**End Of Line2.0模块设计文档**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Change** | **Version** | **Date** |
| 闫硕 | 初始版本 | 2.0 | 2017-02-23 |
| 闫硕 | 完成版本 | 2.0 | 2017-04-05 |

目录

[1. EOL2.0功能定义 4](#_Toc479329159)

[1.1. EOL2.0设计要求 4](#_Toc479329160)

[1.2. EOL2.0模块优点 4](#_Toc479329161)

[1.3. EOL2.0标定工位布置 4](#_Toc479329162)

[1.4. EOL2.0模块处理流程 5](#_Toc479329163)

[2. EOL2.0 方案设计 6](#_Toc479329164)

[2.1. 标定板样式设计 6](#_Toc479329165)

[2.2. 标定工位样式设计 7](#_Toc479329166)

[2.2.1. 左右三块棋盘格时标定工位设计 7](#_Toc479329167)

[2.2.2. 左右4块棋盘格时标定工位设计 8](#_Toc479329168)

[2.3. 模块错误码 9](#_Toc479329169)

[2.4. 数据结构设计 9](#_Toc479329170)

[2.5. 接口设计 10](#_Toc479329171)

[3. 核心函数介绍 12](#_Toc479329172)

[3.1. 核心函数处理流程图 12](#_Toc479329173)

[3.1.1. 算法整体流程图 12](#_Toc479329174)

[3.1.2. EOL\_init函数流程图 12](#_Toc479329175)

[3.1.3. EOL\_Process函数流程图 13](#_Toc479329176)

[3.1.4. Eol\_deinit函数流程图 14](#_Toc479329177)

[3.1.5. Eol\_chessboard\_detect函数流程图 14](#_Toc479329178)

[3.1.6. Eol\_absolute\_estimator\_driver函数流程图 15](#_Toc479329179)

[3.1.7. cvFindChessboardCorners3 函数流程图 16](#_Toc479329180)

[3.2. 核心函数说明 16](#_Toc479329181)

[3.2.1. EOL\_Process中调用函数说明 16](#_Toc479329182)

[3.2.2. 角点检测模块重要函数说明 18](#_Toc479329183)

[3.2.3. 优化器模块主要函数说明 23](#_Toc479329184)

[4. 其他 25](#_Toc479329185)

[4.1. 配置项位置 25](#_Toc479329186)

[4.2. 移植所需文件 25](#_Toc479329187)

[4.3. 待加入功能 26](#_Toc479329188)

# EOL2.0功能定义

## EOL2.0设计要求

EOL2.0相机线下标定(End of Line)系统模块主要用于完成全景拼接系统的流水线下线标定，在对该模块进行设计时，应满足如下要求：

(1) 占地空间有限：考虑到工厂面积有限，EOL模块不应占用过大面积。

(2) 有限时间内完成：下线标定通常为于汽车生产流水线中一环，故算法耗时不应太高，以免造成产品线产品积压，影响产能。

(3) 鲁棒性要求：标定模块应对一定范围内的光照、镜头安装位置、车辆停放位置等偏差保持鲁棒，从而保证标定的成功率。

(4) 自动化要求：标定模块最好自动进行，应尽可能少的减少人为操作，已减少出错概率。

(6) 不同车型间易于切换：同一标定工位应支持车辆长宽近似的不同车型以提高标定工位的可重用性。

## EOL2.0模块优点

相机线下标定(End of Line, EOL)系统应能准确检测一组前、后、左、右4幅图像中标定板的角点，并根据检测结果输出相机外参，实现图像无缝拼接。较EOL1.0而言，

(1) EOL2.0采用基于棋盘格的自动角点检测算法代替EOL1.0中的方块检测方法，该方法可在光纤较暗时正确检测棋盘格中角点，有效的提高了光照较差时角点检测算法鲁棒性。

(2) EOL2.0 在对相机参数进行优化时，加入相机内参进行优化并将世界坐标系中的投影误差加入LM算法的目标函数，该优化可确保当某幅图像棋盘格发生漏检时标定算法正常收敛，有效的提高了优化器的鲁棒性及准确性。

## EOL2.0标定工位布置

标定现场应提供长为8.8m，宽为5.8m的标定场地用于EOL相机标定，场地无杂物且现场灯光均匀明亮。标定工位中标定板应严格按照图1.1进行布置，以确保标定结果的准确性。标定场地不允许存在非标定棋盘格干扰，标定进行时，不允许障碍物对棋盘格进行遮挡。



