### 目录

[AI\_DETECT 类 API 文档 2](#_Toc18781)

[CAM\_USB 类 API 文档 5](#_Toc8095)

[Licence\_board 类 API 文档 7](#_Toc13706)

[Board\_pwm 类 API 文档 9](#_Toc3911)

[Board\_net 类 API 文档 11](#_Toc32698)

[MY\_FIFO 类 API 文档 13](#_Toc12751)

[ImageShow 类的 API 文档 15](#_Toc5753)

[uaCom 类的 API 文档 16](#_Toc4714)

[MainWindow 类的 API 文档 17](#_Toc29975)

[StyleInstaller 类的 API 文档 18](#_Toc6093)

[HeaderBar 类 API 文档 20](#_Toc10771)

# AI\_DETECT 类 API 文档

1. 类构造函数 AI\_DETECT::AI\_DETECT(void)

功能：初始化 AI\_DETECT 对象，读取配置文件并初始化模型和检测。

参数：无

返回值：无

1. 成员函数 bool AI\_DETECT::Model\_Reset()

功能：重新初始化模型和检测，并更新配置文件中的模型路径。

参数：无

返回值：- true - 模型重置成功 - false - 模型重置失败

1. int AI\_DETECT::Model\_init(rknn\_context \*ctx)

功能：加载模型文件并初始化模型。

参数：- ctx - 指向 rknn\_context 的指针

返回值：- 0 - 初始化成功 - -1 - 初始化失败

1. int AI\_DETECT::Detect\_init(rknn\_context ctx)

功能：初始化检测，包括查询 SDK 版本和输入输出数目，设置输入输出属性。

参数：- ctx - rknn\_context 对象

返回值：- 0 - 初始化成功 - -1 - 初始化失败

1. int AI\_DETECT::Detect\_run(Mat input\_image, Mat& output\_image)

功能：运行检测，对输入图像进行预处理并运行模型，获取输出结果并进行后处理。

参数：- input\_image - 输入图像 (Mat 类型)

- output\_image - 输出图像 (Mat 类型)，包含检测结果

返回值：- 检测结果（整数表示）

辅助函数

1. float Deqnt\_affine\_to\_f32(int8\_t qnt, float scale, float zero\_point)

功能：将量化值转换为浮点值。

参数：- qnt - 量化值 - scale - 缩放因子 - zero\_point - 零点

返回值：- 浮点值

1. int Detect\_head(float \*input\_data, int stride, float threshold, std::vector<BoxInfo> &boxes)

功能：处理模型输出的头部数据。

参数：- input\_data - 输入数据 - stride - 步长

- threshold - 阈值 - boxes - 检测框信息的容器

返回值：- 处理结果（整数表示）

1. void Detect\_draw(Mat& bgr, std::vector<BoxInfo>& result)

功能：在图像上绘制检测框。

参数：- bgr - 输入图像 (Mat 类型) - result - 检测框信息的容器

返回值：无

1. int Detect\_post(float threshold, float nms\_threshold, std::vector<BoxInfo>& bboxes, rknn\_output\* outputs)

功能：后处理检测结果。

参数：- threshold - 阈值

- nms\_threshold - 非极大值抑制阈值

- bboxes - 检测框信息的容器

- outputs - 模型输出数据

返回值：- 处理结果（整数表示）

1. void Para\_Save()

功能：保存参数到配置文件。

参数：无

返回值：无

1. void Threshold\_Save()

功能：保存阈值到配置文件。

参数：无

返回值：无

1. std::string Get\_Model\_Path()

功能：获取模型路径。

参数：无

返回值：- 模型路径（字符串）

1. std::string Get\_Model\_Version()

功能：获取模型版本。

参数：无

返回值：- 模型版本（字符串）

1. int Get\_Model\_Class()

功能：获取模型类别。

参数：无

返回值：- 模型类别（整数）

# CAM\_USB 类 API 文档

1. 类构造函数

CAM\_USB::CAM\_USB(void)

功能：初始化 CAM\_USB 对象，读取配置文件并初始化相机参数。

参数：无

返回值：无

2. 成员函数

2.1 void CAM\_USB::Cam\_init(void)

功能：初始化相机设备，包括查询设备、打开设备、设置缓存、增益和曝光参数。

参数：无

返回值：无

2.2 bool CAM\_USB::Convert2Mat(dvpFrame\* pFrameInfo, unsigned char\* pData, cv::Mat& srcImage)

功能：将相机帧数据转换为 OpenCV 的 Mat 格式。

参数：

- pFrameInfo - 帧信息（dvpFrame 类型）

- pData - 帧数据的内存首地址（unsigned char\* 类型）

- srcImage - 转换后的图像（cv::Mat 类型）

返回值：

- true - 转换成功

- false - 转换失败

2.3 cv::Mat CAM\_USB::Get\_frame(int a)

