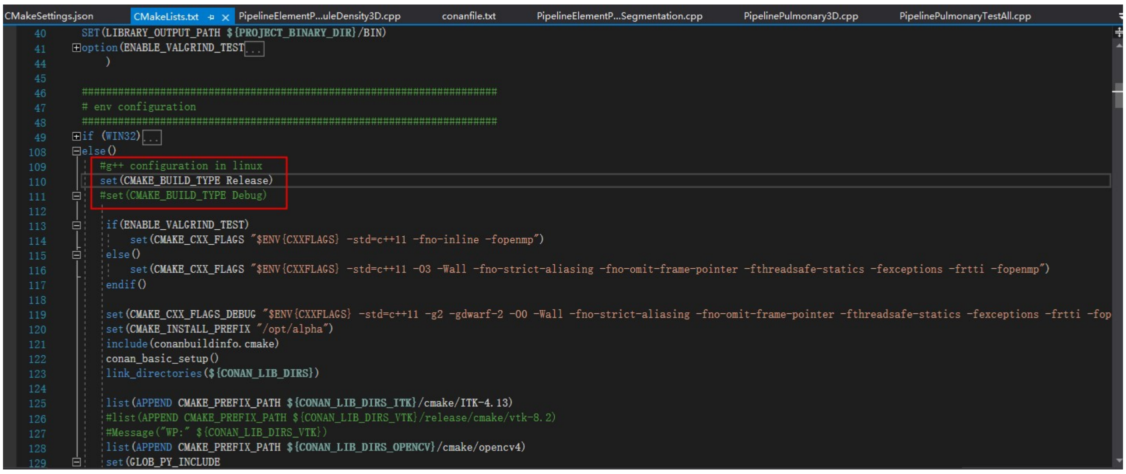
Debug Alpha

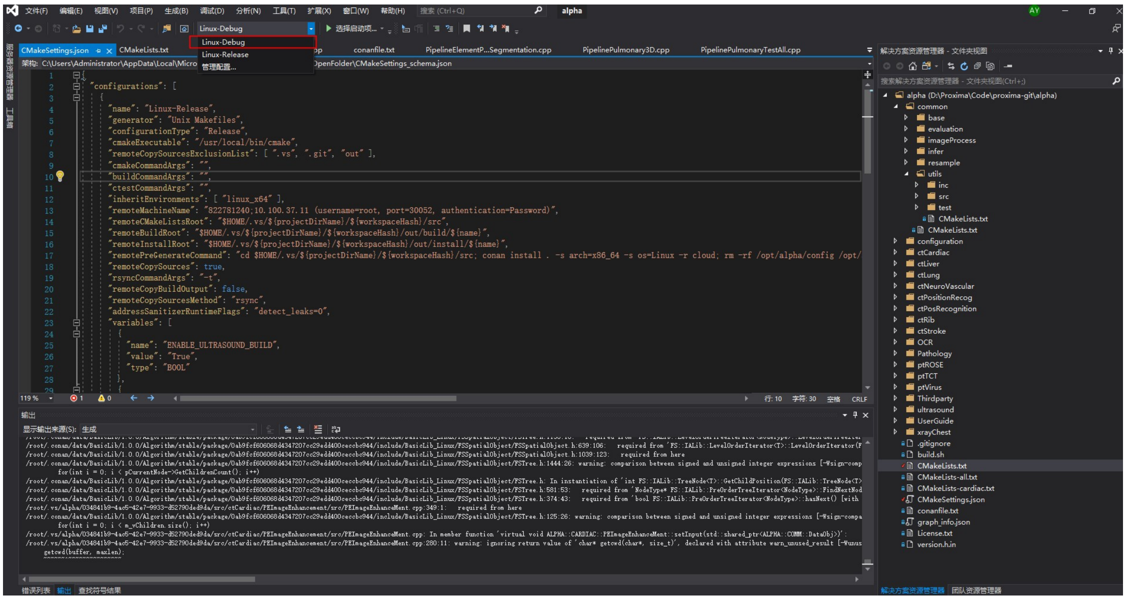
- 修改CMakeLists.txt编译模式为Debug

```
#g++ configuration in linux
#set(CMAKE_BUILD_TYPE Release)
set(CMAKE_BUILD_TYPE Debug)
```