图1.1 标定场地布置图

## EOL2.0模块处理流程

该模块的输入、处理流程及输出如下：

**模块输入：**

(1) 于标定场地采集的包含标定棋盘格的4幅原始图像；

(2) 包含相机内外参数及棋盘格在世界坐标系中位置的SMC。

**模块处理流程：**

(1) 角点检测及坐标计算：检测并确定棋盘格中角点在图像中的坐标；

(2) 世界坐标计算：通过SMC计算出各棋盘格中角点在世界坐标系中的坐标；

(3) 优化器优化：基于(1)与(2)中角点的一一对应关系建立方程组，采用LM算法对相机内外参进行优化。

**模块输出：**

优化后的相机内外参。

# EOL2.0 方案设计

## 标定板样式设计

图1.1中，标定工位中两相机公用部分采用存在4个角点了3\*3棋盘格的设计，较仅存在1个角点2\*2棋盘格而言，该设计中角点间拓扑结构可用来排除干扰点的影响有效提高角点检测算法的鲁棒性。此外由于该棋盘格通常位于鱼眼图像中边缘位置，该设计在保留了角点间拓扑结构的前提下最大限度的增大了棋盘格在图像中的面积，提高了角点的检测率。

将标定板的可配置参数设计如图2.1所示，其中点 (x, y) 表示棋盘格左上角点在棋盘格中的坐标，inner\_quad\_width 和 inner\_quad\_height 分别表示棋盘格中方格的宽和高，pattern\_width 和 pattern\_height 分别表示棋盘格的宽和高，上述参数的单位均为mm。该参数以数组的形式在cvcalibinit3.h头文件中进行初始化，一经确定，不得修改。



图2.1 棋盘格配置参数

## 标定工位样式设计

### 左右三块棋盘格时标定工位设计

由于不同车型间车宽变化不大，故前后相机中3块标定板即可保证保证精度。对左右相机而言，不同车型间车长变化较大，故当车身长度 <= 5.2m时，左右侧铺设三块棋盘格即可满足标定精度要求。图2.2中所标识距离即为SMC中可配置参数，当车型发生变化时，可适当调整并在SMC中如实填写该参数。标定工位设计如图2.2所示：

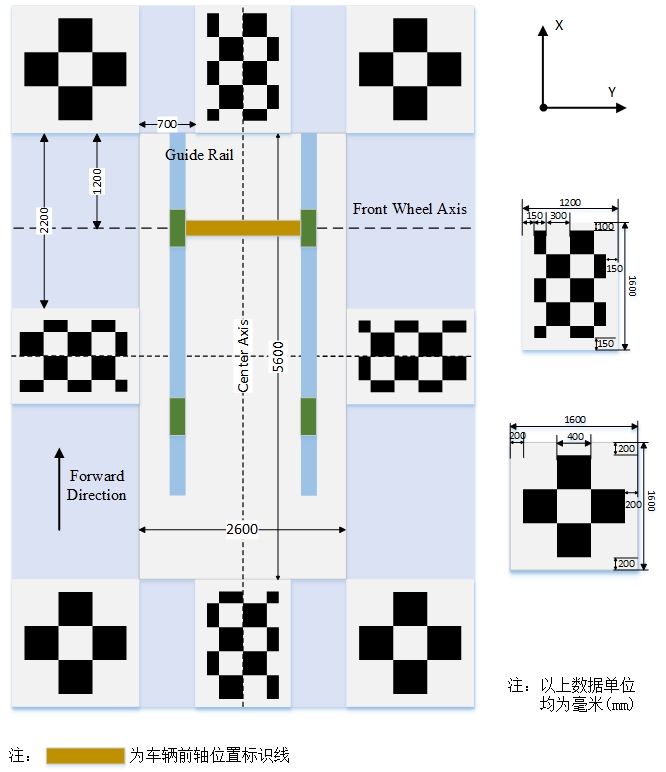


图2.2标定工位设计(左右3块标定板)

此时，对应的SMC中，Bev\_Eol\_Param参数配置如表2\_1所示。当标定工位中标定板铺设位置发生变化时SMC中数据即表2\_1中配置也应随之变化。

表2\_1 SMC配置(左右3块标定板)

|  |  |  |
| --- | --- | --- |
| **变量名称** | **变量值** | **变量意义** |
| BOARD\_VEH\_FLBOARD2CENTRALAXIS | 1300 | 前左板到车辆纵向中轴的距离 |
| BOARD\_VEH\_FLBOARD2FWHEEL | 1200 | 前左板到车辆前轮的距离 |
| LR\_BOARD\_NUM | 3 | 左右侧棋盘格数目 |
| STATION\_HEIGHT | 5600 | 停车工位长度 |
| STATION\_WIDTH | 2600 | 停车工位宽度 |
| BOARD\_FLBOARD2FCBOARD | 700 | 前左板到前中板的距离 |
| BOARD\_FLBOARD2LUCBOARD | 2200 | 前左板到左中板的距离 |
| BOARD\_FLBOARD2LLCBOARD | - | 无效 |

