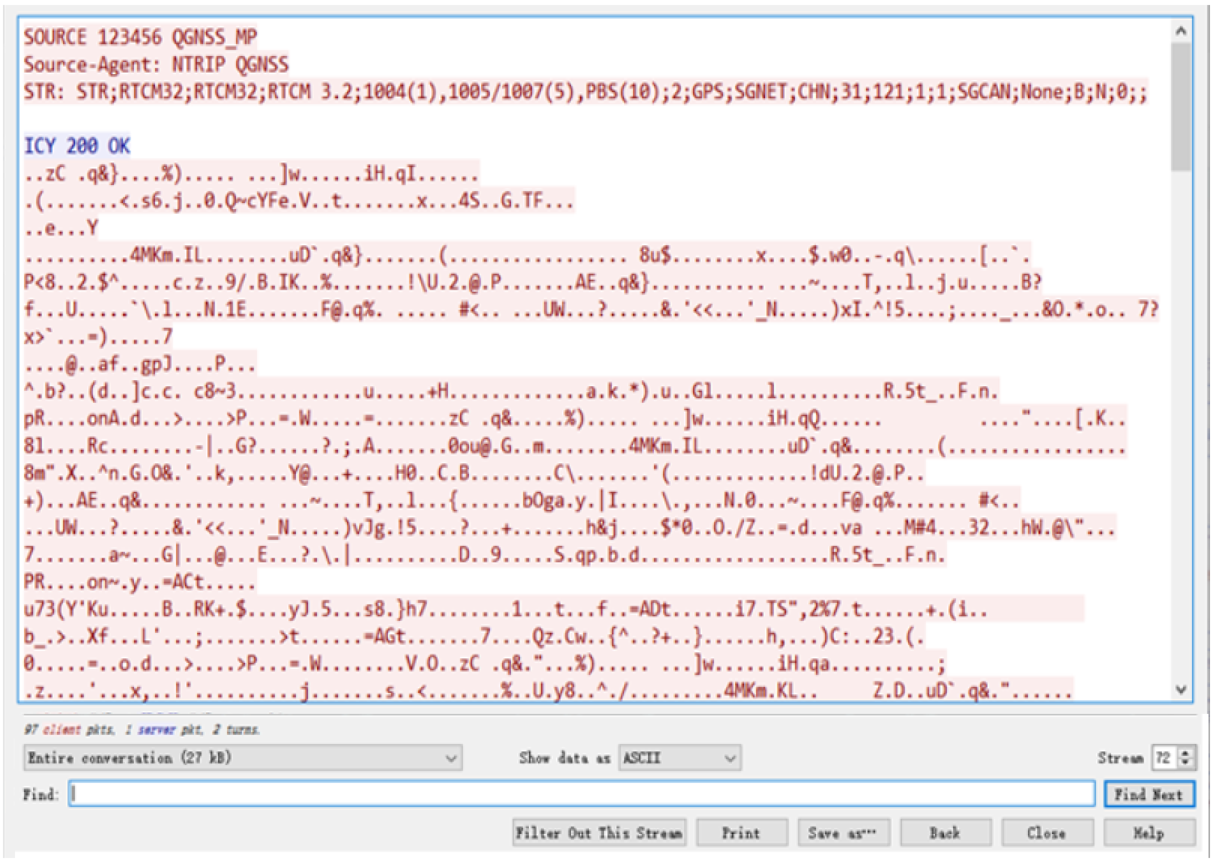
## 3.2.2. NTRIP Packets Captured via Wireshark

Use the Wireshark tool to capture the interaction data between NtripServer and NtripCaster, as follows :



```
tcp.stream eq 9

Protocol Length                                   Info
TCP                                   66 58033 → 7990 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
TCP                                   66 [TCP Out-Of-Order] 58033 → 7990 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
TCP                                   66 7990 → 58033 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
TCP                                   54 58033 → 7990 [ACK] Seq=1 Ack=1 Win=262656 Len=0
TCP                                   54 [TCP Dup ACK 329#1] 58033 → 7990 [ACK] Seq=1 Ack=1 Win=262656 Len=0
TCP                                  215 58033 → 7990 [PSH, ACK] Seq=1 Ack=1 Win=262656 Len=161
TCP                                  215 [TCP Retransmission] 58033 → 7990 [PSH, ACK] Seq=1 Ack=1 Win=262656 Len=161
TCP                                   66 7990 → 58033 [PSH, ACK] Seq=1 Ack=162 Win=2102272 Len=12[Malformed Packet]
TCP                                  627 58033 → 7990 [PSH, ACK] Seq=162 Ack=13 Win=262656 Len=573
TCP                                  627 [TCP Retransmission] 58033 → 7990 [PSH, ACK] Seq=162 Ack=13 Win=262656 Len=573
TCP                                   60 7990 → 58033 [ACK] Seq=13 Ack=735 Win=2101760 Len=0
TCP                                  828 58033 → 7990 [PSH, ACK] Seq=735 Ack=13 Win=262656 Len=774
TCP                                  828 [TCP Retransmission] 58033 → 7990 [PSH, ACK] Seq=735 Ack=13 Win=262656 Len=774
TCP                                   60 7990 → 58033 [ACK] Seq=13 Ack=1509 Win=2100992 Len=0
TCP                                  627 58033 → 7990 [PSH, ACK] Seq=1509 Ack=13 Win=262656 Len=573
TCP                                  627 [TCP Retransmission] 58033 → 7990 [PSH, ACK] Seq=1509 Ack=13 Win=262656 Len=573
TCP                                   60 7990 → 58033 [ACK] Seq=13 Ack=2082 Win=2102272 Len=0
TCP                                  600 58033 → 7990 [PSH, ACK] Seq=2082 Ack=13 Win=262656 Len=546
TCP                                  600 [TCP Retransmission] 58033 → 7990 [PSH, ACK] Seq=2082 Ack=13 Win=262656 Len=546
TCP                                   60 7990 → 58033 [ACK] Seq=13 Ack=2628 Win=2101760 Len=0
```

**Figure 4: NtripServer Data Interaction Capture**



```
SOURCE 123456 QGNSS_MP
Source-Agent: NTRIP QGNSS
STR: STR;RTCM32;RTCM32;RTCM 3.2;1004(1),1005/1007(5),PBS(10);2;GPS;SGNET;CHN;31;121;1;1;SGCAN;None;B;N;0;;

ICY 200 OK
..zC .q&}....%)..... ...]w......iH.qI......
.(.......<.s6.j..0.Q~cYFe.V..t.......x...4S..G.TF...
..e...Y
..........4MKm.IL........uD`.q&}.......(................. 8u$........x....$.w0..-.q\......[..`.
P<8..2.$^.....c.z..9/.B.IK..%.......!\U.2.@.P.......AE..q&}........... ...~....T,..1..j.u.....B?
f...U.....`\.1...N.1E.......F@.q%. ..... #<.. ...UW...?.....&.'<<...'_N.....)xI.^!5....;...._...&0.*.o.. 7?
x>`...=).....7
....@..af..gpJ....P...
^.b?..(d..]c.c. c8~3............u.....+H.............a.k.*).u..Gl.....l..........R.5t_..F.n.
pR....onA.d...>....>P...=.W.....=.......zC .q&.....%)..... ...]w......iH.qQ......         ...."....[.K..
81....Rc........-|..G?......?.;.A.......0ou@.G..m........4MKm.IL........uD`.q&........(................
8m".X..^n.G.O&.'..k,.....Y@...+....H0..C.B........C\.......'(.............!dU.2.@.P..
+)...AE..q&........... ...~....T,..1...{......bOga.y.|I....\.,...N.0...~....F@.q%....... #<..
...UW...?.....&.'<<...'_N.....)vJg.!5....?...+.......h&j....$*0..O./Z..=.d...va ...M#4...32...hW.@\"...
7.......a~...G|...@...E...?.\.|..........D..9.....S.qp.b.d.................R.5t_..F.n.
PR....on~.y..=ACt.....
u73(Y'Ku.....B..RK+.$....yJ.5...s8.}h7........1...t...f..=ADt......i7.TS",2%7.t......+.(i..
b_.>..Xf...L'...;.......>t......=AGt.......7....Qz.Cw..{^..?+..}......h,...)C:..23.(.
0.....=..o.d...>....>P...=.W........V.O..zC .q&."...%)..... ...]w......iH.qa..........;
.z....'...x,..!'..........j.......s..<.......%..U.y8..^./.........4MKm.KL..     Z.D..uD`.q&."......
```

97 client pkts, 1 server pkt, 2 turns.