CMakeSettings.json
CMakeLists.txt
PipelineElementP...uleDensity3D.cpp
conanfile.txt
PipelineElementP...Segmentation.cpp
PipelinePulmonary3D.cpp
PipelinePulmonaryTestAll.cpp

```
SET(LIBRARY_OUTPUT_PATH ${PROJECT_BINARY_DIR}/BIN)
option(ENABLE_VALGRIND_TEST...
    )

##########################################################################
# env configuration
##########################################################################
if (WIN32)...
else()
    #g++ configuration in linux
    set(CMAKE_BUILD_TYPE Release)
    #set(CMAKE_BUILD_TYPE Debug)

    if(ENABLE_VALGRIND_TEST)
        set(CMAKE_CXX_FLAGS "$ENV{CXXFLAGS} -std=c++11 -fno-inline -fopenmp")
    else()
        set(CMAKE_CXX_FLAGS "$ENV{CXXFLAGS} -std=c++11 -O3 -Wall -fno-strict-aliasing -fno-omit-frame-pointer -fthreadsafe-statics -fexceptions -frtti -fopenmp")
    endif()

    set(CMAKE_CXX_FLAGS_DEBUG "$ENV{CXXFLAGS} -std=c++11 -g2 -gdwarf-2 -O0 -Wall -fno-strict-aliasing -fno-omit-frame-pointer -fthreadsafe-statics -fexceptions -frtti -fop
    set(CMAKE_INSTALL_PREFIX "/opt/alpha")
    include(conanbuildinfo.cmake)
    conan_basic_setup()
    link_directories(${CONAN_LIB_DIRS})

    list(APPEND CMAKE_PREFIX_PATH ${CONAN_LIB_DIRS_ITK}/cmake/ITK-4.13)
    #list(APPEND CMAKE_PREFIX_PATH ${CONAN_LIB_DIRS_VTK}/release/cmake/vtk-8.2)
    #Message("WP:" ${CONAN_LIB_DIRS_VTK})
    list(APPEND CMAKE_PREFIX_PATH ${CONAN_LIB_DIRS_OPENCV}/cmake/opencv4)
    set(GLOB_PY_INCLUDE
```

