**1.安装Ubantu系统**

**2.安装开发环境**

**2.1 安装开发必须的软件包**

sudo apt-get update

sudo apt-get install build-essential flex bison lzop libncurses5-dev libssl-dev gcc g++ make cmake cmake-gui subversion vim libgtk2.0-dev pkg-config

# 如果使用的是 64 位的 Ubuntu，还需要安装：

sudo apt-get install libc6:i386

#安装 ARM 交叉编译工具链和编译内核相关软件包：

sudo apt-get install gcc-arm-linux-gnueabihf lzop libncurses5-dev libssl1.0.0 libssl-dev

#安装中文语言包

sudo apt-get install language-pack-zh-hans

中英文切换：

#sudo vim /etc/default/locale

中文设置为：（英文切换为中文不确定是否需要安装语言包）  
LANG="zh\_CN.UTF-8"  
LANGUAGE="zh\_CN:zh"

英文设置为：  
LANG="en\_US.UTF-8"  
LANGUAGE="en\_US:en"

重启系统生效

**2.2 通过SVN下载DF\_VISION\_SDK：**

cd /

svn --username=licui checkout https://192.168.10.188:8443/svn/DragonflyVisionSVN/Products\&Projects/DragonflyVisionAssistant/Src/Software/DF\_Vision\_SDK /

注：红色部分为自己的svn账号，输入以上指令之后会要求输入密码；

注：以上svn地址与windows有一点小差异，因为有&符号，所以要添加反斜杠才能 正常使用。

**2.3** **安装GTK**（必须先安装GTK再编译opencv才可以正常调用cvShowImage进行图片显示）

sudo apt-get install gnome-core-devel libgtk2.0-dev libgtk2.0\* libglib2.0-doc libgtk2.0-doc devhelp

**2.4 安装cmake,编译Opencv**

#安装编译器  
sudo apt-get install gcc g++ lzop libncurses5-dev libssl1.0.0 libssl-dev

#安装cmake 和cmake-gui  
sudo apt-get install cmake cmake-gui

#解压使用cmake配置opencv  
mkdir /work/opencv  
mv opencv-2.4.8.tar.gz /work/opencv  
tar xzvf opencv-2.4.8.tar.gz  
mkdir opencv\_arm #以下将opencv\_arm作为编译生成路径，可自行修改  
cmake-gui

注：实际时应根据自己需求设置，本文中是使用opencv\_arm；

选择编译器,对于PC安装直接使用以下默认的编译器即可：

选择Configure正常配置时界面如下：

勾选BUILD\_JPEG、BUILD\_PNG、BUILD\_TIFF、BUILD\_ZLIB；  
修改CMAKE\_INSTALL\_PREFIX为编译安装的路径，如/work/opencv/opencv\_arm/install  
该值为make install时的路径；  
  
查看是否已经默认勾选WITH\_GTK，如未勾选则可能环境搭建有问题；  
其他保留默认即可。

如果编译成静态库的话则不选择BUILD\_SHARED\_LIBS如下：(默认我们选择编译动态库)

