Transfer depth data using sockets

Our camera can capture the depth data, we can transfer data from client to server using sockets. You can test it step-by-step.

1. Two compute module on Linux(x86, x64, ARM), one is client, the other is server, we must ensure that the two networks are working.

   Client ip address:

```
orbbec@orbbec ~/Desktop $ ifconfig
ens33     Link encap:Ethernet  HWaddr 00:0c:29:b4:65:da
          inet addr:192.168.62.129  Bcast:192.168.62.255  Mask:255.255.255.0
          inet6 addr: fe80::51e1:8c72:a08e:6261/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:565939 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2808002 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:41616650 (41.6 MB)  TX bytes:4150495755 (4.1 GB)
          Interrupt:19 Base address:0x2024

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:844721 errors:0 dropped:0 overruns:0 frame:0
          TX packets:844721 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:1262098429 (1.2 GB)  TX bytes:1262098429 (1.2 GB)
```

   Server ip address:

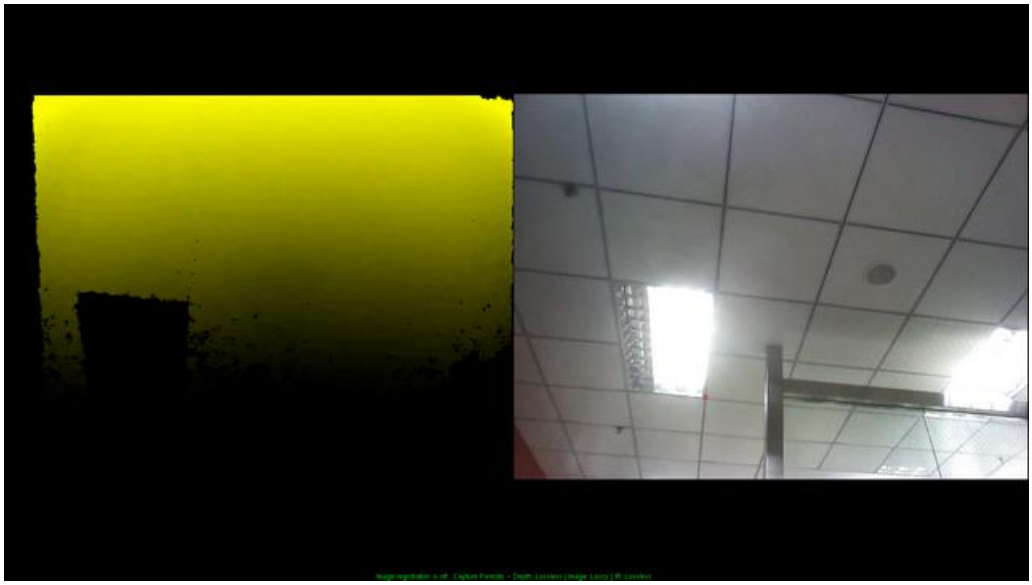
```
orbbec@ubuntu:~/Desktop$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:8e:d2:b9
          inet addr:192.168.62.130  Bcast:192.168.62.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe8e:d2b9/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3836647 errors:0 dropped:0 overruns:0 frame:0
          TX packets:681314 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:5691274781 (5.6 GB)  TX bytes:45135303 (45.1 MB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:162 errors:0 dropped:0 overruns:0 frame:0
          TX packets:162 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:14112 (14.1 KB)  TX bytes:14112 (14.1 KB)
```

   We can use ping command to test the network:

```
orbbec@orbbec ~/Desktop $ ping 192.168.62.130
PING 192.168.62.130 (192.168.62.130) 56(84) bytes of data.
64 bytes from 192.168.62.130: icmp_seq=1 ttl=64 time=0.980 ms
64 bytes from 192.168.62.130: icmp_seq=2 ttl=64 time=1.95 ms
64 bytes from 192.168.62.130: icmp_seq=3 ttl=64 time=1.37 ms
64 bytes from 192.168.62.130: icmp_seq=4 ttl=64 time=0.867 ms
64 bytes from 192.168.62.130: icmp_seq=5 ttl=64 time=2.24 ms
64 bytes from 192.168.62.130: icmp_seq=6 ttl=64 time=1.56 ms
64 bytes from 192.168.62.130: icmp_seq=7 ttl=64 time=1.71 ms
64 bytes from 192.168.62.130: icmp_seq=8 ttl=64 time=1.97 ms
64 bytes from 192.168.62.130: icmp_seq=9 ttl=64 time=0.908 ms
```

2. Connect one Orbbec camera to the client compute module, and ensure that the camera is working. You can use the OpenNI tool NiViewer to test. If you don't know how to use the Orbbec camera on Linux, you should see our OpenNI SDK document.