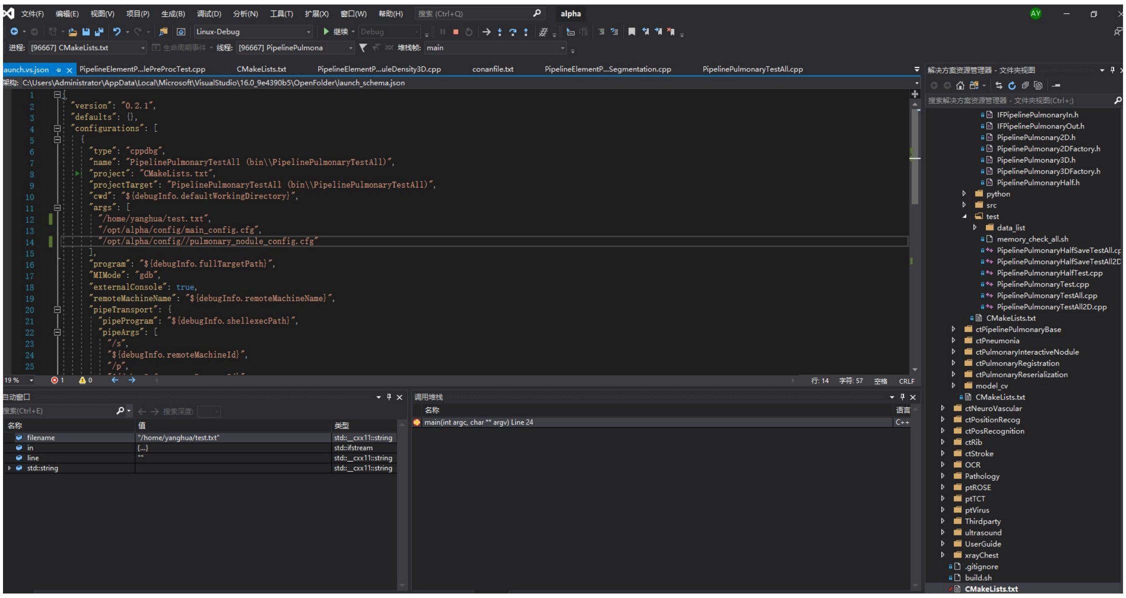
- 配置选项选择 Linux-Debug，编译



```
{
  "configurations": [
    {
      "name": "Linux-Release",
      "generator": "Unix Makefiles",
      "configurationType": "Release",
      "cmakeExecutable": "/usr/local/bin/cmake",
      "remoteCopySourcesExclusionList": [ ".vs", ".git", "out" ],
      "cmakeCommandArgs": "",
      "buildCommandArgs": "",
      "ctestCommandArgs": "",
      "inheritEnvironments": [ "linux_x64" ],
      "remoteMachineName": "822781240;10.100.37.11 (username=root, port=30052, authentication=Password)",
      "remoteCMakeListsRoot": "$HOME/.vs/${projectDirName}/${workspaceHash}/src",
      "remoteBuildRoot": "$HOME/.vs/${projectDirName}/${workspaceHash}/out/build/${name}",
      "remoteInstallRoot": "$HOME/.vs/${projectDirName}/${workspaceHash}/out/install/${name}",
      "remotePreGenerateCommand": "cd $HOME/.vs/${projectDirName}/${workspaceHash}/src; conan install . -s arch=x86_64 -s os=Linux -r cloud; rm -rf /opt/alpha/config /opt/
      "remoteCopySources": true,
      "rsyncCommandArgs": "-t",
      "remoteCopyBuildOutput": false,
      "remoteCopySourcesMethod": "rsync",
      "addressSanitizerRuntimeFlags": "detect_leaks=0",
      "variables": [
        {
          "name": "ENABLE_ULTRASOUND_BUILD",
          "value": "True",
          "type": "BOOL"
        },
        {
```

```
/root/.conan/data/BasicLib/1.0.0/Algorithm/stable/package/0ab9fcf606068d4347207cc29edd400ceccbc944/include/BasicLib_Linux/FSSpatialObject/FSSpatialObject.h:639:106:   required from 'FS::IALib::LevelOrderIterator<T>::LevelOrderIterator(F
/root/.conan/data/BasicLib/1.0.0/Algorithm/stable/package/0ab9fcf606068d4347207cc29edd400ceccbc944/include/BasicLib_Linux/FSSpatialObject/FSSpatialObject.h:1039:123:   required from here
/root/.conan/data/BasicLib/1.0.0/Algorithm/stable/package/0ab9fcf606068d4347207cc29edd400ceccbc944/include/BasicLib_Linux/FSSpatialObject/FSTree.h:1444:26: warning: comparison between signed and unsigned integer expressions [-Wsign-comp
         for(int i = 0; i < pCurrentNode->GetChildrenCount(); i++)
/root/.conan/data/BasicLib/1.0.0/Algorithm/stable/package/0ab9fcf606068d4347207cc29edd400ceccbc944/include/BasicLib_Linux/FSSpatialObject/FSTree.h: In instantiation of 'int FS::IALib::TreeNode<T>::GetChildPosition(FS::IALib::TreeNode<T>
/root/.conan/data/BasicLib/1.0.0/Algorithm/stable/package/0ab9fcf606068d4347207cc29edd400ceccbc944/include/BasicLib_Linux/FSSpatialObject/FSTree.h:581:53:   required from 'NodeType* FS::IALib::PreOrderTreeIterator<NodeType>::FindNextNod
/root/.conan/data/BasicLib/1.0.0/Algorithm/stable/package/0ab9fcf606068d4347207cc29edd400ceccbc944/include/BasicLib_Linux/FSSpatialObject/FSTree.h:374:43:   required from 'bool FS::IALib::PreOrderTreeIterator<NodeType>::hasNext() [with
/root/.vs/alpha/034841b9-4ac5-42e7-9933-d52790ded9da/src/ctCardiac/PEImageEnhancement/src/PEImageEnhanceMent.cpp:349:1:   required from here
/root/.conan/data/BasicLib/1.0.0/Algorithm/stable/package/0ab9fcf606068d4347207cc29edd400ceccbc944/include/BasicLib_Linux/FSSpatialObject/FSTree.h:125:26: warning: comparison between signed and unsigned integer expressions [-Wsign-compa
         for(int i = 0; i < m_vChildren.size(); i++)
/root/.vs/alpha/034841b9-4ac5-42e7-9933-d52790ded9da/src/ctCardiac/PEImageEnhancement/src/PEImageEnhanceMent.cpp: In member function 'virtual void ALPHA::CARDIAC::PEImageEnhanceMent::setInput(std::shared_ptr<ALPHA::COMM::DataObj>)':
/root/.vs/alpha/034841b9-4ac5-42e7-9933-d52790ded9da/src/ctCardiac/PEImageEnhancement/src/PEImageEnhanceMent.cpp:280:11: warning: ignoring return value of 'char* getcwd(char*, size_t)', declared with attribute warn_unused_result [-Wunus
      getcwd(buffer, maxlen);
      ~~~~~~^~~~~~~~~~~~~~~~
```

- 选择对应启动项，并完成参数配置



```
{
  "version": "0.2.1",
  "defaults": {},
  "configurations": [
    {
      "type": "cppdbg",
      "name": "PipelinePulmonaryTestAll (bin\\PipelinePulmonaryTestAll)",
      "project": "CMakeLists.txt",
      "projectTarget": "PipelinePulmonaryTestAll (bin\\PipelinePulmonaryTestAll)",
      "cwd": "${debugInfo.defaultWorkingDirectory}",
      "args": [
        "/home/yanghua/test.txt",
        "/opt/alpha/config/main_config.cfg",
        "/opt/alpha/config//pulmonary_nodule_config.cfg"
      ],
      "program": "${debugInfo.fullTargetPath}",
      "MIMode": "gdb",
      "externalConsole": true,
      "remoteMachineName": "${debugInfo.remoteMachineName}",
      "pipeTransport": {
        "pipeProgram": "${debugInfo.shellexecPath}",
        "pipeArgs": [
          "/s",
          "${debugInfo.remoteMachineId}",
          "/p",
```