Entire conversation (27 kB)

Show data as ASCII

Stream 72

Find:

Figure 5: NtripServer Flow Tracking

## 3.3. Communication Between NtripClient and NtripCaster

### 3.3.1. Communication Interaction Process

NtripClient needs to send a HTTP request message to NtripCaster, including the mount point, user agent and authentication. The detailed format is as follows :

```
GET /<mountpoint> HTTP/1.0 <CR><LF>

User-Agent: QNTRIP < product|comment > <CR><LF>

Accept: */* <CR><LF>

Connection: close <CR><LF>

Authorization: Basic < user:password > <CR><LF><CR><LF>
```

<mountpoint>: Caster mount point of the request source
<product|comment>: information about the user agent that initiated the request
<user:password>: username and password converted to base64 ,

Example:
The client sends a request message:

```
GET /AUTO HTTP/1.0

User-Agent: QNTRIP Quectel -GNSS

Accept: */*

Connection: close

Authorization: Basic UUxfSEYxOjEyMzQ1Ng==

NtripCaster receives a right request, it will send a response message to NtripClient :

ICY 200 OK
```

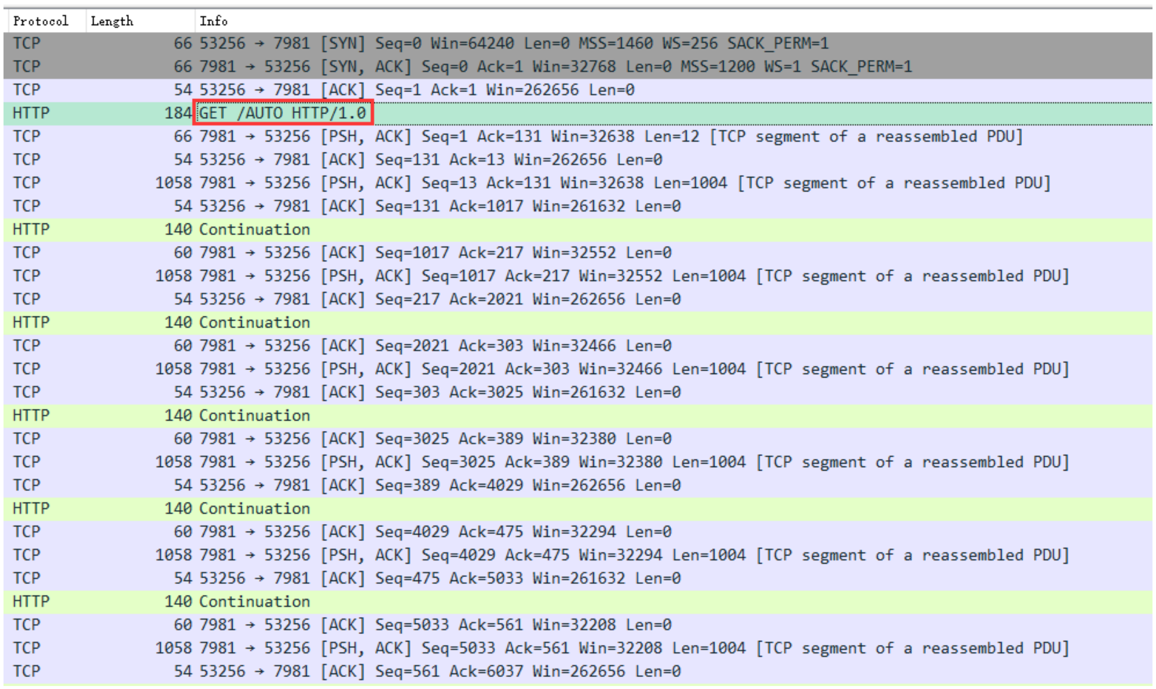
NtripClient receives the correct response from NtripCaster, it can directly send the current GGA sentence message to NtripCaster :

```
$GNGGA,062517.00,3149. 30207339,N ,11706.92065527,E,1,20,0.90,95.4583,M,-5.0226,M,,*6
```

After receiving the GGA message sent by NtripClient, NtripCaster will send the correction data in RTCM format corresponding to the mount point (for example: AUTO) to NtripClient.

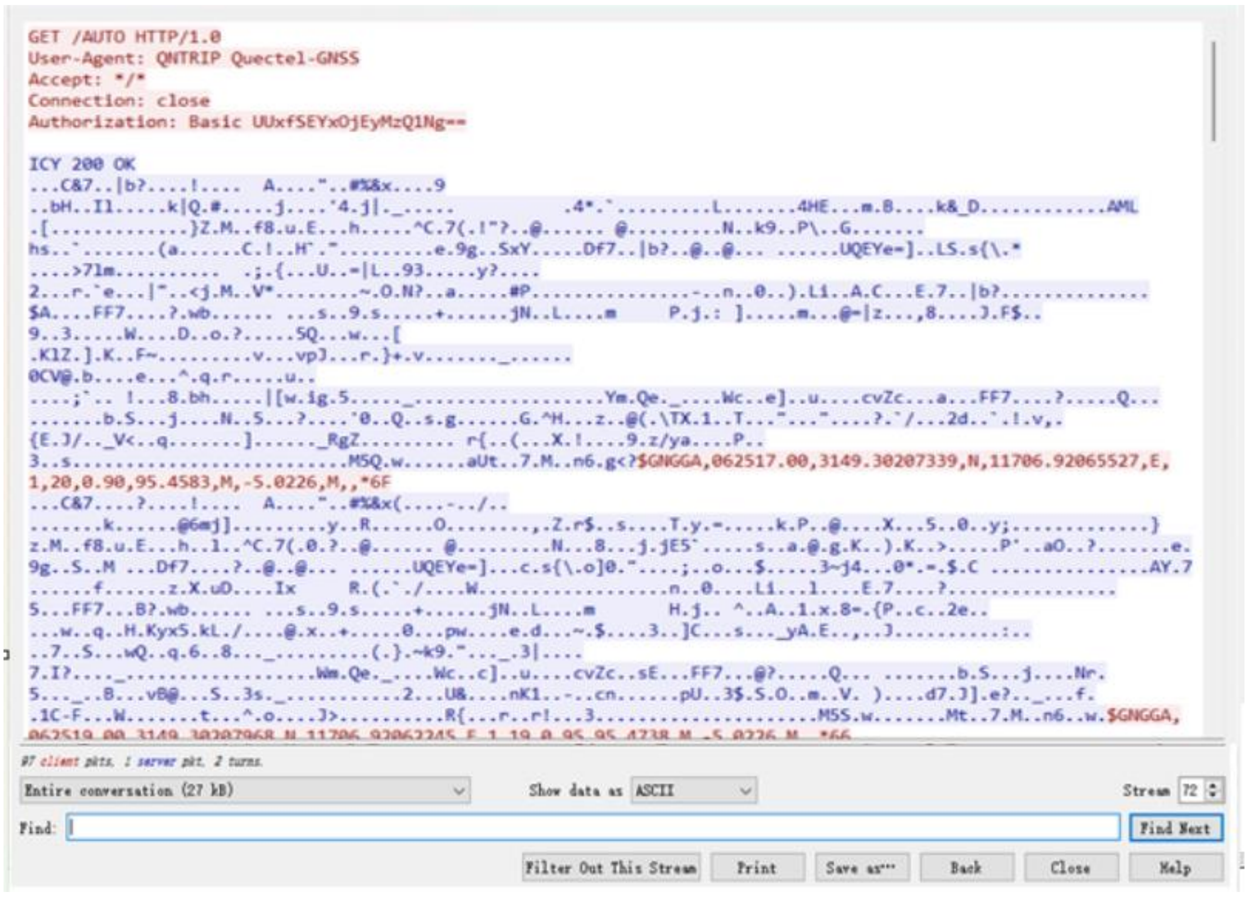
### 3.3.2. NTRIP Packets Captured via Wireshark

