XingYing_SDK is a communication plug-in written in C++ language, in the form of a dynamic library DLL plug-in. By including the SDK into the C++ project and calling the SDK's public interface, Makrer point,rigid body information and force plate including other simulation channel information can be obtained. Use reference SampleClient.

1. Establish connection and communication:

(1) Create a NokovSDKClient object and enter the IP address of the XingYing target. If running on the same computer as NokovSDKClient, enter "10.1.1.198". The C++ code looks like this:

```
        // The first parameter is the SDK broadcast IP of XINGYING software,see:Setting
=>NetWork =>LocalAddress
        // ServerIP:10.1.1.198

        NokovSDKClient* theClient = new NokovSDKClient();
        theClient->Initialize("10.1.1.198");
```

2. Received data:

(1) Set the callback function to receive the transferred data. The C++ code is as follows:

```
void DataHandler(sFrameOfMocapData* data, void* pUserData);
theClient->SetDataCallback(DataHandler, theClient);
```

3. Data parsing (reading Marker data and rigid-body data) :

(1) Get all MarkerSet number:

```
int markerSetNumber = data.nMarkerSets;
```

(2) Obtain the position information of the j-th marker of the i-th MarkerSet:

```
float point = data.MocapData[i].Markers[j][0]; // 0-x 1-y 2-z
```

(3) Get all rigid bodies number:

```
int number = data.nRigidBodies;
```

(4) Get the attitude information of the i-th rigidbody(position,rotation) :

```
float point = data.RigidBodies[i].x; // x y z qx qz qw
```

(5) Get all NamedMarker number:

```
int number = data.nLabeledMarkers;
```

(6) Obtain the location information of the i-th NamedMarker:

```
float point = data.LabeledMarkers[i].x; // x y z
```

(7) Get all UnNamedMarker number:

```
int number = data.nOtherMarkers;
```

(8) Obtain the location information of the i-th UnNamedMarker:

```
float point = data.OtherMarkers[i][0]; //0-x 1-y 2-z
```

(9) Get the number of analog channels:

```
int number = data.nAnalogdatas;
```

(10) Get the data of the i-th analog channel:

```
float point = data.Analogdata[i];
```

(11) Get the absolute timestamp of the frame:

```
long long timestamp = data.iTimeStamp //Number of milliseconds since 1970-01-01
```

4. Use of force plate:

(1) Initialize the force plate:

```
// Wait for the force plate to be initialized,0ms indicates infinite wait.
retCode = theClient->WaitForForcePlateInit(0);
```

(2) Set the callback of force plate:

```
void ForcePlateHandler(sForcePlates* pForcePlate, void* pUserData);
theClient->SetForcePlateCallback(ForcePlateHandler, theClient);
```

(3) Analyzing the data of the force plate, the force (N) on the X-axis of the i-th force table is as follows:

```
pForcePlate->ForcePlates[i].Fxyz[0];
```

5. Close the connection:

```
theClient->Uninitialize();
```