VelodyneControl.cpp

+ SendData()

VelodyneWrapper.cpp

- VelodyneData
- VelodyneBlock

+ SendData()
+ SetAzimuth(azimuth)
+ SetTimeStamp(timeStamp)
+ SetChannel(distance, ref)
+ CloseBlock()

VelodynePluginAPI.cpp -> libvelodyne.so

+ Velodyne16CreateObject()
+ Velodyne32CreateObject()
+ VelodyneDeleteObject(VelodyneWrapper*)
+ VelodyneSetAzimuth(VelodyneWrapper*, azimuth)
+ VelodyneSetTimeStamp(VelodyneWrapper*, timeStamp)
+ VelodyneSetChannel(VelodyneWrapper*, distance, ref)
+ VelodyneCloseBlock(VelodyneWrapper*)
+ VelodyneSenddata(VelodyneWrapper*)

Velodyne16.cs

- VelodyneWrapper

VelodyneWrapper.cs

- VelodyneWrapper*

DLLImport(libvlp.so)
Create, delete, set...

+ SetAzimith(azimuth)
+ SetTimeStamp(timeStamp)
+ SetChannel(distance, ref)
+ CloseBlock()
+ SendData()

Explanation:
1. VelodyneControl is the ICD that gets data and sends it with UDP port
2. VelodyneWrapper is designed to allow "c" prototypes of the functions. It is compiled with VelodynePluginAPI that externs the functions and it creates a dynamic library (libvlp.so)
3. VelodyneWrapper.cs designed to connect between the cs code and cpp code (dynamic library), so Velodyne.cs could simply use it.