Use the Wireshark tool to capture the interaction data between NtripClient and NtripCaster, as follows :



```
Protocol  Length  Info
TCP           66 53256 → 7981 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
TCP           66 7981 → 53256 [SYN, ACK] Seq=0 Ack=1 Win=32768 Len=0 MSS=1200 WS=1 SACK_PERM=1
TCP           54 53256 → 7981 [ACK] Seq=1 Ack=1 Win=262656 Len=0
HTTP         184 GET /AUTO HTTP/1.0
TCP           66 7981 → 53256 [PSH, ACK] Seq=1 Ack=131 Win=32638 Len=12 [TCP segment of a reassembled PDU]
TCP           54 53256 → 7981 [ACK] Seq=131 Ack=13 Win=262656 Len=0
TCP         1058 7981 → 53256 [PSH, ACK] Seq=13 Ack=131 Win=32638 Len=1004 [TCP segment of a reassembled PDU]
TCP           54 53256 → 7981 [ACK] Seq=131 Ack=1017 Win=261632 Len=0
HTTP         140 Continuation
TCP           60 7981 → 53256 [ACK] Seq=1017 Ack=217 Win=32552 Len=0
TCP         1058 7981 → 53256 [PSH, ACK] Seq=1017 Ack=217 Win=32552 Len=1004 [TCP segment of a reassembled PDU]
TCP           54 53256 → 7981 [ACK] Seq=217 Ack=2021 Win=262656 Len=0
HTTP         140 Continuation
TCP           60 7981 → 53256 [ACK] Seq=2021 Ack=303 Win=32466 Len=0
TCP         1058 7981 → 53256 [PSH, ACK] Seq=2021 Ack=303 Win=32466 Len=1004 [TCP segment of a reassembled PDU]
TCP           54 53256 → 7981 [ACK] Seq=303 Ack=3025 Win=261632 Len=0
HTTP         140 Continuation
TCP           60 7981 → 53256 [ACK] Seq=3025 Ack=389 Win=32380 Len=0
TCP         1058 7981 → 53256 [PSH, ACK] Seq=3025 Ack=389 Win=32380 Len=1004 [TCP segment of a reassembled PDU]
TCP           54 53256 → 7981 [ACK] Seq=389 Ack=4029 Win=262656 Len=0
HTTP         140 Continuation
TCP           60 7981 → 53256 [ACK] Seq=4029 Ack=475 Win=32294 Len=0
TCP         1058 7981 → 53256 [PSH, ACK] Seq=4029 Ack=475 Win=32294 Len=1004 [TCP segment of a reassembled PDU]
TCP           54 53256 → 7981 [ACK] Seq=475 Ack=5033 Win=261632 Len=0
HTTP         140 Continuation
TCP           60 7981 → 53256 [ACK] Seq=5033 Ack=561 Win=32208 Len=0
TCP         1058 7981 → 53256 [PSH, ACK] Seq=5033 Ack=561 Win=32208 Len=1004 [TCP segment of a reassembled PDU]
TCP           54 53256 → 7981 [ACK] Seq=561 Ack=6037 Win=262656 Len=0
```

**Figure 6: NtripClient Data Interaction Capture**



```
GET /AUTO HTTP/1.0
User-Agent: QNTRIP Quectel-GNSS
Accept: */*
Connection: close
Authorization: Basic UUxfSEYxOjEyMzQ1Ng==

ICY 200 OK
...C&7..|b?....!....  A...."..#%&x....9
..bH..I1.....k|Q.#.....j....'4.j|._.....           .4*.`.........L.......4HE...m.B....k&_D............AML
.[..............}Z.M..f8.u.E...h.....^C.7(.!"?..@...... @..........N..k9..P\..G.......
hs..`.......(a......C.!..H`."..........e.9g..SxY.....Df7..|b?..@..@... .......UQEYe=]..LS.s{\.*
....>71m..........  .;.{...U..=|L..93.....y?....
2...r.`e...|"..<j.M..V*.........~.O.N?..a.....#P..................-..n..0..).Li..A.C...E.7..|b?...............
$A....FF7....?.wb...... ...s..9.s.....+.......jN..L....m     P.j.: ].....m...@=|z...,8....J.F$..
9..3.....W....D..o.?.....5Q...w...[
.K1Z.].K..F~..........v...vpJ...r.}+.v......._......
0CV@.b....e...^.q.r.....u..
....;`.. !...8.bh.....|[w.ig.5....._...................Ym.Qe._....Wc..e]..u....cvZc...a...FF7....?.....Q...
.......b.S...j....N..5...?....'0..Q..s.g......G.^H...z..@(.\TX.1..T..."..."....?.`/...2d..`.!.v,.
{E.J/.._V<..q.......]......_RgZ.......... r{..(...X.!....9.z/ya....P..
3..s...........................M5Q.w.......aUt..7.M..n6.g<?$GNGGA,062517.00,3149.30207339,N,11706.92065527,E,
1,20,0.90,95.4583,M,-5.0226,M,,*6F
...C&7....?....!....  A...."..#%&x(....-../..
........k.......@6mj]..........y..R.......O.........,.Z.r$..s....T.y.=......k.P..@....X...5..0..y;..............}
z.M..f8.u.E...h..1..^C.7(.0.?..@...... @..........N...8...j.jE5`.....s..a.@.g.K..).K..>.....P`..aO..?........e.
9g..S..M ...Df7....?..@..@... .......UQEYe=]...c.s{\.o]0."....;..o...$.....3~j4...0*.=.$.C ................AY.7
......f.......z.X.uD....Ix     R.(.`./....W...................n..0....Li...1....E.7....?.................
5...FF7...B?.wb...... ...s..9.s.....+.......jN..L....m       H.j.. ^..A..1.x.8=.{P..c..2e..
...w..q..H.Kyx5.kL./....@.x..+......0...pw....e.d...~.$....3..]C...s..._yA.E..,..J..............:..
..7..S...wQ..q.6..8..._...........(.}.~k9."..._.3|....
7.I?...._....................Wm.Qe._....Wc..c]..u....cvZc..sE...FF7...@?.....Q... ........b.S...j....Nr.
5..._..B...vB@...S..3s._.............2...U&....nK1..-..cn.......pU..3$.S.O..m..V. )....d7.J].e?.._....f.
.1C-F...W........t...^.o....J>...........R{...r..r!...3.......................M5S.w........Mt..7.M..n6..w.$GNGGA,
062519.00,3149.30207968,N,11706.92062245,E,1,19,0.95,95.4738,M,-5.0226,M,,*66
```

97 client pkts, 1 server pkt, 2 turns.

Entire conversation (27 kB)    Show data as ASCII    Stream 72

Find:

Figure 7: NtripClient Flow Tracking

# 4 High-precision Positioning Through RTK

There are two ways to achieve high-precision RTK positioning:

1. Directly purchase the package and account of the service operator to quickly achieve RTK high-precision positioning. In this process, the service operator provides base stations and forwards correction data.
2. You build your own RTK base stations and correction data forwarding systems.

This chapter will focus on how to manually build a verification test system for RTK high-precision solutions.

## 4.1. Test Environment and Antenna

The antenna of the RTK base station needs to be installed in a completely open sky place, which cannot be blocked by other objects. It is recommended to install it on the roof of a building. Then use a dedicated RF cable to introduce the signal into the base station equipment.

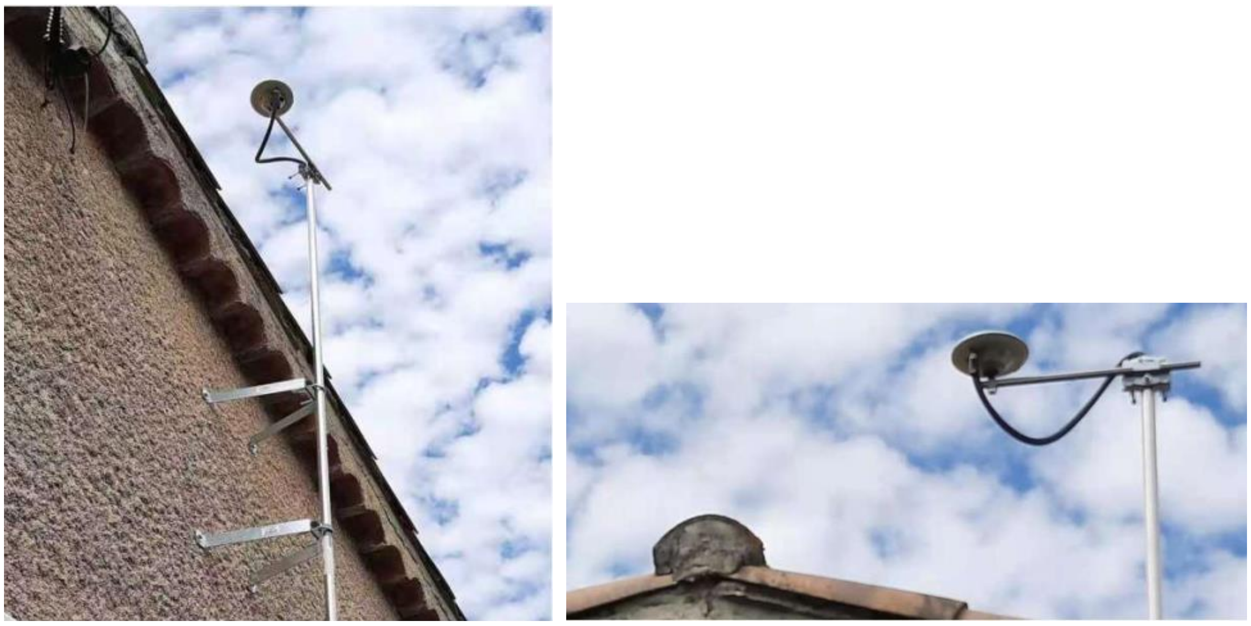


**Figure 8: Base Station Antenna Installation**

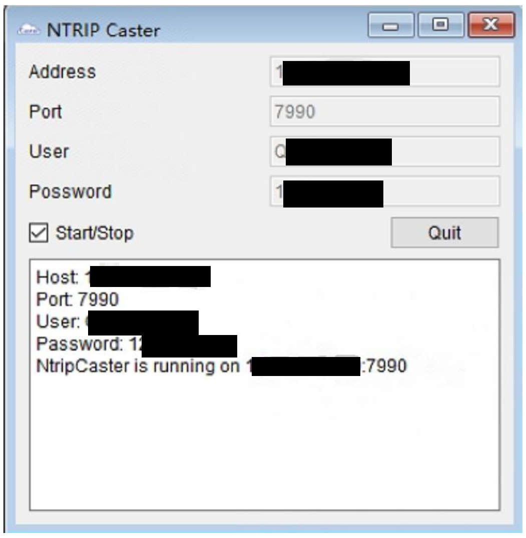
For rover and base station antennas, Quectel Evaluation Kit provides corresponding antennas for rover and base station antennas. The specific antenna form is subject to the evaluation kit.

