1 统一平台，便于兼容；

Windows　　建议 win7 x64

Windows 下各开发工具大致如下：

Visual studio　建议 2013

OpenCV　　建议 2.4.11

Halcon　　建议 2012

PCL　　建议 1.8.0

QT　　建议 5.5

Python 视情况选择 2.7 或 3.5，另 python 开发推荐使用 pycharm 软件，可视化与编辑等功能较全。

Ubuntu　　建议 Ubuntu 14.04 LTS

Ubuntu 下各开发工具大致如下：

OpenCV　　建议 2.4.11

PCL　　建议 1.8.0

QT　　建议 5.5

Python　　建议 2.7，python 开发推荐使用 sublime Text

2 选择合适的代码管理平台，明细化访问、读写、维护等各种权限，推荐使用 git 平台。

3 代码的规范性

建议从每一行代码的开发与编写过程中，就尽力规范化，有代码规范工具 StyleCorp，但不推荐使用。

3.1 命名规范

3.1.1 变量前缀：以变量实际名称为 Var 为例：

变量类型	变量名	变量类型	变量名	变量类型	变量名
int	nVar/iVar	float	fVar	double	dVar
bool	bVar	byte	btVar	string	strVar
char	cVar	unsigned char	ucVar	unsigned int	unVar

各类型指针前缀如下表：

指针类型	前缀	指针类型	前缀	指针类型	前缀
普通指针	ptr	shared_ptr	shr_ptr	unique_ptr	unq_ptr
weak_ptr	weak_ptr				

关于如 vector、list、map 的 STL 泛型容器一类变量，各容器具体前缀如下表：

容器类型	前缀	容器类型	前缀	容器类型	前缀
vector	vec	list	lst	set	set
queue	que	map	map	stack	stk

对于全局变量与局部变量的区分，全局变量加 g_前缀，成员变量加 m_前缀予以表示。

结构体名称前加大写字母 S 前缀，类名称前加大写字母 C 前缀，共用体名称前加大写字母 U 前缀。

3.1.2 变量起名规则

1 严禁局部变量与全局变量重名；

2 变量定义应尽可能接近第一次使用地方，并尽早销毁，减少内存消耗；

3 变量定义需初始化，尤其是指针类型变量，以避免出现未知风险，如内存泄漏、野指针，浮动变量等；

4 变量名体现出变量实际凸显的功能或用途，命名相对意义的变量，用明确的反义词对；

5 变量的实际意义可能包含多个单词，单词之间的区分可以使用 Unix 风格，即单词之间以”_”分隔，也可以使用 Windows 风格，即每个单词的第一个字母予以大写，但整体风格应保持一致

6 尽量用 const 替换掉#define

3.2 函数规范

1 函数定义时参数顺序应保持为，输入参数在前，输出参数在后，输入参数一般为值和常数引用，输出参数或输入/输出参数一般为非常数指针；

2 函数尽量定义为 bool 类型，并明确是否成功执行，给出正确与否的返回值与日志信息，对于需要对输入数据进行处理获得结果的函数，尽量将需要返回的数据通过引用方式传递；

3 类一定要自己完成类中成员的初始化，无论是否在构造函数内，以避免不可预知的问题；

4 可以多定义相关成员函数，但尽量避免对操作符进行重载；

5 函数以实现具体功能为主，如果没有特殊的需求，尽量控制在 40 行以内，如果过长不影响程序的运行的情况，可以将长函数予以分隔；

6 尽量避免定义参数缺省函数，以避免错误理解及运行结果；

7 尽量避免使用 RTTI，即 dynamic_cast 一类，推荐使用 virtual 方法；

8 在函数与函数之间、逻辑段落之间、变量申明和函数之间多加以空格予以区分；

9 在==、=、+、-、*、/、>、<、>=、<=、:这些

运算符两端加空格；标点符号如 for 循环内的";"前面不加空格，后面加一个空格；if、for、while 与括号之间加一个空格；多重括号运算用空格标明层次关系；->、.、++、--前后不加空格；数组名与[]之间及函数名和括号之间不加空格；尽量用.at()运算符替代[]运算符；