自动窗口

名称	值	类型
filename	"/home/yanghua/test.txt"	std::__cxx11::string
in	{...}	std::ifstream
line	""	std::__cxx11::string
std::string		std::__cxx11::string

调用堆栈

名称	语言
main(int argc, char ** argv) Line 24	C++

文件(F)
编辑(E)
视图(V)
项目(P)
生成(B)
调试(D)
分析(N)
工具(T)
扩展(X)
窗口(W)
帮助(H)
搜索 (Ctrl+Q)
alpha
Linux-Debug
继续
Debug
进程: [96667] CMakeLists.txt
生命周期事件
线程: [96667] PipelinePulmona
堆栈帧: main
launch.vs.json
PipelineElementP...lePreProcTest.cpp
CMakeLists.txt
PipelineElementP...uleDensity3D.cpp
conanfile.txt
PipelineElementP...Segmentation.cpp
PipelinePulmonaryTestAll.cpp
架构: C:\Users\Administrator\AppData\Local\Microsoft\VisualStudio\16.0_9e4390b5\OpenFolder\launch_schema.json
19 %
行: 14
字符: 57
空格
CRLF
搜索(Ctrl+E)
搜索深度:
解决方案资源管理器 - 文件夹视图
搜索解决方案资源管理器 - 文件夹视图(Ctrl+;)
IFPipelinePulmonaryIn.h
IFPipelinePulmonaryOut.h
PipelinePulmonary2D.h
PipelinePulmonary2DFactory.h
PipelinePulmonary3D.h
PipelinePulmonary3DFactory.h
PipelinePulmonaryHalf.h
python
src
test
data_list
memory_check_all.sh
PipelinePulmonaryHalfSaveTestAll.cp
PipelinePulmonaryHalfSaveTestAll2D
PipelinePulmonaryHalfTest.cpp
PipelinePulmonaryTest.cpp
PipelinePulmonaryTestAll.cpp
PipelinePulmonaryTestAll2D.cpp
CMakeLists.txt
ctPipelinePulmonaryBase
ctPneumonia
ctPulmonaryInteractiveNodule
ctPulmonaryRegistration
ctPulmonaryReserialization
model_cv
CMakeLists.txt
ctNeuroVascular
ctPositionRecog
ctPosRecognition
ctRib
ctStroke
OCR
Pathology
ptROSE
ptTCT
ptVirus
Thirdparty
ultrasound
UserGuide
xrayChest
.gitignore
build.sh
CMakeLists.txt

Debug Alpha-acl

- 运行环境

```
version: '2.4'
services:
  yanghua_alpha_api:
    image: harbor.do.proxima-ai.com/alpha/alpha-deploy:1.3
    restart: always
    env_file:
      - alpha.env
    environment:
      - LANG=C.UTF-8
      - ENV=RUN
    ports:
      - "30052:22"
      - "30053:8088"
    security_opt:
      - seccomp:unconfined
    container_name: yanghua-alpha-prod
    volumes:
      - /data:/data
      - /fileser:/fileser
      - /ssd:/ssd
      - /ssd2:/ssd2
      - /home:/home
      - /etc/localtime:/etc/localtime:ro
      - ../third/lib:/opt/third/lib
      - ../third/include:/opt/third/include
      - ../alpha:/opt/alpha
      - ../acl:/opt/acl
      - ../install:/opt/alpha/install
      - ./:/opt/alpha/run
      - ../test_case:/opt/alpha/test_case
      - ../bad_case:/opt/alpha/bad_case
      -
/opt/ringv2/deploy/static/cornerstone:/data/ring/deploy/static/cornerstone
```

```
    network_mode:
      bridge
    shm_size: 48G
```