Test camera on client:



3. Compile the client program on Client compute module, copy the Depth-socket-tcp.zip to the client. Before compiling, you should modify the **Src/client.cpp** file

```
55  int send_depth()
56  {
57      ///sockfd
58      int sock_cli = socket(AF_INET, SOCK_STREAM, 0);
59
60      struct sockaddr_in servaddr;
61      memset(&servaddr, 0, sizeof(servaddr));
62      servaddr.sin_family = AF_INET;
63      servaddr.sin_port = htons(MYPORT);
64      servaddr.sin_addr.s_addr = inet_addr("192.168.62.130");
65      //servaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
66      //servaddr.sin_addr.s_addr = inet_addr("192.168.1.108");
67
68
69      if (connect(sock_cli, (struct sockaddr *)&servaddr, sizeof(servaddr)) < 0)
70      {
71          perror("connect");
72          exit(1);
73      }
74
75      int needSend = sizeof(Node);
76      char *buffer = (char*)malloc(needSend);
77      memcpy(buffer,myNode,needSend);
78
```

Server IP

Compile the client program:

unzip Depth-socket-tcp.zip
cd Depth-socket-tcp/
cd Platform/Linux(YOUR PLATFORM)
make client

If you compile it ok, you can see the client executable file on your platform directory.

```
orbbec@orbbec ~/Desktop/Depth-socket-tcp/Platform/Linux(x64) $ ls -al
total 644
drwxr-xr-x 3 orbbec orbbec   4096 Mar  9 14:37 .
drwxr-xr-x 5 orbbec orbbec   4096 Feb 23 14:02 ..
-rwxr-xr-x 1 orbbec orbbec  76176 Mar  9 14:37 client
-rw-r--r-- 1 orbbec orbbec  51710 Mar  9 14:37 libOpenNI2.jni.so
-rw-r--r-- 1 orbbec orbbec 481860 Mar  9 14:37 libOpenNI2.so
-rw-r--r-- 1 orbbec orbbec   1007 Mar  9 11:10 makefile
drwxr-xr-x 3 orbbec orbbec   4096 Mar  9 14:37 OpenNI2
-rw-r--r-- 1 orbbec orbbec    458 Mar  9 14:37 OpenNI.ini
-rw-r--r-- 1 orbbec orbbec  22877 Mar  9 14:37 org.openni.jar
```

4. Compile the server program on Server compute module, copy the Depth-socket-tcp.zip to the server.

   Compile the server program:

unzip Depth-socket-tcp.zip
cd Depth-socket-tcp/
cd Platform/Linux(YOUR PLATFORM)
make server

   If you compile it ok, you can see the server executable file on your platform directory.

```
orbbec@ubuntu:~/Desktop/Depth-socket-tcp/Platform/Linux(x64)$ ls -al
total 28
drwxrwxr-x 2 orbbec orbbec  4096 Mar  8 22:47 .
drwxrwxr-x 5 orbbec orbbec  4096 Feb 23 14:02 ..
-rw-rw-r-- 1 orbbec orbbec  1007 Mar  9  2017 makefile
-rwxrwxr-x 1 orbbec orbbec 15990 Mar  8 22:47 server
```

5. Run the server executable program on your server compute module.

```
orbbec@ubuntu:~/Desktop/Depth-socket-tcp/Platform/Linux(x64)$ ./server
```

6. Run the client executable program on your client compute module.

```
orbbec@orbbec ~/Desktop/Depth-socket-tcp/Platform/Linux(x64) $ ./client
Warning: USB events thread - failed to set priority. This might cause loss of data...
---------------------Hotkey----------------------
1.Press s to send one frame depth data.
2.Press q to quit.
```