## 4.2. High-precision Positioning Through QGNSS Tool

Quectel's QGNSS tool has implemented the NtripClient, NtripServer, and NtripCaster functions. By running the QGNSS tool on the RTK base station, data forwarder, and RTK rover , and configuring the correct parameters, the high-precision GNSS positioning function can be achieved. For more details about the QGNSS tool, see document [1] user guide.

### 4.2.1. Start NtripCaster

NtripCaster is a correction data forwarder. Its function is to collect the raw data of the RTK base station and forward it to NtripClient in real time. For the startup process of NtripCaster, see *Chapter 5.6.1* in document [1] user guide. The interface after successful startup is as follows:



Address: 1[redacted]
Port: 7990
User: Q[redacted]
Possword: 1[redacted]
☑ Start/Stop

```
Host: 1[redacted]
Port: 7990
User: [redacted]
Password: 12[redacted]
NtripCaster is running on 1[redacted]:7990
```

Figure 9: NtripCaster Successfully Started

**NOTE**

In order to facilitate users to verify the RTK function, Quectel provides the testing for NtripCaster:
IP address: 172.29.104.32
Port number: 7990
Username: QL_NTRIP
Password: 123456
If users build NtripCaster by themselves, please make sure that the IP address and port can be accessed from the external network.

### 4.2.2. Establishing an RTK Base Station

The RTK base station can provide the original correction data source for the rover. Take Quectel's LC29H (BS) module as an example, connect the LC29H (BS) module serial port to the QGNSS tool (NtripServer) as an RTK base station.

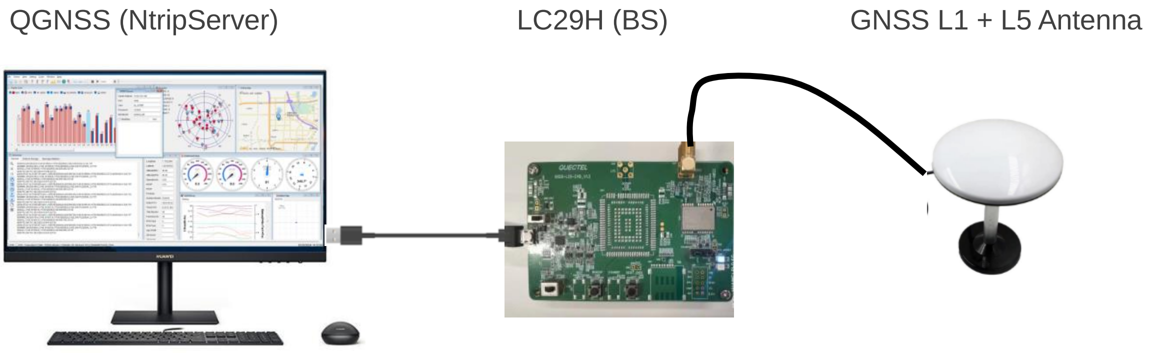
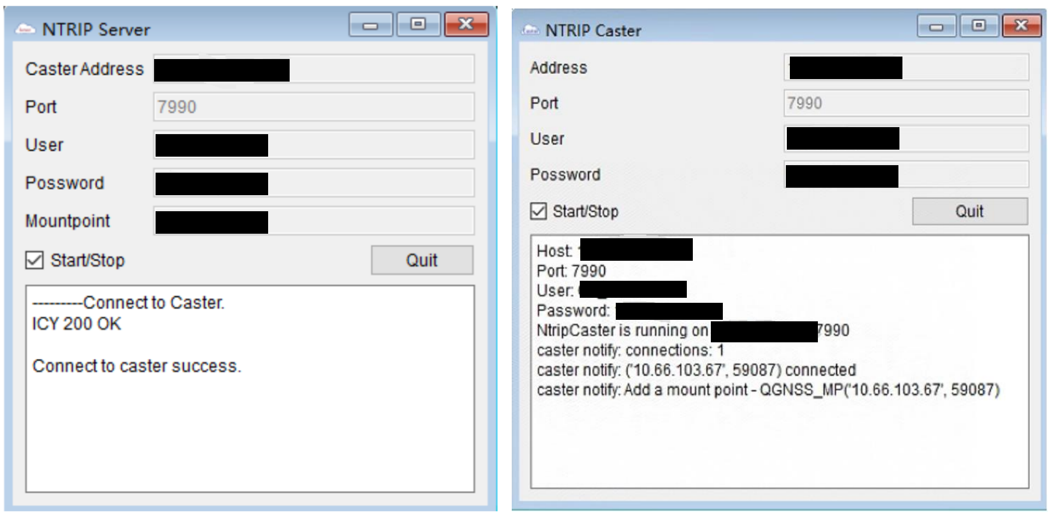


**Figure 10: RTK Base Station Connection**

Fill in the relevant parameters of NtripCaster in the NtripServer interface and connect to NtripCaster. NtripServer provides NtripCaster with the original data generated by LC29H (BS). For details on the operation of the QGNSS tool, see Chapter 5.6 in the *document [1] user guide*. After successfully connecting to NtripCaster, the interface is as follows:



NTRIP Server

| | |
|---|---|
| Caster Address | [redacted] |
| Port | 7990 |
| User | [redacted] |
| Password | [redacted] |
| Mountpoint | [redacted] |

☑ Start/Stop Quit

```
---------Connect to Caster.
ICY 200 OK

Connect to caster success.
```

NTRIP Caster

| | |
|---|---|
| Address | [redacted] |
| Port | 7990 |
| User | [redacted] |
| Password | [redacted] |

☑ Start/Stop Quit

```
Host: [redacted]
Port: 7990
User: [redacted]
Password: [redacted]
NtripCaster is running on [redacted]7990
caster notify: connections: 1
caster notify: ('10.66.103.67', 59087) connected
caster notify: Add a mount point - QGNSS_MP('10.66.103.67', 59087)
```

**Figure 11: NtripServer Successfully Connects to NtripCaster**

**NOTE**

When LC29H (BS) is used as a rover, you need to manually set a known location. If the known location information cannot be predicted, you can use the module's Survey-in function to obtain the current location information.

### 4.2.3. RTK Rover Obtains High-precision Positioning Information

The high-precision positioning module can be used as an RTK rover. Take LC29H (EA) as an example. Connect the LC29H (EA) module serial port to the QGNSS (NtripClient) tool :