### 左右4块棋盘格时标定工位设计

当车身长度 >5.2m时，由于车长太大，左右侧需铺设四块棋盘格，方可满足标定精度要求。详细设计如图2.3所示：

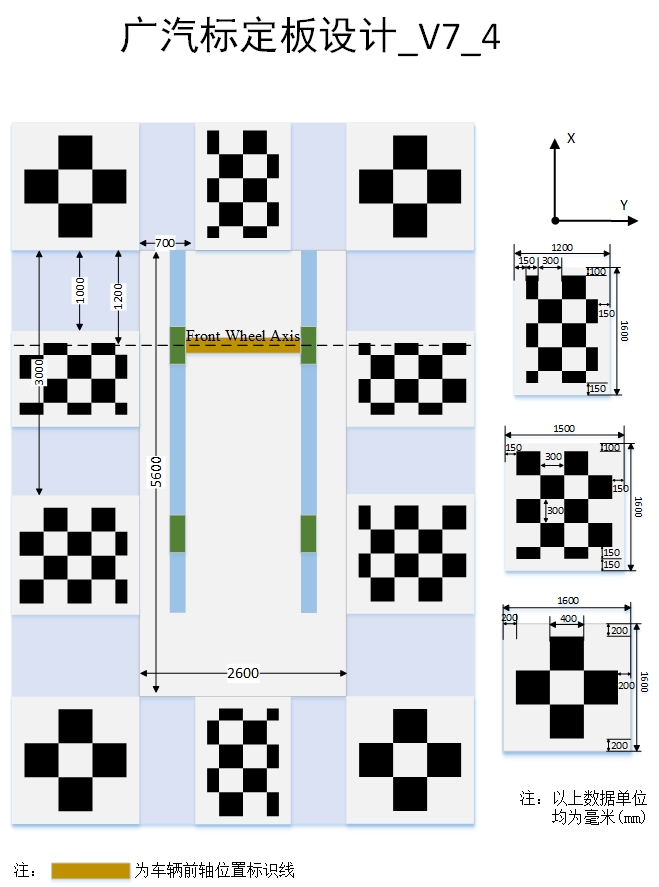


图2.3标定工位设计(左右4块标定板)

与之对应的SMC配置图标2\_2所示：

表2\_2 SMC配置(左右4块标定板)

|  |  |  |
| --- | --- | --- |
| **变量名称** | **变量值** | **变量意义** |
| BOARD\_VEH\_FLBOARD2CENTRALAXIS | 1300 | 前左板到车辆纵向中轴的距离 |
| BOARD\_VEH\_FLBOARD2FWHEEL | 1200 | 前左板到车辆前轮的距离 |
| LR\_BOARD\_NUM | 4 | 左右侧棋盘格数目 |
| STATION\_HEIGHT | 5600 | 停车工位长度 |
| STATION\_WIDTH | 2600 | 停车工位宽度 |
| BOARD\_FLBOARD2FCBOARD | 700 | 前左板到前中板的距离 |
| BOARD\_FLBOARD2LUCBOARD | 1000 | 前左板到左中上板的距离 |
| BOARD\_FLBOARD2LLCBOARD | 3000 | 前左板到左中下板的距离 |