- 依赖项
  - Alpha
    - 【已默认实现】使用CPack打包（抽取头文件和库文件）

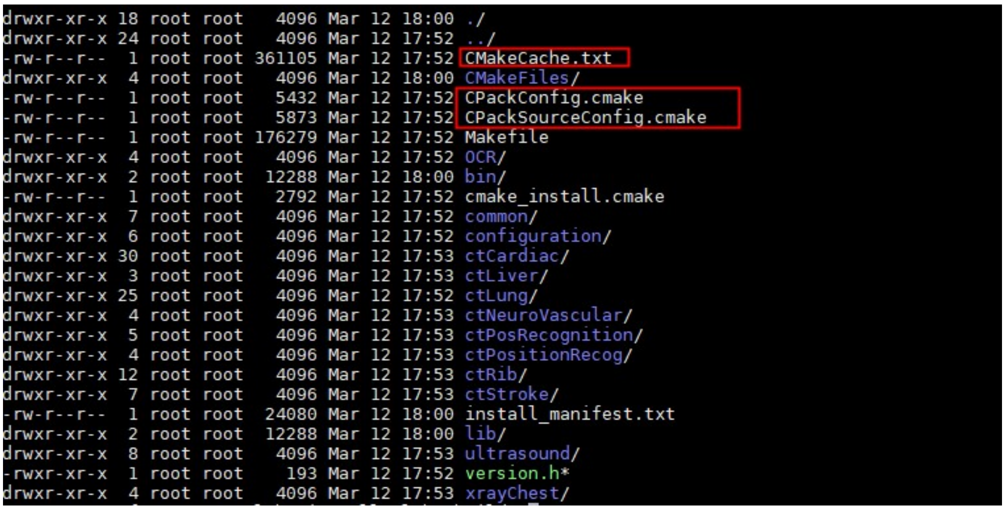
      1. 在根目录CMakeLists.txt的底部添加几行代码

         ```
         include (InstallRequiredSystemLibraries)
         set (CPACK_RESOURCE_FILE_LICENSE
         "${CMAKE_CURRENT_SOURCE_DIR}/License.txt")
         set (CPACK_PACKAGE_VERSION_MAJOR "${VERSION_INFO_MAJOR}")
         set (CPACK_PACKAGE_VERSION_MINOR "${VERSION_INFO_MAJOR}")
         include (CPack)
         ```

      2. 在各自项目中添加要打包的文件，以common项目中的CommonUtils模块为例，在 "Alpha\common\utils\CMakelist.txt" 末尾添加如下代码

         ```
         install (TARGETS CommonUtils DESTINATION bin)
         install (FILES ${CMAKE_CURRENT_SOURCE_DIR}/inc/CommonUtils.h
         DESTINATION include/common)
         ```

    - 在docker镜像内任意目录拉取Alpha仓库，并执行 sh ./build.sh，在 build目录生成如下产物



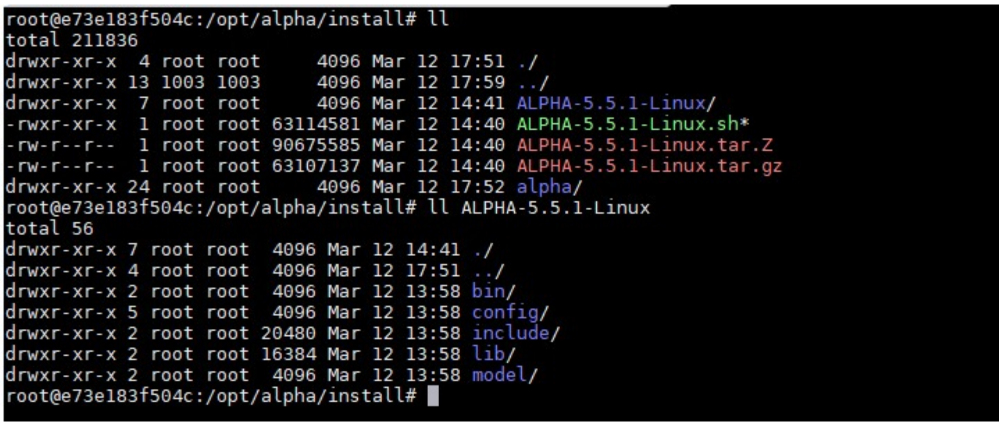
```
drwxr-xr-x 18 root root   4096 Mar 12 18:00 ./
drwxr-xr-x 24 root root   4096 Mar 12 17:52 ../
-rw-r--r--  1 root root 361105 Mar 12 17:52 CMakeCache.txt
drwxr-xr-x  4 root root   4096 Mar 12 18:00 CMakeFiles/
-rw-r--r--  1 root root   5432 Mar 12 17:52 CPackConfig.cmake
-rw-r--r--  1 root root   5873 Mar 12 17:52 CPackSourceConfig.cmake
-rw-r--r--  1 root root 176279 Mar 12 17:52 Makefile
drwxr-xr-x  4 root root   4096 Mar 12 17:52 OCR/
drwxr-xr-x  2 root root  12288 Mar 12 18:00 bin/
-rw-r--r--  1 root root   2792 Mar 12 17:52 cmake_install.cmake
drwxr-xr-x  7 root root   4096 Mar 12 17:52 common/
drwxr-xr-x  6 root root   4096 Mar 12 17:52 configuration/
drwxr-xr-x 30 root root   4096 Mar 12 17:53 ctCardiac/
drwxr-xr-x  3 root root   4096 Mar 12 17:53 ctLiver/
drwxr-xr-x 25 root root   4096 Mar 12 17:52 ctLung/
drwxr-xr-x  4 root root   4096 Mar 12 17:53 ctNeuroVascular/
drwxr-xr-x  5 root root   4096 Mar 12 17:53 ctPosRecognition/
drwxr-xr-x  4 root root   4096 Mar 12 17:53 ctPositionRecog/
drwxr-xr-x 12 root root   4096 Mar 12 17:53 ctRib/
drwxr-xr-x  7 root root   4096 Mar 12 17:53 ctStroke/
-rw-r--r--  1 root root  24080 Mar 12 18:00 install_manifest.txt
drwxr-xr-x  2 root root  12288 Mar 12 18:00 lib/
drwxr-xr-x  8 root root   4096 Mar 12 17:53 ultrasound/
-rwxr-xr-x  1 root root    193 Mar 12 17:52 version.h*
drwxr-xr-x  4 root root   4096 Mar 12 17:53 xrayChest/
```