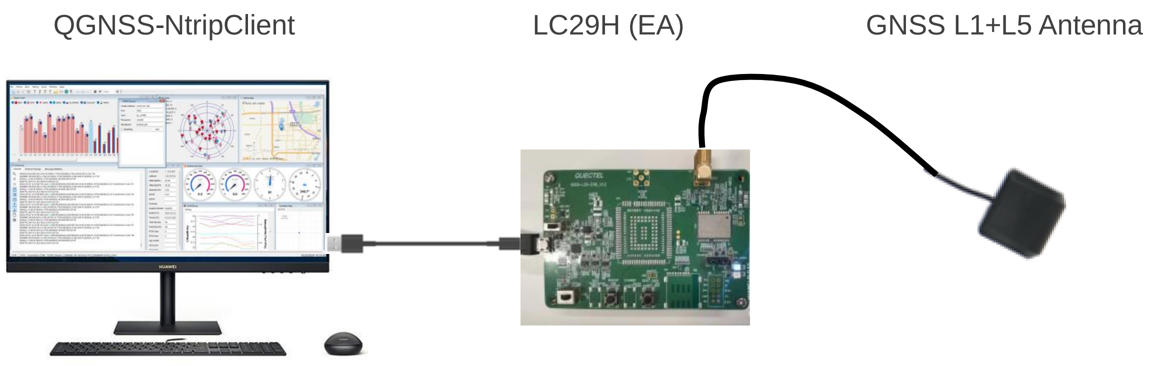
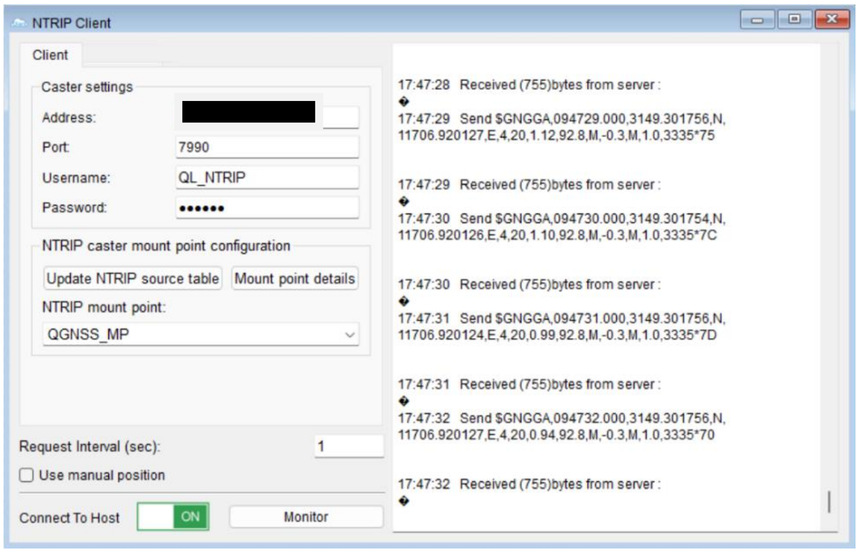


**Figure 12: RTK Base Station Connection**

After the LC29H (EA) module is successfully positioned, open the NtripClient interface and fill in the relevant parameters of NtripCaster. After successfully connecting to NtripCaster, the RTK rover (LC29H (EA) module) will soon obtain high-precision positioning information. For details on the operation of the QGNSS tool, see *Chapter 5.6.3* in [document [1] user guide](#).



Caster settings

| | |
|---|---|
| Address: | [redacted] |
| Port: | 7990 |
| Username: | QL_NTRIP |
| Password: | •••••• |

NTRIP caster mount point configuration

NTRIP mount point: QGNSS_MP

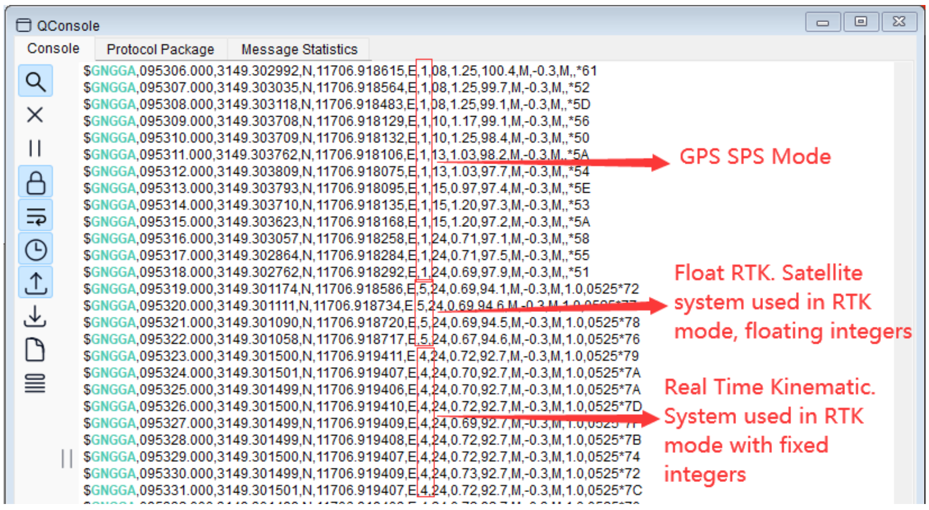
Request Interval (sec): 1

☐ Use manual position

Connect To Host: ON

```
17:47:28  Received (755)bytes from server :
�
17:47:29  Send $GNGGA,094729.000,3149.301756,N,
11706.920127,E,4,20,1.12,92.8,M,-0.3,M,1.0,3335*75


17:47:29  Received (755)bytes from server :
�
17:47:30  Send $GNGGA,094730.000,3149.301754,N,
11706.920126,E,4,20,1.10,92.8,M,-0.3,M,1.0,3335*7C


17:47:30  Received (755)bytes from server :
�
17:47:31  Send $GNGGA,094731.000,3149.301756,N,
11706.920124,E,4,20,0.99,92.8,M,-0.3,M,1.0,3335*7D


17:47:31  Received (755)bytes from server :
�
17:47:32  Send $GNGGA,094732.000,3149.301756,N,
11706.920127,E,4,20,0.94,92.8,M,-0.3,M,1.0,3335*70


17:47:32  Received (755)bytes from server :
�
```

**Figure 13: Successfully Connected to NtripCaster and Obtained Correction Data**


QConsole
Console
Protocol Package
Message Statistics
GPS SPS Mode
Float RTK. Satellite system used in RTK mode, floating integers
Real Time Kinematic. System used in RTK mode with fixed integers

```
$GNGGA,095306.000,3149.302992,N,11706.918615,E,1,08,1.25,100.4,M,-0.3,M,,*61
$GNGGA,095307.000,3149.303035,N,11706.918564,E,1,08,1.25,99.7,M,-0.3,M,,*52
$GNGGA,095308.000,3149.303118,N,11706.918483,E,1,08,1.25,99.1,M,-0.3,M,,*5D
$GNGGA,095309.000,3149.303708,N,11706.918129,E,1,10,1.17,99.1,M,-0.3,M,,*56
$GNGGA,095310.000,3149.303709,N,11706.918132,E,1,10,1.25,98.4,M,-0.3,M,,*50
$GNGGA,095311.000,3149.303762,N,11706.918106,E,1,13,1.03,98.2,M,-0.3,M,,*5A
$GNGGA,095312.000,3149.303809,N,11706.918075,E,1,13,1.03,97.7,M,-0.3,M,,*54
$GNGGA,095313.000,3149.303793,N,11706.918095,E,1,15,0.97,97.4,M,-0.3,M,,*5E
$GNGGA,095314.000,3149.303710,N,11706.918135,E,1,15,1.20,97.3,M,-0.3,M,,*53
$GNGGA,095315.000,3149.303623,N,11706.918168,E,1,15,1.20,97.2,M,-0.3,M,,*5A
$GNGGA,095316.000,3149.303057,N,11706.918258,E,1,24,0.71,97.1,M,-0.3,M,,*58
$GNGGA,095317.000,3149.302864,N,11706.918284,E,1,24,0.71,97.5,M,-0.3,M,,*55
$GNGGA,095318.000,3149.302762,N,11706.918292,E,1,24,0.69,97.9,M,-0.3,M,,*51
$GNGGA,095319.000,3149.301174,N,11706.918586,E,5,24,0.69,94.1,M,-0.3,M,1.0,0525*72
$GNGGA,095320.000,3149.301111,N,11706.918734,E,5,24,0.69,94.6,M,-0.3,M,1.0,0525*77
$GNGGA,095321.000,3149.301090,N,11706.918720,E,5,24,0.69,94.5,M,-0.3,M,1.0,0525*78
$GNGGA,095322.000,3149.301058,N,11706.918717,E,5,24,0.67,94.6,M,-0.3,M,1.0,0525*76
$GNGGA,095323.000,3149.301500,N,11706.919411,E,4,24,0.72,92.7,M,-0.3,M,1.0,0525*79
$GNGGA,095324.000,3149.301501,N,11706.919407,E,4,24,0.70,92.7,M,-0.3,M,1.0,0525*7A
$GNGGA,095325.000,3149.301499,N,11706.919406,E,4,24,0.70,92.7,M,-0.3,M,1.0,0525*7A
$GNGGA,095326.000,3149.301500,N,11706.919410,E,4,24,0.72,92.7,M,-0.3,M,1.0,0525*7D
$GNGGA,095327.000,3149.301499,N,11706.919409,E,4,24,0.69,92.7,M,-0.3,M,1.0,0525*7F
$GNGGA,095328.000,3149.301499,N,11706.919408,E,4,24,0.72,92.7,M,-0.3,M,1.0,0525*7B
$GNGGA,095329.000,3149.301500,N,11706.919407,E,4,24,0.72,92.7,M,-0.3,M,1.0,0525*74
$GNGGA,095330.000,3149.301499,N,11706.919409,E,4,24,0.73,92.7,M,-0.3,M,1.0,0525*72
$GNGGA,095331.000,3149.301501,N,11706.919407,E,4,24,0.72,92.7,M,-0.3,M,1.0,0525*7C
```