## 模块错误码

该模块的错误返回码定义在eol\_landmark\_detector.h文件中，其意义定义表2\_4所示，

表 2\_4 模块错误码

|  |  |  |  |
| --- | --- | --- | --- |
| **变量名** | **值** | **变量名** | **值** |
| EOL\_SUCCESS | 768 | CALIB\_DATA\_BUFFER\_SIZE\_ERROR | 783 |
| LOAD\_INIT\_IMG\_FAIL | 769 | LM\_FRONT\_ERROR\_OVERSIZE | 784 |
| LOAD\_SMC\_FAIL | 770 | LM\_REAR\_ERROR\_OVERSIZE | 785 |
| SMC\_CONFIG\_ERROR | 771 | LM\_LEFT\_ERROR\_OVERSIZE | 786 |
| SMC\_IMG\_NOT\_COMPATIBLE | 772 | LM\_RIGHT\_ERROR\_OVERSIZE | 787 |
| LOAD\_IMG\_FAIL | 773 | LM\_FRONT\_TRANS\_OVERSIZE | 788 |
| MEM\_MALLOC\_FAIL | 774 | LM\_REAR\_TRANS\_OVERSIZE | 789 |
| UNDEFINED\_VEHICLE\_TYPE | 775 | LM\_LEFT\_TRANS\_OVERSIZE | 790 |
| OPEN\_FILE\_ERROR | 776 | LM\_RIGHT\_TRANS\_OVERSIZE | 791 |
| FRONT\_IMG\_CORNER\_NUM\_WRONG | 777 | LM\_FRONT\_CENTER\_OVERSIZE | 792 |
| REAR\_IMG\_CORNER\_NUM\_WRONG | 778 | LM\_REAR\_CENTER\_OVERSIZE | 793 |
| LEFT\_IMG\_CORNER\_NUM\_WRONG | 779 | LM\_LEFT\_CENTER\_OVERSIZE | 794 |
| RIGHT\_IMG\_CORNER\_NUM\_WRONG | 780 | LM\_RIGHT\_CENTER\_OVERSIZE | 795 |
| EOL\_CB\_WRONG\_CORNER\_NUM | 781 | INSUFFICIENT\_CORNER\_NUM | 796 |
| CONVERGENCE\_FAIL | 782 |  |  |

## 数据结构设计

该模块中所用到的数据结构结构体在EOL\_init中进行内存分配，在EOL\_deinit函数中进行内存释放，结构体的定义如表2\_3所示，

表 2\_3 标定模块数据结构

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Eol\_Param\* p\_Eol\_param  (eol\_pose\_estimator.h)  /\*station level chessboard pattern config \*/ | Car\_type car\_type;  /\*enum type\*/ | GAC | | |
| JAC | | |
| GEELY | | |
| OFILM | | |
| int32\_t station\_height; | | | |
| int32\_t station\_width; | | | |
| CbPattern\_group cb\_pattern\_group[4]  (eol\_pose\_estimator.h)  /\*camera level chessboard pattern config \*/ | int32\_t valid\_num; | | |
| Rect2f roi\_mask[4]; | | |
| Point2f refer\_point[4]; | | |
| chessBoard\_pattern  pattern[4]  (cvcalibinit3.h) | int32\_t valid\_corner\_num[4]; | |
| int32\_t rows; | |
| int32\_t cols; | |
| int32\_t flag[4][4]; | |
| Point2f corner\_point[4][4]; | |
| Point2f ground\_corner\_point[4][4]; | |
| EOL\_pattern\_config  pattern\_config;  (cvcalibinit3.h) | float32\_t top\_left\_corner\_x; |
| float32\_t top\_left\_corner\_y; |
| int32\_t inner\_quad\_height; |
| int32\_t inner\_quad\_width; |
| int32\_t pattern\_height; |
| int32\_t pattern\_width; |
| int32\_t left\_boundary\_to\_axis; |
| chessBoard\_corner\_config  chessboard\_config;  (cvcalibinit3.h) | CvSize board\_size; |
| int32\_t in\_number\_of\_corners; |
| int32\_t rect\_mask\_num; |
| int32\_t reference\_pointNum; |
| int32\_t camid; |
| float32\_t width\_ratio; |
| float32\_t height\_ratio; |
| float32\_t RectMinAreaRatio; |
| int32\_t reflection; |
| int32\_t gloabal\_min\_dilate; |

## 接口设计

该模块主要有EOL\_init，EOL\_process，EOL\_deinit及EOL\_GetResultProcess四个接口，现将各接口的函数定义及功能说明如下：

(1) EOL\_init: 该接口主要实现内存分配及参数初始化。

/\*

Function Name: EOL\_init

Function Function : Malloc ppEolHandle memory and init chessboard config

Input :

ppEolHandle : P\_EOL\_Param structure, invisible out of this module

pSMC : The SMC information

Return : error code described in eol\_landmark\_detector.h

\*/

int EOL\_init(void\*\* ppEolHandle, Smc\_Cal\_T\* pSMC)；

(2) EOL\_process: 该接口为模块的核心处理函数。

/\*

Function Name: EOL\_Process

Function Function : detect corners, set world coordinate and optimize camera parameters