7. Send depth data from client to server, Press s hotkey on your client program, and check the server program receive message.

```
orbbec@orbbec ~/Desktop/Depth-socket-tcp/Platform/Linux(x64) $ ./client
Warning: USB events thread - failed to set priority. This might cause loss of data...
--------------------Hotkey---------------------
1.Press s to send one frame depth data.
2.Press q to quit.
s
>>>>>>>>>>Send info test>>>>>>>>>>
index:1
depth[5000] : 1702 depth[10000]: 1782 depth[15000]: 1840 depth[20000]: 1933
depth[25000]: 2025 depth[30000]: 2125 depth[35000]: 2237 depth[40000]:    0
s
>>>>>>>>>>Send info test>>>>>>>>>>
index:4525
depth[5000] : 1719 depth[10000]: 1773 depth[15000]: 1870 depth[20000]: 1944
depth[25000]: 2001 depth[30000]: 2152 depth[35000]:    0 depth[40000]:    0
s
>>>>>>>>>>Send info test>>>>>>>>>>
index:4609
depth[5000] : 1719 depth[10000]: 1773 depth[15000]: 1870 depth[20000]: 1944
depth[25000]: 2001 depth[30000]: 2152 depth[35000]: 2222 depth[40000]:    0
s
>>>>>>>>>>Send info test>>>>>>>>>>
index:4665
depth[5000] : 1719 depth[10000]: 1773 depth[15000]: 1880 depth[20000]: 1944
depth[25000]: 1989 depth[30000]: 2152 depth[35000]: 2222 depth[40000]:    0
s
>>>>>>>>>>Send info test>>>>>>>>>>
index:4712
depth[5000] : 1719 depth[10000]: 1773 depth[15000]: 1880 depth[20000]: 1944
depth[25000]: 1989 depth[30000]: 2152 depth[35000]: 2222 depth[40000]:    0
s
>>>>>>>>>>Send info test>>>>>>>>>>
index:4752
depth[5000] : 1719 depth[10000]: 1773 depth[15000]: 1880 depth[20000]: 1944
depth[25000]: 1989 depth[30000]: 2152 depth[35000]: 2222 depth[40000]:    0
s
>>>>>>>>>>Send info test>>>>>>>>>>
index:4881
depth[5000] :  555 depth[10000]:  629 depth[15000]:  528 depth[20000]:  594
depth[25000]:  679 depth[30000]:  562 depth[35000]:  639 depth[40000]:    0
s
>>>>>>>>>>Send info test>>>>>>>>>>
index:4971
depth[5000] :  578 depth[10000]:  619 depth[15000]:  590 depth[20000]:  636
depth[25000]:  684 depth[30000]:  646 depth[35000]:  704 depth[40000]:    0
```

```
orbbec@ubuntu:~/Desktop/Depth-socket-tcp/Platform/Linux(x64)$ ./server
<<<<<<<<<<Rece info test<<<<<<<<<<
index:1
depth[5000] : 1702 depth[10000]: 1782 depth[15000]: 1840 depth[20000]: 1933
depth[25000]: 2025 depth[30000]: 2125 depth[35000]: 2237 depth[40000]:    0
<<<<<<<<<<Rece info test<<<<<<<<<<
index:4525
depth[5000] : 1719 depth[10000]: 1773 depth[15000]: 1870 depth[20000]: 1944
depth[25000]: 2001 depth[30000]: 2152 depth[35000]:    0 depth[40000]:    0
<<<<<<<<<<Rece info test<<<<<<<<<<
index:4609
depth[5000] : 1719 depth[10000]: 1773 depth[15000]: 1870 depth[20000]: 1944
depth[25000]: 2001 depth[30000]: 2152 depth[35000]: 2222 depth[40000]:    0
<<<<<<<<<<Rece info test<<<<<<<<<<
index:4665
depth[5000] : 1719 depth[10000]: 1773 depth[15000]: 1880 depth[20000]: 1944
depth[25000]: 1989 depth[30000]: 2152 depth[35000]: 2222 depth[40000]:    0
<<<<<<<<<<Rece info test<<<<<<<<<<
index:4712
depth[5000] : 1719 depth[10000]: 1773 depth[15000]: 1880 depth[20000]: 1944
depth[25000]: 1989 depth[30000]: 2152 depth[35000]: 2222 depth[40000]:    0
<<<<<<<<<<Rece info test<<<<<<<<<<
index:4752
depth[5000] : 1719 depth[10000]: 1773 depth[15000]: 1880 depth[20000]: 1944
depth[25000]: 1989 depth[30000]: 2152 depth[35000]: 2222 depth[40000]:    0
<<<<<<<<<<Rece info test<<<<<<<<<<
index:4881
depth[5000] :  555 depth[10000]:  629 depth[15000]:  528 depth[20000]:  594
depth[25000]:  679 depth[30000]:  562 depth[35000]:  639 depth[40000]:    0
<<<<<<<<<<Rece info test<<<<<<<<<<
index:4971
depth[5000] :  578 depth[10000]:  619 depth[15000]:  590 depth[20000]:  636
depth[25000]:  684 depth[30000]:  646 depth[35000]:  704 depth[40000]:    0
```