**Figure 14: Successfully Achieved High-precision Positioning**

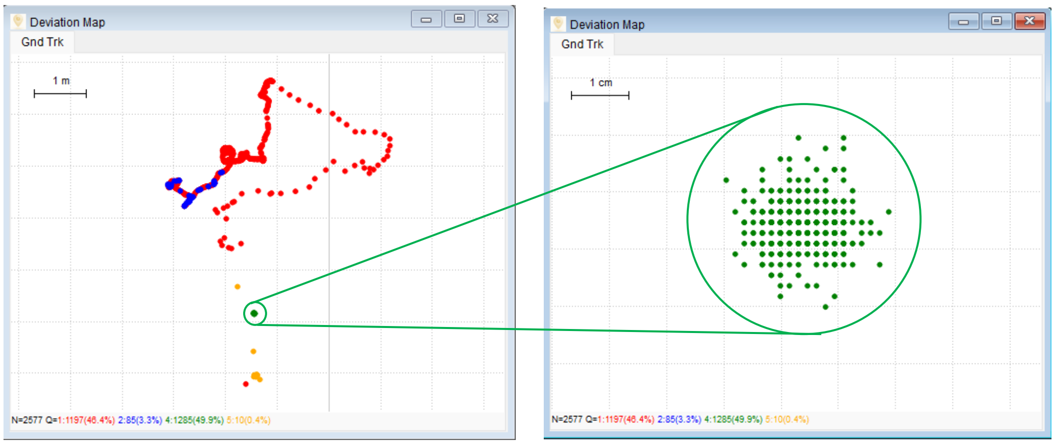


**Figure 15: Positioning Deviation in Single-point Solution (in Red) and Fixed Solution (in Green) in Static**

**NOTE**

The GNSS module positioning mode can be obtained from GGA sentence and RMC sentence.

## 4.3. High-precision Positioning Through EVB

Quectel provides a GNSS evaluation board with a built-in MCU. The EVB supports NtripClient and NtripServer functions by default. Therefore, users can also achieve high-precision positioning functions through the EVB. For more information about the EVB, see *document [2] user guide*.

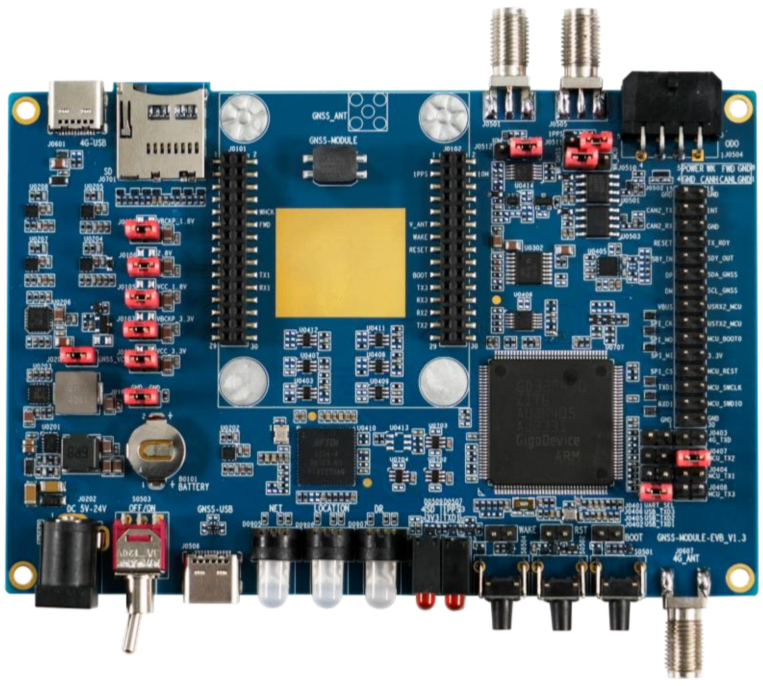
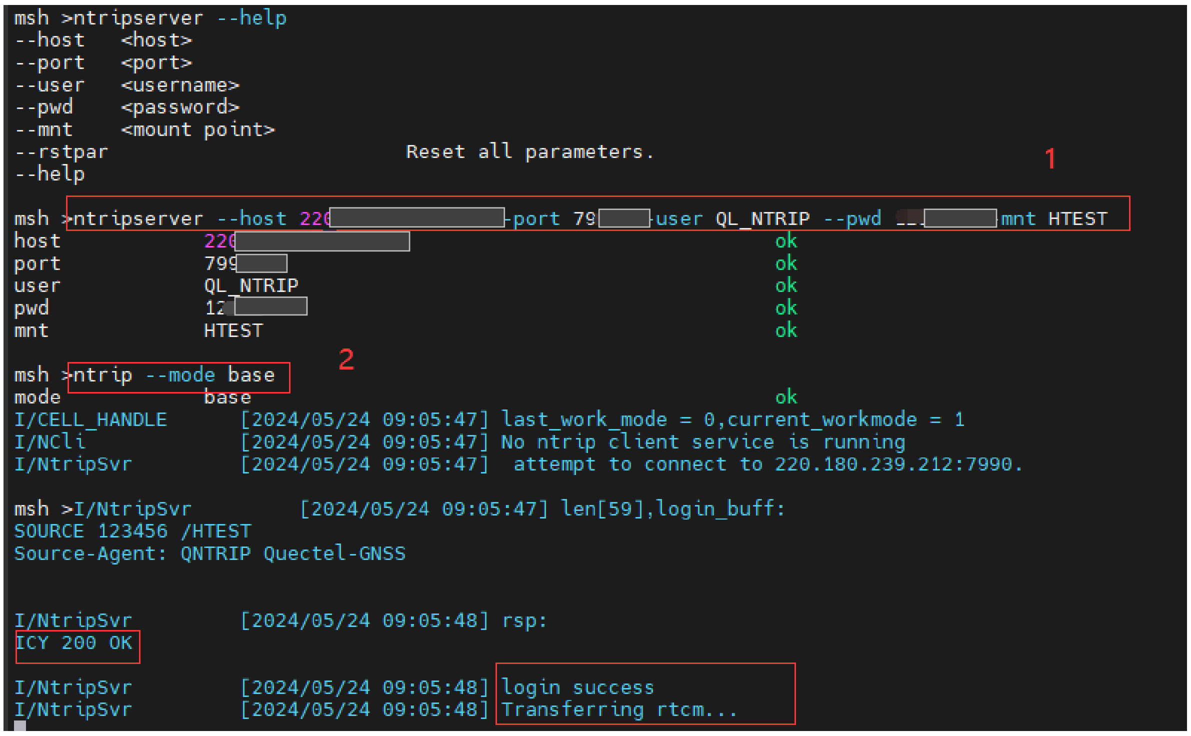


**Figure 16: Evaluation Board**

Since the EVB is equipped with a LTE module, it cannot provide the NtripCaster forwarding function. Therefore, the QGNSS-NtripCaster tool is still used to provide the forwarding function. For related operations, see *Chapter 4.2.1 Start NtripCaster*.

### 4.3.1. Establishing an RTK Base Station

The RTK base station can provide the original correction data source for the rover. The EVB provides the ntripserver command, which can configure the EVB to the NtripServer mode (RTK base station). For detailed configuration steps, see Chapter 8.7 of *document [2] user guide*.



```
msh >ntripserver --help
--host    <host>
--port    <port>
--user    <username>
--pwd     <password>
--mnt     <mount point>
--rstpar                         Reset all parameters.
--help

msh >ntripserver --host 220[redacted]-port 79[redacted]-user QL_NTRIP --pwd [redacted]mnt HTEST
host            220[redacted]                              ok
port            799[redacted]                              ok
user            QL_NTRIP                                 ok
pwd             12[redacted]                               ok
mnt             HTEST                                    ok

msh >ntrip --mode base
mode            base                                     ok
I/CELL_HANDLE      [2024/05/24 09:05:47] last_work_mode = 0,current_workmode = 1
I/NCli             [2024/05/24 09:05:47] No ntrip client service is running
I/NtripSvr         [2024/05/24 09:05:47]  attempt to connect to 220.180.239.212:7990.

msh >I/NtripSvr         [2024/05/24 09:05:47] len[59],login_buff:
SOURCE 123456 /HTEST
Source-Agent: QNTRIP Quectel-GNSS


I/NtripSvr         [2024/05/24 09:05:48] rsp:
ICY 200 OK

I/NtripSvr         [2024/05/24 09:05:48] login success
I/NtripSvr         [2024/05/24 09:05:48] Transferring rtcm...
```