    - 在build目录，执行如下命令生成打包文件：ALPHA-xxx-Linux.sh，ALPHA-xxx-Linux..tar.gz，ALPHA-xxx-Linux.tar.z

      ```
      cpack -C CPackConfig.cmake
      ```

    - 执行如下命令，解压生成ALPHA-xxx-Linux目录，提取头文件和库文件

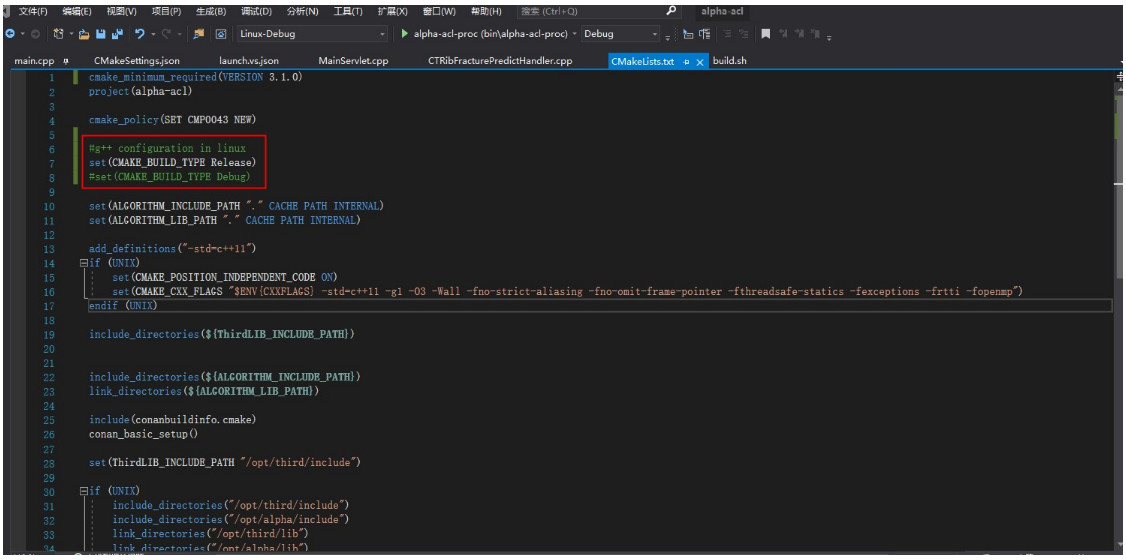
      ```
      sh ./ALPHA-5.5.1-Linux.sh
      ```



```
root@e73e183f504c:/opt/alpha/install# ll
total 211836
drwxr-xr-x  4 root root     4096 Mar 12 17:51 ./
drwxr-xr-x 13 1003 1003     4096 Mar 12 17:59 ../
drwxr-xr-x  7 root root     4096 Mar 12 14:41 ALPHA-5.5.1-Linux/
-rwxr-xr-x  1 root root 63114581 Mar 12 14:40 ALPHA-5.5.1-Linux.sh*
-rw-r--r--  1 root root 90675585 Mar 12 14:40 ALPHA-5.5.1-Linux.tar.Z
-rw-r--r--  1 root root 63107137 Mar 12 14:40 ALPHA-5.5.1-Linux.tar.gz
drwxr-xr-x 24 root root     4096 Mar 12 17:52 alpha/
root@e73e183f504c:/opt/alpha/install# ll ALPHA-5.5.1-Linux
total 56
drwxr-xr-x 7 root root  4096 Mar 12 14:41 ./
drwxr-xr-x 4 root root  4096 Mar 12 17:51 ../
drwxr-xr-x 2 root root  4096 Mar 12 13:58 bin/
drwxr-xr-x 5 root root  4096 Mar 12 13:58 config/
drwxr-xr-x 2 root root 20480 Mar 12 13:58 include/
drwxr-xr-x 2 root root 16384 Mar 12 13:58 lib/
drwxr-xr-x 2 root root  4096 Mar 12 13:58 model/
root@e73e183f504c:/opt/alpha/install# 
```