点击configure  
点击Generate生成编译目录；

#编译  
cd /work/opencv/opencv\_arm  
make

#安装  
make install

安装完后，/work/opencv/opencv\_arm/install路径下的所有文件就是编译安装的文  
件。

**2.5 Opencv编译安装后拷贝库文件到DF\_Vision\_SDK相应目录**

#使用cmake进行编译安装

cd /work/opencv\_pc\_install/lib #这个路径根据自己编译make install的安装路径

#拷贝文件到目录下

#64bit系统下

sudo cp \* -d /DF\_Vision\_SDK/linuxbin\_x64/ -rf

#32bit系统下

sudo cp \* -d /DF\_Vision\_SDK/linuxbin/ -rf

#打开动态库配置文件添加路径

vi /etc/ld.so.conf

在文本末尾添加运行时动态库查找路径，添加内容如下：

#64bit系统下

/DF\_Vision\_SDK/linuxbin\_x64/

#32bit系统下

/DF\_Vision\_SDK/linuxbin/

sudo ldconfig #进行动态库搜索（更新）

**2.6** **安装eclipse**

sudo apt-get install eclipse

**2.7安装cdt**

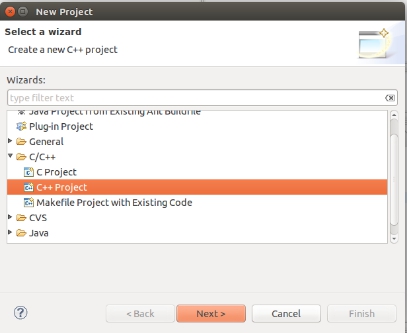
sudo apt-get install eclipse-cdt

**2.8** **启动eclipse**

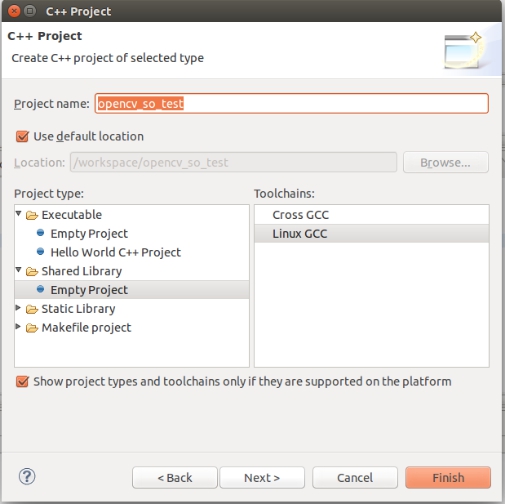
sudo eclipse -s

2.8.1新建动态库工程

file->new->project->C/C++->C++ Project