10 函数内每行代码不要超过 80 列，在逻辑分割点予以折行，并且折行内容尽量对齐；

11 逻辑判断时尽量将使用如 if (2 == x)的结构，并且如果判断是否为零，尽量使用！运算符；

3.3 代码注释

1 代码注释用英文编写，尽量避免中文及其他语言；

2 关键代码及部分过程需要进行注释，代码注释应在 20%以上；

3 注释和程序一同更新；

4 减少缩写，避免歧义；

5 对于协议标准或对 paper 方法的实现编程应加入标准文献的索引；

6 逻辑段落的注释放在段落前一行，语句的注释放在语句之后，注释与被注释的代码之间无空行；

7 头文件开头需要注释段落，头文件中函数声明前需要注释段落，全局变量需要详细注释，特殊功能的局部变量需要详细注释；

8 头文件开头注释段落参考模板如下：

```
/************************************************
Copyright (C), 20xx - 20xx, xxx Group
File name: xxx

Description: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Function List:
func1: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
func2: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

History:
Author: xxx Version: xxx Date: xxx Email: xxx
Description: Create

Author: xxx Version: xxx Date: xxx Email: xxx
Description: Fix xxx bugs
************************************************/
```

9 函数开头注释段落参考模板如下：

```
/************************************************
    Name: xxxx
    Description :xxxxxxxxxxxxxxxxx
    Para: xxxxxx [输入输出类型]
    Return: xxxx
    Notes: xxxx
************************************************/
```

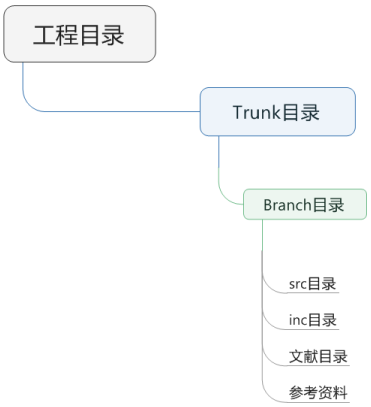

图 1.工程结构

3.4 文件规范

1 头文件包含依次为：类对应头文件、C 系统头文件、C++系统头文件、其他库头文件、本项目内头文件，且各部分头文件之间通过空行区分；

2 文件编译声明尽量使用前向声明，减少文件依赖；

3 尽量使用#ifndef #define 结构，避免使用#pragma once 方法；

4 工程结构

工程结构建议为四层树形结构，即工程目录、Trunk 目录、Branch 目录与具体文档目录。其中，Trunk 目录主要为大方向上的分类并列目录，Branch 目录为其中一个方向上具体一个技术点的目录，第四级目录为具体到各文件实现的目录，该目录中，需要存放的除了代码文件外，还应该有具体的文献索引或文件，技术手册或链接，开发参考的 URL 链接或具体文档资料。

```
CMAKE_MINIMUM_REQUIRED(VERSION 2.6) #CMake 最低版本要求

PROJECT(CircleDetect) #定义工程名

add_definitions(-std=c++11)
SET(CMAKE_BUILE_TYPE Realease) #指定编译类型，Debug 或 Realease
                    #Debug 会生成相关调试信息，可以使用 GDB 进行
                    #Realease 不会生成调试信息。当无法进行调试时可查看此处是否设置为 Debug。

SET(CMAKE_CXX_FLAGS_DEBUG "-g-Wall")   #指定编译器
                        # CMAKE_C_FLAGS_DEBUG ——— C 编译器
                        # CMAKE_CXX_FLAGS_DEBUG ——— C++ 编译器
                        #-g:只是编译器，在编译的时候，产生调试信息。
                        #-Wall：生成所有警告信息。
#ADD_SUBDIRECTORY(utility)   #添加子目录

FIND_PACKAGE(OpenCV REQUIRED)   #查找 opencv 包

if(WIN32)
ADD_EXECUTABLE(CircleDetect main.cpp CircleDetec.cpp SerialPort.cpp DataProcess.cpp )
else()
ADD_EXECUTABLE(CircleDetect main.cpp CircleDetec.cpp ManiFold_UART1.cpp DataProcess.cpp )   #这里括号里面的两个参数分别是工程项目名和我们要编译文件名
endif()

include_directories(${OpenCV_INCLUDE_DIRS})              #头文件
TARGET_LINK_LIBRARIES(CircleDetect ${OpenCV_LIBS})   #链接到 OpenCV 库，第一个参数为工程项目名
```