功能：获取相机帧数据，并根据输入参数设置分辨率。

参数：- a - 分辨率参数（整数类型），1 表示低分辨率，其他值表示高分辨率

返回值：- 图像帧数据（cv::Mat 类型）

2.4 float CAM\_USB::CalculateTheVariance(bool sw)

功能：计算图像的方差，用于评估图像质量。

参数：- sw - 选择区域参数（布尔类型），false 表示第一次调光区域，true 表示第二次调光区域

返回值：- 图像方差（浮点数）

2.5 std::string CAM\_USB::Save\_img(cv::Mat src, bool sw)

功能：保存图像到指定路径，并根据输入参数选择保存原图或结果图。

参数：- src - 要保存的图像（cv::Mat 类型）

- sw - 图像类型参数（布尔类型），false 表示保存原图，true 表示保存结果图

返回值：- 图像保存路径（字符串）

2.6 void CAM\_USB::Para\_Save(void)

功能：将当前参数保存到配置文件中。

参数：无

返回值：无

2.7 void CAM\_USB::Cam\_release(void)

功能：释放相机资源，停止和注销相机设备。

参数：无

返回值：无

3. 析构函数

CAM\_USB::~CAM\_USB(void)

功能：销毁 CAM\_USB 对象，释放相机资源。

参数：无

返回值：无

# Licence\_board 类 API 文档

1. 类构造函数

Licence\_board::Licence\_board()

描述：初始化 Licence\_board 对象并从配置文件中加载参数。

参数：无

返回：无

2. 获取 CPU ID

string Licence\_board::Get\_CPU\_ID()

描述：获取 CPU 的序列号并保存到配置文件中。

参数：无

返回：CPU 序列号（string）

3. 创建授权码

string Licence\_board::Create\_Key()

描述：根据 CPU ID 创建授权码。

参数：无

返回：授权码（string）

1. 软件激活

bool Licence\_board::Software\_Activation(string input\_key)

描述：激活软件。

参数：输入的授权码（string）

返回：激活结果（bool）

5. 软件检查

void Licence\_board::Software\_Check()

描述：检查软件是否已授权。

参数：无

返回：无

6. 获取 ID

string Licence\_board::Get\_ID()

描述：获取 CPU ID。

参数：无

返回：CPU ID（string）

7. 获取授权码

string Licence\_board::Get\_Key()

描述：获取授权码。

参数：无

返回：授权码（string）

8. 保存 ID 参数

void Licence\_board::Para\_Save\_ID()

描述：保存 CPU ID 到配置文件。

参数：无

返回：无

9. 保存授权码参数

void Licence\_board::Para\_Save\_Key()

描述：保存授权码到配置文件。

参数：无

返回：无

10. 类析构函数

Licence\_board::~Licence\_board()

描述：销毁 Licence\_board 对象。

参数：无

返回：无

11. Licence\_board 全局对象

Licence\_board licence\_board

描述：Licence\_board 类的全局对象。

参数：无

返回：无

# Board\_pwm 类 API 文档

1. 类构造函数

Board\_pwm::Board\_pwm(void)

描述：初始化 Board\_pwm 对象并从配置文件中加载参数，初始化 PWM 通道。

参数：无

返回：无

2. 更新参数值

bool Board\_pwm::UpdateParameterValue(const std::string paramName, int intValue)

描述：更新指定参数的值并保存到配置文件中。

参数：

- paramName：参数名称（string）

- intValue：参数值（int）

返回：更新结果（bool）

3. 设置 PWM 值

void Board\_pwm::Set\_val(bool sw\_p, int val)

描述：设置 PWM 通道的值并更新参数。

参数：

- sw\_p：选择 PWM 通道（bool）

- val：PWM 值（int）

返回：无

4. 获取 PWM 值

int Board\_pwm::Get\_val(bool sw\_p)

描述：获取 PWM 通道的值。

参数：

- sw\_p：选择 PWM 通道（bool）

返回：PWM 值（int）

5. PWM 调光

void Board\_pwm::Pwm\_light(bool sw\_p)

描述：根据 PWM 值调整灯光。

参数：

- sw\_p：选择 PWM 通道（bool）

返回：无

6. 关闭 PWM

void Board\_pwm::Pwm\_off()

描述：关闭 PWM 通道。

参数：无

返回：无

7. 重置参数

void Board\_pwm::Rest\_param()

描述：重置最佳值和最佳 PWM。

参数：无

返回：无

8. 自动调光

void Board\_pwm::Auto\_light(bool sw\_p)