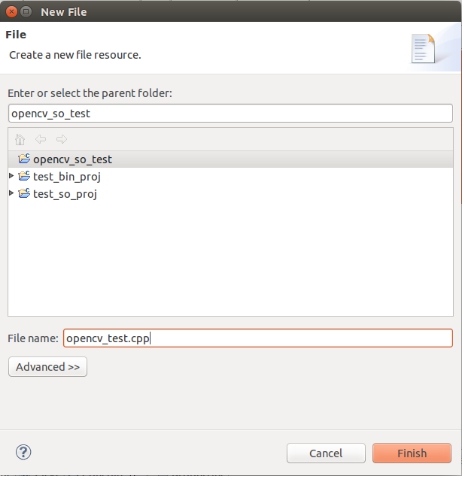


2.8.2选择动态库和编译工具



2.8.3添加文件：

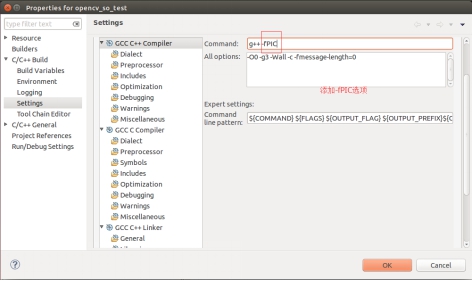
file->new->file



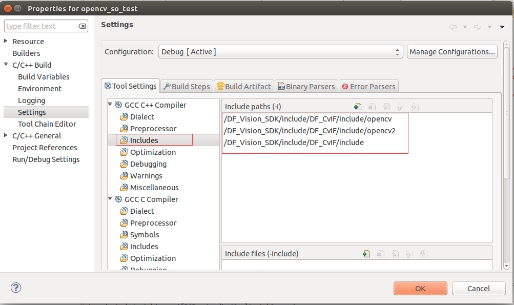
设置工程

右击工程->Properties->C/C++ Build->Settings

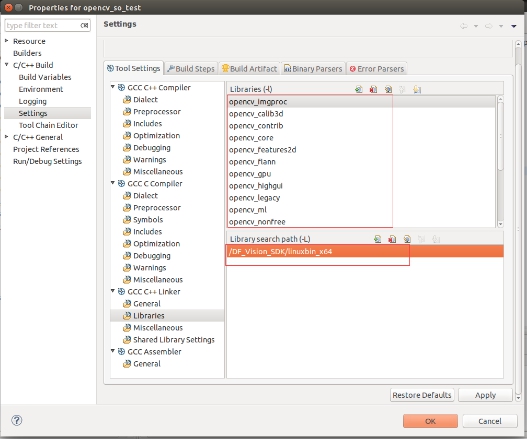
GCC C++ Compiler Command中添加-fPIC选项



GCC C++ Compiler和GCC C Compiler的includes添加对应的SDK头文件路径

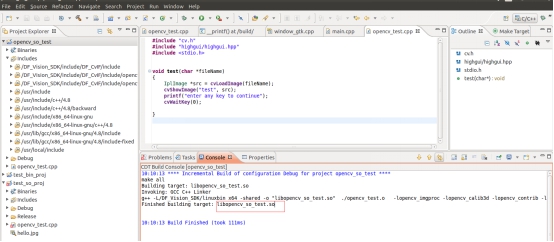


GCC G++ Linker添加对应的库路径和库名称



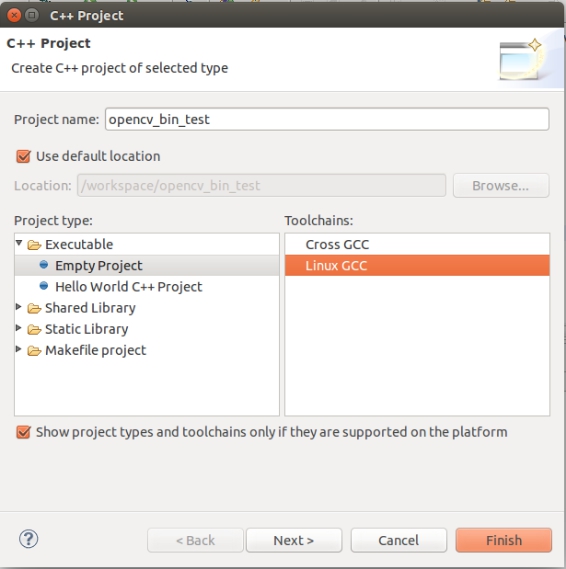
编译运行

选中工程，右击->Build Project即可；



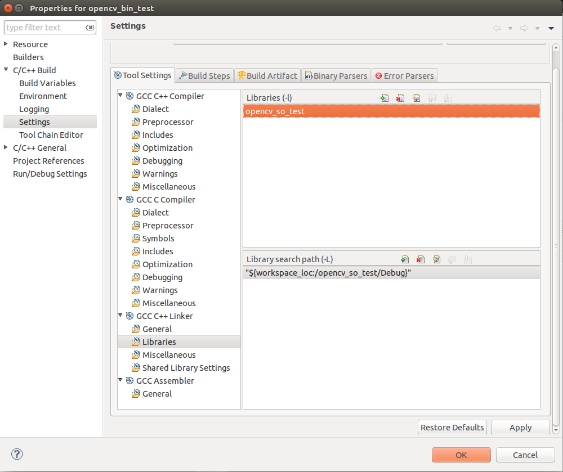
新建测试工程

同新建动态库工程，只是在选择工程类型时选择Executable工程即可；

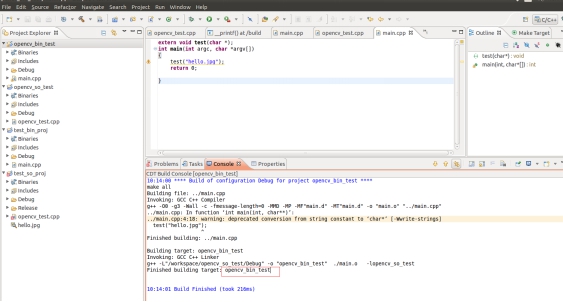


工程配置设置

添加要调试的库路径和名称



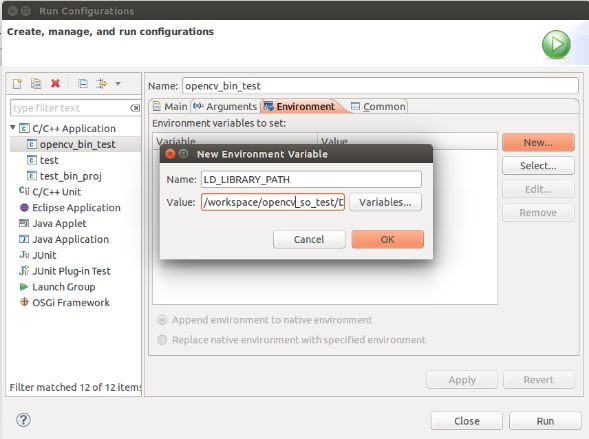
编译



调试环境设置：

右键->run as->run configuration->Environment->new

新建一个变量如：LD\_LIBRARY\_PATH,变量值为对应调试库的路径如：/workspace/opencv\_so\_test/Debug



调试

右键->run as->run configuration->run