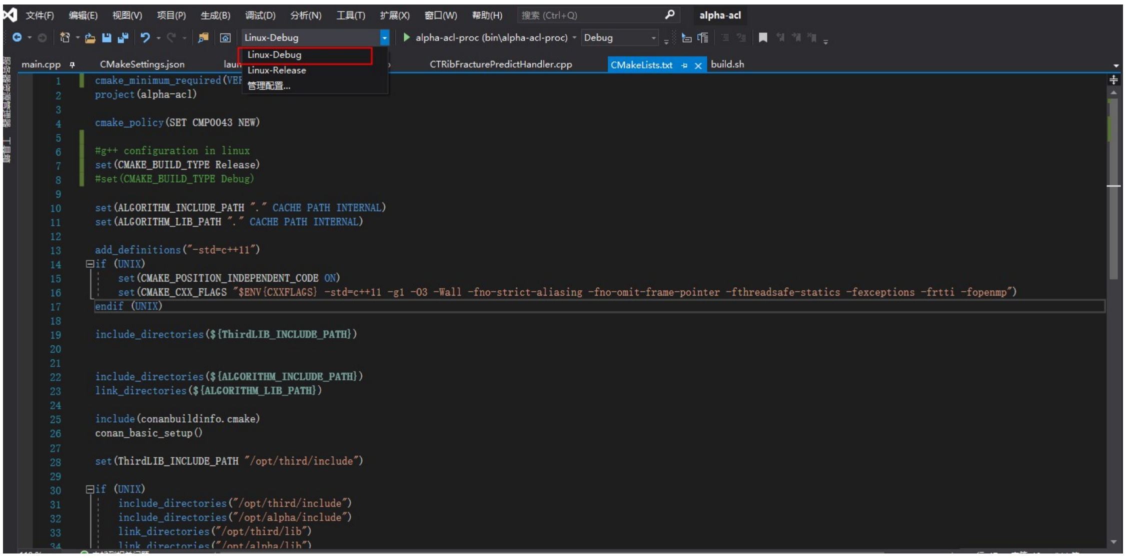
        - 将bin，config，include，lib，model文件夹下内容拷贝到/opt/alpha对应目录
    - Hemo
        - 在镜像内任意目录拉取Hemo仓库，并执行 sh ./build.sh（CMakeLists.txt会将产物拷贝至/opt/alpha对应目录）
- 修改CMakeLists.txt编译模式为Debug

```
#g++ configuration in linux
#set(CMAKE_BUILD_TYPE Release)
set(CMAKE_BUILD_TYPE Debug)
```



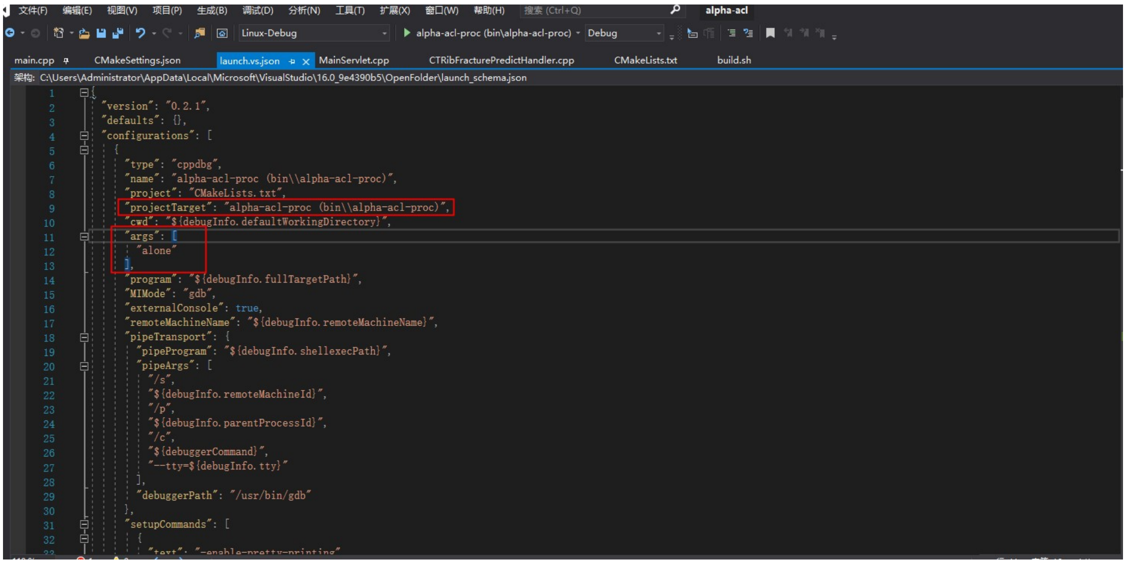
```
cmake_minimum_required(VERSION 3.1.0)
project(alpha-acl)

cmake_policy(SET CMP0043 NEW)

#g++ configuration in linux
set(CMAKE_BUILD_TYPE Release)
#set(CMAKE_BUILD_TYPE Debug)

set(ALGORITHM_INCLUDE_PATH "." CACHE PATH INTERNAL)
set(ALGORITHM_LIB_PATH "." CACHE PATH INTERNAL)

add_definitions("-std=c++11")
if (UNIX)
    set(CMAKE_POSITION_INDEPENDENT_CODE ON)
    set(CMAKE_CXX_FLAGS "$ENV{CXXFLAGS} -std=c++11 -g1 -O3 -Wall -fno-strict-aliasing -fno-omit-frame-pointer -fthreadsafe-statics -fexceptions -frtti -fopenmp")
endif (UNIX)

include_directories(${ThirdLIB_INCLUDE_PATH})


include_directories(${ALGORITHM_INCLUDE_PATH})
link_directories(${ALGORITHM_LIB_PATH})

include(conanbuildinfo.cmake)
conan_basic_setup()

set(ThirdLIB_INCLUDE_PATH "/opt/third/include")

if (UNIX)
    include_directories("/opt/third/include")
    include_directories("/opt/alpha/include")
    link_directories("/opt/third/lib")
    link_directories("/opt/alpha/lib")
```