**Figure 17: Configured in NtripServer Mode and Successfully Connected to NtripCaster**

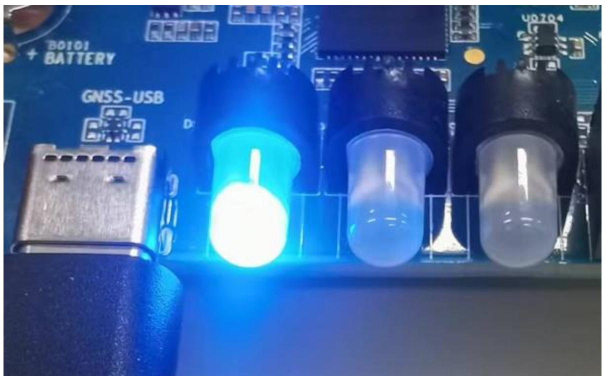
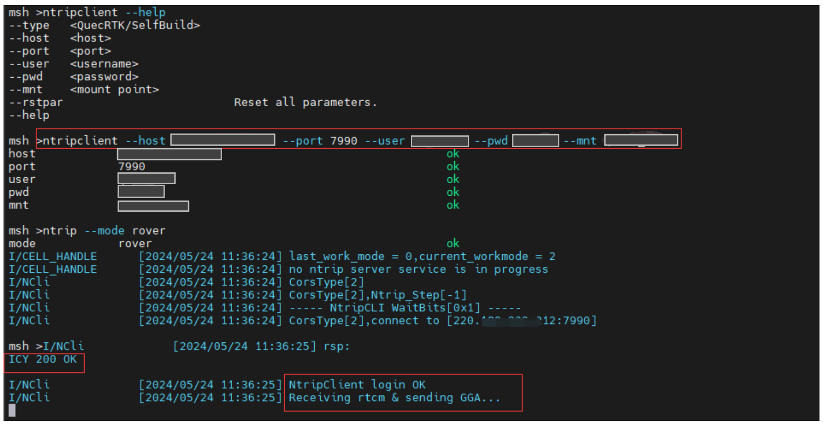


**Figure 18: D0905 Indicator Status in Base Station Mode**

## 4.3.2. RTK Rover Obtains High-precision Information

The RTK rover can be a EVB. Through the ntripclient command, we can configure the EVB to ntripclient mode (RTK rover). In this mode, you can also choose self-built base station mode / standard NTRIP mode. For detailed configuration steps, see Chapter 8.7 in document [2] user guide.



```
msh >ntripclient --help
--type   <QuecRTK/SelfBuild>
--host   <host>
--port   <port>
--user   <username>
--pwd    <password>
--mnt    <mount point>
--rstpar                        Reset all parameters.
--help

msh >ntripclient --host [redacted] --port 7990 --user [redacted] --pwd [redacted] --mnt [redacted]
host            [redacted]                              ok
port            7990                                    ok
user            [redacted]                              ok
pwd             [redacted]                              ok
mnt             [redacted]                              ok

msh >ntrip --mode rover
mode            rover                                   ok
I/CELL_HANDLE      [2024/05/24 11:36:24] last_work_mode = 0,current_workmode = 2
I/CELL_HANDLE      [2024/05/24 11:36:24] no ntrip server service is in progress
I/NCli             [2024/05/24 11:36:24] CorsType[2]
I/NCli             [2024/05/24 11:36:24] CorsType[2],Ntrip_Step[-1]
I/NCli             [2024/05/24 11:36:24] ----- NtripCLI WaitBits[0x1] -----
I/NCli             [2024/05/24 11:36:24] CorsType[2],connect to [220.[redacted]12:7990]

msh >I/NCli             [2024/05/24 11:36:25] rsp:
ICY 200 OK

I/NCli             [2024/05/24 11:36:25] NtripClient login OK
I/NCli             [2024/05/24 11:36:25] Receiving rtcm & sending GGA...
```

**Figure 19: Self-built Base Station Mode/Standard NTRIP Mode**

When the GNSS module successfully enters the high-precision positioning solution, the indicator lights on the EVB are as follows:

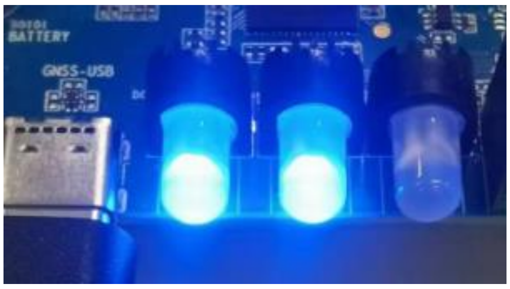


**Figure 20: GNSS Module Enters High-precision Positioning State**

# 5 Common Issues and Troubleshooting

## 5.1. Issues Related to NtripCaster Connection

1. If there is no response after NtripClient sends the request data, you need to check the following points:
    a) Whether the IP address and Port of NtripCaster are valid.
    b) Check whether your SIM card is restricted from accessing certain servers or has data traffic access restrictions if using cellular wireless for communication.
    c) If the requested NtripCaster server address is a domain name, please check whether the domain name is resolved correctly.

2. If the NtripCaster server returns an error warning after NtripClient sends the request data, you need to check the following points:
    a) Are the carriage returns and line feeds in the request message correct.
    b) Whether the username and password fields are included correctly.
    c) Whether the mountpoint is correct.

## 5.2. Why the GNSS Module Fails to Enter RTK Solution Mode

1. The algorithm of the GNSS high-precision position module needs to be activated through an authentication code. The module is activated by default. If the module loses the default authentication code due to other reasons, please contact Quectel Technical Support.
2. Ensure that the GNSS antenna is in an unobstructed environment and that the frequency band supported by the GNSS antenna matches the frequency band of the GNSS module.
3. Check whether the correction data received by the GNSS module has serious cycle slip problems
4. Check whether the number of satellites searched by the GNSS module and the number of satellites participating in positioning calculation are too few. It is generally recommended that more than 10 satellites participate in positioning calculation. (Refer to GSV and GSA messages in NMEA 0183 protocol).
5. Check whether the C/$N_0$ value of the satellite detected by the GNSS module is too low. It is generally recommended to be greater than 30 dB-Hz. (Refer to the GSV message in the NMEA 0183 protocol).
6. Check the constellations and frequency bands of the injected correction data. When the injected correction data lacks certain constellations or frequency bands, the module may be unable to enter RTK fixed solution.

7. Check the distance between the base station and rover. When the distance between the base station and the rover exceeds 30km, the module may be unable to enter the RTK fixed solution.
8. Check the timeliness of the injected correction data. If there is a delay in the correction data, the module may be unable to enter the RTK fixed solution.
9. Check whether the position in GGA messages uploaded by the rover deviates greatly from the actual position.

