

我们希望在 B 功能包 引用 A 功能包的头文件和实现的函数

B 是 Pure\_pursuit

A 是 Kinematic\_Model

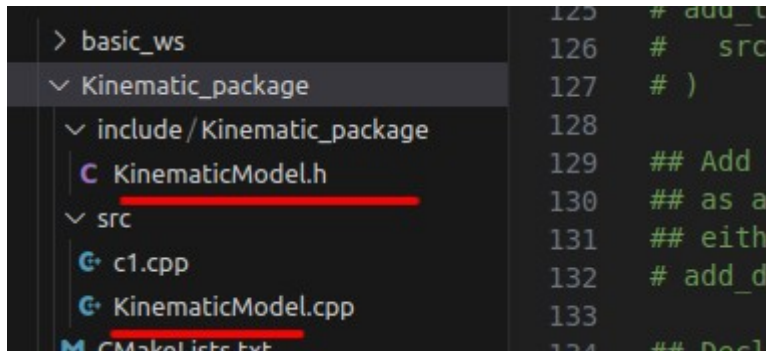
<https://blog.csdn.net/lanhuazhiyue/article/details/132570957>

参考连接

A 功能包的修改 Kinematic\_Model

1、在 include/Kinematic\_Model/创建.h 的头文件

2、在 src 下实现.cpp 文件



3、配置 cmake 文件

让其余功能包能够找到 这个功能包的头文件

```
106  ## DEPENDS: system dependencies of this project
107  catkin_package(
108    | INCLUDE_DIRS include
109    # LIBRARIES Kinematic_package
110    # CATKIN_DEPENDS roscpp rospy std_msgs
111    # DEPENDS system_lib
112  )
113
```

要让**功能包名**与 实现的 运动学模型的.h 头文件和.cpp 实现文件 建立关系

```
## Declare a C++ library
add_library(Kinematic_package
  include/Kinematic_package/KinematicModel.h
  src/KinematicModel.cpp
)
## Add cmake target dependencies of the library
```

将我们的**功能包名**与编译空间关联

```
## Specify libraries to link a library or executable
target_link_libraries(Kinematic_package
  ${catkin_LIBRARIES}
)
```

## B 功能包 Pure\_pursuit

### 1、找功能包

```
## is used, also find other catkin packages
find_package(catkin REQUIRED COMPONENTS
  roscpp
  rospy
  std_msgs
  Kinematic_package
)
```

### 2、编译 我们在编译这个文件的时候，我们依赖这个功能包

```
## CATKIN_DEPENDS: catkin_packages dependent projects also need
## DEPENDS: system dependencies of this project that dependent projects also need
catkin_package(
  # INCLUDE_DIRS include
  # LIBRARIES Pure_pursuit
  CATKIN_DEPENDS roscpp rospy std_msgs Kinematic_package
  # DEPENDS system_lib
)

#####
```

### 3、

add\_executable 我们的 mytry 要调用

add\_dependencies mytry 编译前 要把相关的依赖编译了

target\_link\_libraries mytry 会用哪些链接库 会用我们自己的 catkin\_libraries 的 以及我们使用的 A 功能包的

```
135 ## with catkin_make all packages are built within a single cmake context
136 ## The recommended prefix ensures that target names across packages don't collide
137 add_executable(mytry src/mytry.cpp)
138
139 ## Rename C++ executable without prefix
140 ## The above recommended prefix causes long target names, the following renames the
141 ## target back to the shorter version for ease of user use
142 ## e.g. "roslaunch someones_pkg node" instead of "roslaunch someones_pkg someones_pkg_node"
143 # set_target_properties(${PROJECT_NAME}_node PROPERTIES OUTPUT_NAME node PREFIX "")
144
145 ## Add cmake target dependencies of the executable
146 ## same as for the library above
147 add_dependencies(mytry ${${PROJECT_NAME}_EXPORTED_TARGETS} ${catkin_EXPORTED_TARGETS})
148
149 ## Specify libraries to link a library or executable target against
150 target_link_libraries(mytry
151   Kinematic_package
152   ${catkin_LIBRARIES}
153 )
154
```

例子 在 mytry.cpp 文件

我们还给 Kinematic\_package 在 add\_link\_library()

```
mytry.cpp | tasks.json | pid_track.cpp | CMakeLists.txt .../Pid_pathtracking | C n
> Pure_pursuit > src > mytry.cpp > main(int, char * [])
1  #include "ros/ros.h"
2  #include "Kinematic_package/KinematicModel.h"
3  #include "Kinematic_package/mul_remap.h"
4  int main(int argc, char* argv[])
5  {
6
7
8      KinematicModel ugv(0, -1, 0.5, 2, 2, 0.1);
9      say_hello();
10
11     return 0;
12 }
```

Kinematic\_package 不仅仅与 Kinematic Model 还和一个 mul\_remap 的.h 和.cpp 文件

```
53
54  ## Specify libraries to link a library or executable
55  target_link_libraries(Kinematic_package
56  mul_remap
57  | ${catkin_LIBRARIES}
58  )
59  target_link_libraries(mul_remap
```

最后的运行的结果

```
say hello
● qinghuan@qinghuan-System-Product-Name:~/env_cv/demo06_ws$ source ./devel/setup.bash
● qinghuan@qinghuan-System-Product-Name:~/env_cv/demo06_ws$ rosrun Pure_pursuit mytry
0, -1, 0.5, 2, 2, 0.1,
say hello
○ qinghuan@qinghuan-System-Product-Name:~/env_cv/demo06_ws$ □
```