- 配置选项选择 Linux-Debug，编译


文件(F) 编辑(E) 视图(V) 项目(P) 生成(B) 调试(D) 分析(N) 工具(T) 扩展(X) 窗口(W) 帮助(H)
搜索 (Ctrl+Q)
alpha-acl
Linux-Debug
Linux-Debug
Linux-Release
管理配置...
alpha-acl-proc (bin\alpha-acl-proc)
Debug
main.cpp
CMakeSettings.json
CTRibFracturePredictHandler.cpp
CMakeLists.txt
build.sh

```
cmake_minimum_required(VEF
project(alpha-acl)

cmake_policy(SET CMP0043 NEW)

#g++ configuration in linux
set(CMAKE_BUILD_TYPE Release)
#set(CMAKE_BUILD_TYPE Debug)

set(ALGORITHM_INCLUDE_PATH "." CACHE PATH INTERNAL)
set(ALGORITHM_LIB_PATH "." CACHE PATH INTERNAL)

add_definitions("-std=c++11")
if (UNIX)
    set(CMAKE_POSITION_INDEPENDENT_CODE ON)
    set(CMAKE_CXX_FLAGS "$ENV{CXXFLAGS} -std=c++11 -g1 -O3 -Wall -fno-strict-aliasing -fno-omit-frame-pointer -fthreadsafe-statics -fexceptions -frtti -fopenmp")
endif (UNIX)

include_directories(${ThirdLIB_INCLUDE_PATH})


include_directories(${ALGORITHM_INCLUDE_PATH})
link_directories(${ALGORITHM_LIB_PATH})

include(conanbuildinfo.cmake)
conan_basic_setup()

set(ThirdLIB_INCLUDE_PATH "/opt/third/include")

if (UNIX)
    include_directories("/opt/third/include")
    include_directories("/opt/alpha/include")
    link_directories("/opt/third/lib")
    link_directories("/opt/alpha/lib")
```

- 选择对应启动项，并完成参数配置


文件(F) 编辑(E) 视图(V) 项目(P) 生成(B) 调试(D) 分析(N) 工具(T) 扩展(X) 窗口(W) 帮助(H)
搜索 (Ctrl+Q)
alpha-acl
Linux-Debug
alpha-acl-proc (bin\alpha-acl-proc)
Debug
main.cpp
CMakeSettings.json
launch.vs.json
MainServlet.cpp
CTRibFracturePredictHandler.cpp
CMakeLists.txt
build.sh

架构: C:\Users\Administrator\AppData\Local\Microsoft\VisualStudio\16.0_9e4390b5\OpenFolder\launch_schema.json

```
{
  "version": "0.2.1",
  "defaults": {},
  "configurations": [
    {
      "type": "cppdbg",
      "name": "alpha-acl-proc (bin\\alpha-acl-proc)",
      "project": "CMakeLists.txt",
      "projectTarget": "alpha-acl-proc (bin\\alpha-acl-proc)",
      "cwd": "${debugInfo.defaultWorkingDirectory}",
      "args": [
        "alone"
      ],
      "program": "${debugInfo.fullTargetPath}",
      "MIMode": "gdb",
      "externalConsole": true,
      "remoteMachineName": "${debugInfo.remoteMachineName}",
      "pipeTransport": {
        "pipeProgram": "${debugInfo.shellexecPath}",
        "pipeArgs": [
          "/s",
          "${debugInfo.remoteMachineId}",
          "/p",
          "${debugInfo.parentProcessId}",
          "/c",
          "${debuggerCommand}",
          "--tty=${debugInfo.tty}"
        ],
        "debuggerPath": "/usr/bin/gdb"
      },
      "setupCommands": [
        {
          "text": "-enable-pretty-printing"
```