# 6 Reference Codes

## 6.1. NtripClient C Codes

The following is a C language version of the NtripClient example, which can be compiled and run directly in the Linux environment.

```c
#include < stdio.h >
#include < stdlib.h >
#include < string.h >
#include < netdb.h >
#include <sys/ types.h >
#include <sys/ socket.h >
#include < arpa / inet.h >
#include <unistd.h>

#define NTRIP_IP           "10.66.103.109"
#define NTRIP_PORT         7981
#define NTRIP_USERNAME     "QL_HF1"
#define NTRIP_PASSWORD     "123456"
#define NTRIP_MOUNT_POINT  "Auto"


#define GGA_STRING         "$GPGGA,115713.000,3149.301528,N,11706.920684,E,1,17,0.88,98.7,M,-3.6,M,,*58\r\n"
#define ICY_200_OK         "ICY 200 OK"

static const char base64EncodingTable [64] = {
  'A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P',
  'Q','R','S','T','U','V','W','X','Y','Z','a','b','c','d','e','f',
  'g','h','i','j','k','l','m','n','o','p','q','r','s','t','u','v',
  'w','x','y','z','0','1','2','3','4','5','6','7','8','9','+','/'
};

// format user:pwd  --> encodedata
static int Ql_Ntrip_Encode(char *buf, int size, const char *user, const char *pwd)
{
    unsigned char tmpbuf[3];
    char *output = buf;
    int sep = 0;
    int fill = 0;
    int bytes = 0;
    int actual_len = size - 1;

    int total_len = strlen(user) + strlen(pwd) + 1;
    unsigned char * total_buf = malloc(total_len + 1);
    if(NULL == total_buf)
    {
        printf("malloc buf failed");
        return bytes;
    }
    unsigned char *tmpptr = total_buf;

    memset(tmpptr,0,total_len + 1);

    //Combining Strings
    strncpy(tmpptr,user,strlen(user));
    tmpptr[strlen(user)] = ':';
    strcat(tmpptr,pwd);

    while(*tmpptr)
    {
        for(int i = 0;i < 3;i++)
        {
            if(*tmpptr)
            {
                tmpbuf[i] = *(tmpptr++);
            }
            else
            {
                tmpbuf[i] = 0;
                fill += 1;
            }
        }

        if(output-buf < actual_len)
        {
            *(output++) = base64EncodingTable[(tmpbuf[0] & 0xFC) >> 2];
            if(output-buf < actual_len)
            {
                *(output++) = base64EncodingTable[((tmpbuf[0] & 0x03) << 4)|((tmpbuf [1] & 0xF0) >> 4)];
                if(output-buf < actual_len)
                {
                    if(fill == 2)
                    {
                        *(output++) = '=';
                    }
                    else
                    {
                        *(output++) = base64EncodingTable[((tmpbuf[1] & 0x0F) << 2)|((tmpbuf [2] & 0xC0) >> 6)];
                    }
                    if(output-buf < actual_len)
                    {
                        if(fill >= 1)
                        {
                            *(output++) = '=';
                        }
                        else
                        {
                            *(output++) = base64EncodingTable[tmpbuf[2] & 0x3F];
                        }
                    }
                }
            }
        }
        bytes += 4;
    }

          if(output-buf < size)
          {
              *output = 0;
          }

          if(NULL != total_buf)
          {
              free(total_buf);
          }

          return bytes;
      }

      int Ql_Ntrip_BuileAuthMsg(char *buf,int size, char *user, char* pwd, char* ip,char* mount_point)
      {
          int len = 0;
          char b64[512] = {0};

          if (NULL == buf)
          {
              return 0;
          }

          len = Ql_Ntrip_Encode(b64,sizeof(b64),user,pwd);
          if (len <= 0)
          {
              printf("base64 encode fial.\r\n");
              return len;
          }

          len = snprintf(buf,size,"GET /%s HTTP/1.0\r\n"
                          "User-Agent: Quectel GNSS/v1.0\r\n"
                          "Host: %s\r\n"
                          "Accept: */*\r\n"
                          "Connection: close\r\n"
                          "Authorization: Basic %s\r\n\r\n",
                          mount_point,ip,b64);
          return len;
      }


      int main(void) {

          int client_fd, cnt,ret,i;
          unsigned char buf[2048];
          struct sockaddr_in serverAddress;

          if((client_fd = socket(AF_INET, SOCK_STREAM, 0)) == -1) {
              printf("socket fail,exit.");
              exit(1);
          }

          serverAddress.sin_family = AF_INET;
          serverAddress.sin_port = htons(NTRIP_PORT);
          serverAddress.sin_addr.s_addr = inet_addr(NTRIP_IP);
          bzero(&(serverAddress.sin_zero), 8);

          if(connect(client_fd, (struct sockaddr *)&serverAddress, sizeof(struct sockaddr)) == -1) {
              printf("connect fail,exit");
              exit(1);
          }
          printf("connected.\r\n");

          memset(buf,0,sizeof(buf));

          ret = Ql_Ntrip_BuileAuthMsg(buf,sizeof(buf),NTRIP_USERNAME,NTRIP_PASSWORD,NTRIP_IP,NTRIP_MOUNT_POINT);
```

```c

        if(ret > 0)
        {
            send(client_fd, buf, strlen(buf), 0);
            printf("send auth msg to NTRIP caster:%s",buf);

            cnt = recv(client_fd, buf, sizeof(buf), 0);
            buf[cnt] = '\0';
            printf("receive from server:%s\n", buf);
            if(strstr(buf,ICY_200_OK))
            {
                printf("login to NTRIP caster successfully\r\n");
            }
            else
            {
                printf("login to NTRIP caster failed,exit.\r\n");
                exit(1);
            }
        }
        else
        {
            printf("build auth message fail,exit.\r\n");
            exit(1);
        }

        while (1) {
            send(client_fd, GGA_STRING, strlen(GGA_STRING), 0);
            printf("\r\n\r\nsend GGA to NTRIP caster:%s",GGA_STRING);

            cnt = recv(client_fd, buf, sizeof(buf), 0);

            printf("Receive RTCM data from server:%d bytes.(should send the RTCM data to GNSS RTK module)",cnt);
            for (i=0; i < cnt; i++)
            {
                if(i%48 == 0) printf("\r\n");
                printf("%02X ",buf[i]);
            }
            sleep( 1);
    }
    close( client_fd );
    return 0;
    }
```

## 6.2. NtripClient Python Codes

The reference code for Python 3.8:

```python
import socket
import base64
import time

# NTRIP Caster information
NTRIP_IP = "10.66.103.109"
NTRIP_PORT = 7981
NTRIP_USERNAME = "QL_HF1"
NTRIP_PASSWORD = "123456"
NTRIP_MOUNT_POINT = "AUTO"

GGA_STRING = "$GNGGA,115713.000,3149.301528,N,11706.920684,E,1,17,0.88,98.7,M,-3.6,M,,*58\r\n"

# Create a TCP socket and connect to the NTRIP Caster
try:
    sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    sock.connect((NTRIP_IP, NTRIP_PORT))
    print("Connected.")
except socket.error as err:
    print("Connect fail,exit.")
    exit(1)

# Send login request
Authorization = (NTRIP_USERNAME+ ":" + NTRIP_PASSWORD)
Authorization = base64.b64encode(Authorization.encode()).decode()
requestHead =  f"""GET /{NTRIP_MOUNT_POINT} HTTP/1.0\r\nUser-Agent: Quectel GNSS\r\nHost: {NTRIP_IP}\r\nAccept: */*\r\nConnection: close\r\nAuthorization: Basic {Authorization}\r\n\r\n"""
sock.send(requestHead.encode())
print("send auth msg to NTRIP caster:%s" % requestHead)

# Receive and parse the NTRIP login response
response = sock.recv(1024)
print("receive from server:%s" % response.decode())
if "ICY 200 OK" not in response.decode():
    print("login to NTRIP caster fail,exit.")
    exit(1)
print('login to NTRIP caster successful.')

# Send GGA data to NtripCaster
while True:
    sock.send(GGA_STRING.encode())
    print("\r\n\r\nSend GGA to NTRIP caster:%s" % GGA_STRING)
    response = sock.recv(4096)
    print("Receive RTCM data from server:%d bytes.(should send the RTCM data to GNSS RTK module)" % len(response))
hex_string = ' ' .join (format(b, '02x' ) for b in response).upper()
    print ( hex_string )
      time.sleep (1)
```

## 6.3. NtripClient Cellular Module AT Command Example

AT commands to implement NtripClient on the EC600U series module:

```
AT+QIOPEN=1,0,"TCP","106.14.222.3",1883,0,1
OK
+QIOPEN: 0,0
AT+QISEND=0
> GET /RTCM33_GRCEJ HTTP/1.0
User-Agent: NTRIP PostmanRuntime/7.31.0
Authorization: Basic cXVlY3RlbDoxMjM=
Accept: */*
Cache-Control: no-cache
Postman-Token: b1a7196c-1b22-4e27-bf58-244e3a6935c6
Host: rtksouth.quectelcn.com:1883
Accept-Encoding: gzip, deflate, br
Connection: keep-alive

SEND OK
+QIURC: "recv",0,12
ICY 200 OK                          //NtripClient is logged in successfully
AT+QISEND=0
> $GNGGA,071516.000,3149. 301159,N ,11706.919870,E,1,16,0.95,70.2,M,-0.3,M,,*59

SEND OK
+QIURC: "recv",0,1075               //Received 1075 bytes of differential correction data.
// Send the received correction data directly to the GNSS module.
// Continue uploading the latest GGA data and continue this process.
```

**NOTE**

GGA sentences need to include "\r\n" when sending

# 7 Appendix References

Table 2: Related Documents

| Document Name |
|---|
| [1] Quectel_QGNSS_User Guide |
| [2] Quectel_GNSS Module EVB_User Guide |

Table 3: Terms and Abbreviations

| Abbreviation | Description |
|---|---|
| CORS | Continuously Operating Reference Stations |
| DGNSS | Differential Global Navigation Satellite System |
| GNSS | Global Navigation Satellite System |
| GPS | Global Positioning System |
| GGA | Global Positioning System Fix Data |
| IP | Internet Protocol |
| NMEA | NMEA (National Marine Electronics Association) 0183 Interface Standard |
| NTRIP | Networked Transport of RTCM via Internet Protocol |
| MCU | Microcontroller Unit |
| RTCM | Radio Technical Commission for Maritime Services |
| RTK | Real-Time Kinematic |
| TCP | Transmission Control Protocol |
| UDP | User Datagram Protocol |
| VRS | Virtual Reference Station |