描述：根据计算的方差自动调整灯光。

参数：

- sw\_p：选择 PWM 通道（bool）

返回：无

9. PWM 检查

void Board\_pwm::PWM\_Check()

描述：检查 PWM 通道的状态。

参数：无

返回：无

10. 类析构函数

Board\_pwm::~Board\_pwm(void)

描述：释放 PWM 通道。

参数：无

返回：无

全局对象

Board\_pwm boardpwm

描述：Board\_pwm 类的全局实例对象。

# Board\_net 类 API 文档

1. 类构造函数

Board\_net::Board\_net(void)

描述：初始化 Board\_net 对象并设置网络连接，加载 WiFi 配置。

参数：无

返回：无

2. 连接 WiFi

void Board\_net::Wifi\_connect()

描述：连接到指定的 WiFi 网络。

参数：无

返回：无

3. 获取 IP 地址

bool Board\_net::GetIPAddress(bool ip\_switch)

描述：获取指定网络接口的 IP 地址。

参数：- ip\_switch：选择网络接口（true 为 WiFi，false 为 LAN）

返回：获取结果（bool）

4. 检查网络连接

bool Board\_net::Is\_connect()

描述：检查是否成功连接到网络。

参数：无

返回：连接状态（bool）

5. 保存参数

void Board\_net::Para\_Save()

描述：保存 WiFi 名称和密码到配置文件。

参数：无

返回：无

6. 类析构函数

Board\_net::~Board\_net(void)

描述：析构 Board\_net 对象。

参数：无

返回：无

全局对象

Board\_net boardwifi

描述：Board\_net 类的全局实例对象。

# MY\_FIFO 类 API 文档

1. 构造函数

MY\_FIFO(QObject \*parent)

描述: MY\_FIFO 类的构造函数。

参数:

- QObject \*parent: 父对象。

1. 公共方法

int AI\_open\_fifo(void)

描述: 打开 FIFO 进行通信。

返回值:

- 成功时返回 1。

- 如果打开用于读取的 stj\_fifo\_2 失败，返回 -1。

- 如果打开用于写入的 stj\_fifo\_1 失败，返回 -2。

1. int AI\_read\_fifo(u\_int8\_t \*buffer, u\_int16\_t len)

描述: 从 FIFO 读取数据。

参数:

- u\_int8\_t \*buffer: 存储读取数据的缓冲区。

- u\_int16\_t len: 要读取的数据长度。

返回值:

- 读取的字节数。

- 如果读取失败，返回 -1。

- 如果读取被中断，返回 -2。

1. int AI\_write\_fifo(u\_int8\_t \*buffer, u\_int16\_t len)

描述: 向 FIFO 写入数据。

参数:

- u\_int8\_t \*buffer: 包含要写入数据的缓冲区。

- u\_int16\_t len: 要写入的数据长度。

返回值:

- 写入的字节数。

- 如果写入失败，返回 -1。

1. u\_int8\_t data\_buffer\_sum(u\_int8\_t\* dataBuf, u\_int16\_t len)

描述: 计算数据缓冲区的校验和。

参数:

- u\_int8\_t\* dataBuf: 数据缓冲区。

- u\_int16\_t len: 数据缓冲区的长度。

返回值: 数据缓冲区的校验和。

1. void Fifo\_Receive\_Handler(void)

描述: 处理从 FIFO 接收的数据。

参数：无

返回值：无

1. void flag\_solve(void)

描述: 根据标志处理接收到的数据。

参数：无

返回值：无

1. void Fifo\_Send\_Handler(void)

描述: 处理数据发送。

参数：无

返回值：无

# ImageShow 类的 API 文档

1. 构造函数

ImageShow::ImageShow(QObject \*parent = nullptr)

- 描述: 初始化 ImageShow 对象。

- 参数:

- QObject \*parent: 可选参数，指向父对象的指针，默认为 nullptr。

- 返回: 无

1. 公共方法

QImage ImageShow::matToQImage(const cv::Mat &src)

- 描述: 将 OpenCV 的 Mat 格式图像转换为 Qt 的 QImage 格式图像。

- 参数:

- const cv::Mat &src: 输入的 OpenCV Mat 图像。

- 返回: 转换后的 QImage 图像。

1. void ImageShow::working()

- 描述: 持续获取图像并将其转换为 QImage，然后通过信号 imageReady 发送到 UI。

- 参数: 无

- 返回: 无

1. void ImageShow::imageReady(const QImage &display\_img)

- 描述: 图像准备就绪信号。该信号在新的图像准备好并转换为 QImage 后被发射。

- 参数:

- const QImage &display\_img: 转换后的 QImage 图像。

- 返回: 无

私有成员变量

- bool work\_flag: 工作标志，用于控制 working 方法中的循环。

- USBCam usb\_cam: USB 摄像头对象，用于获取图像帧。

依赖库

- OpenCV: 用于图像处理和获取图像帧。

- Qt: 用于 GUI 和图像显示。

# uaCom 类的 API 文档

1. 构造函数

uaCom::uaCom(QObject \*parent = nullptr)

描述: 初始化 `uaCom` 对象。

参数:

- `QObject \*parent`: 可选参数，指向父对象的指针，默认为 `nullptr`。

返回: 无

2. void uaCom::working()

描述: 持续处理 FIFO 管道通信，包括数据发送和接收，并更新相关标志。

参数: 无

返回: 无

私有成员变量

- `MyFIFO my\_fifo`: FIFO 管道对象，用于通信处理。

依赖库

- Qt: 用于 GUI 和信号槽机制。

方法详细描述

1. uaCom::uaCom(QObject \*parent = nullptr)

- 初始化 `uaCom` 对象。父对象默认为 `nullptr`。

2. uaCom::working()

- 打开 FIFO 管道。如果打开失败，设置 `fifo\_failed` 标志为 `true` 并退出方法。

- 初始化 `SendData` 数组并设置 `SendDataLen`。

- 调用 `Fifo\_Send\_Handler` 发送数据。

- 进入循环：

- 调用 `flag\_solve` 处理标志。

- 调用 `Fifo\_Receive\_Handler` 接收数据。

- 再次调用 `flag\_solve` 处理标志。

- 更新 `ST\_fault\_flag` 标志，检查各种状态标志是否存在故障。

# MainWindow 类的 API 文档

1. MainWindow::MainWindow(QWidget \*parent)

描述：构造函数，初始化主窗口，设置 UI，连接信号和槽，并配置初始设置。

参数:

- `parent`: `QWidget\*` 类型，父窗口，默认值为 `nullptr`。

返回值: 无。

1. MainWindow::~MainWindow()

描述：析构函数，处理清理并删除 UI。

参数: 无。

返回值: 无。

1. bool MainWindow::eventFilter(QObject \*obj, QEvent \*event)

描述：事件过滤器，用于处理特定 UI 组件的自定义事件。用于管理特定部件获得焦点时的键盘显示。

参数:

- `obj`: `QObject\*` 类型，指向产生事件的对象。

- `event`: `QEvent\*` 类型，指向要处理的事件。

返回值:

`bool` 类型，指示事件是否被处理。如果事件被处理，返回 `true`，否则返回 `false`。

# StyleInstaller 类的 API 文档

1. static StyleInstaller\* instance()

描述：获取 `StyleInstaller` 类的单例实例。如果实例不存在，则创建一个新的实例。

参数: 无。

返回值:

- `StyleInstaller\*` 类型，返回类的唯一实例指针。

1. static void deleteInstance()

描述：删除 `StyleInstaller` 类的单例实例。

参数: 无。

返回值: 无。

1. void StyleInstaller::install(const StyleString& s)

描述：安装自定义样式表。

参数:

- `s`: `const StyleString&` 类型，包含样式颜色配置的结构体。

返回值: 无。

1. void StyleInstaller::install(const QString &path)

描述：从文件加载并安装样式表。

参数:

- `path`: `const QString&` 类型，样式表文件的路径。

返回值: 无。

1. void StyleInstaller::install(UiStyle ui)

描述：安装预定义样式。

参数:

- `ui`: `UiStyle` 枚举类型，指示要安装的预定义样式类型。

返回值: 无。

1. void StyleInstaller::append(const QString &style)

描述：追加自定义样式到现有样式表。

参数:

- `style`: `const QString&` 类型，追加的样式字符串。

返回值: 无。

1. StyleInstaller::StyleInstaller()

描述：构造函数，初始化 `StyleInstaller` 实例。

参数: 无。

返回值: 无。

# HeaderBar 类 API 文档

1. 构造函数

HeaderBar::HeaderBar(QWidget \*parent)

描述：构造一个 HeaderBar 实例，并初始化界面和按钮。

参数：

- `parent`：父级窗口指针。

1. 析构函数

HeaderBar::~HeaderBar()

描述：销毁 HeaderBar 实例，释放内存。

参数：无。

1. bool HeaderBar::registerName(QToolButton \*btn)

描述：注册工具按钮及其点击事件的处理函数。

参数：

- `btn`：要注册的工具按钮指针。

返回值：

- `true`：成功注册按钮。

- `false`：注册失败。

1. bool HeaderBar::addPage()

描述：向应用添加新页面。

参数：无。

返回值：

- `true`：成功添加页面。

- `false`：添加页面失败。

1. void HeaderBar::onIndexChanged()

描述：处理工具按钮点击事件，发射 `pageChanged` 信号以切换页面。

参数：无。

返回值：无。