Input :

sProcessImg\_in :The four camera's image with chessboard patterns

pSMC : The SMC information

ppEolHandle : P\_EOL\_Param structure, invisible out of this module

Return : Error code described in eol\_landmark\_detector.h

Note : The Function includes three important parts:

1. chessboard detection

2. calc world coordinate by smc input

3. optimize camera extrinsic params r t and camera intrinsic param xc, yc(if necessary)

\*/

int EOL\_Process(IplImage\* sProcessImg\_in[4], Smc\_Cal\_T\* pSMC, void\*\* ppEolHandle)

(3) EOL\_deinit: 内存释放函数。

/\*

Function Name: EOL\_deinit

Function Function : release memory malloced in EOL\_init

Input :

ppEolHandle : P\_EOL\_Param structure, invisible out of this module

Return : Void

\*/

void EOL\_deinit(void\*\* ppEolHandle)

(4) EOL\_GetResultProcess: 该接口将SMC结构体中内外参进行更新。

/\*

Function Name: EOL\_GetResultProcess

Function Function : Get the optimized center for other module

Input :

cam\_pose : For updating camera extrinsic params

cam\_center: Camera center array

result : For updating camera center params

Return : Void

\*/

void EOL\_GetResultProcess(str\_avm\_pose\_t\* cam\_pose, float64\_t\* cam\_center eol\_result\* result)

# 核心函数介绍

## 核心函数处理流程图

### 算法整体流程图

算法整体流程图如图2.4所示。其中算法所需要的内存在EOL\_init中进行申请，主要处理工作在EOL\_Process完成，算法完成后调用EOL\_deinit进行内存释放。



图2.4 算法整体流程图

### EOL\_init函数流程图

EOL\_init函数主要完成函数结构体所需内存的分配，以及将SMC输入的距离配置参数转换为棋盘格相对于世界坐标系中的坐标以方便后续操作。算法流程如图2.5所示。



图2.5 EOL\_init流程图

### EOL\_Process函数流程图

EOL\_Process函数完成算法的核心功能，并将最终的相机内外参进行更新，该算法的流程如图2.6所示，



图2.6 EOL\_Process流程图

### Eol\_deinit函数流程图

该函数主要完成内存释放功能，其主要处理流程如图2.7所示，



图2.7 EOL\_deinit流程图

### Eol\_chessboard\_detect函数流程图

该函数为EOL2.0模块的核心函数，主要完成图像中角点检测功能，该算法的主要处理流程如图2.8所示，



图2.8 Eol\_chessboard\_detect流程图

### Eol\_absolute\_estimator\_driver函数流程图

该函数为EOL2.0模块中优化器的核心函数，主要完成内外参优化功能，该算法的主要处理流程如图2.9所示，



图2.9 Eol\_absolute\_estimator\_driver函数流程图

### cvFindChessboardCorners3 函数流程图

该函数即为图2.8流程图中Corner Detect函数，主要完成角点检测功能， 其处理流程如图2.10所示，



图2.10 cvFindChessboardCorners3函数流程图

## 核心函数说明

### EOL\_Process中调用函数说明

该小节主要完成对EOL\_Process中调用的主要函数的进行介绍。

(1) EOL\_chessboard\_detect函数为角点检测入口函数，该函数完成角点检测功能并将结果写入结构体。

/\*

Function Name: EOL\_Process

Function Function : The entrance of chessboard corner detect function

Input :

sProcessImg\_in: The four camera's image with chessboard patterns

p\_Eol\_param : P\_EOL\_Param structure

Return : Error code described in eol\_landmark\_detector.h

\*/

int32\_t EOL\_chessboard\_detect(IplImage\* sProcessImg\_in[4], P\_EOL\_Param p\_Eol\_param)

(2) EOL\_calculate\_ground\_coordinate函数完成图像中各角点世界坐标的计算

/\*

Function Name : EOL\_calculate\_ground\_coordinate

Function Function : Calculate corner point's world coordinate via input smc config params

Input :

p\_Eol\_param : Eol config param

Return : Void

\*/

void EOL\_calculate\_ground\_coordinate(P\_EOL\_Param p\_Eol\_param)

(2) EOL\_remove\_outliers函数可移除误检的角点

/\*

Function Name: EOL\_remove\_outliers

Function Function : Cal re-projection error and remove outliers

Input :

pSmc : pSmc

avm\_cam\_bev : camera intrinsic param

p\_Eol\_param : p\_Eol\_param

world\_model: describe the relationship between vehicle coordinate system and world

coordinate system

Return : error code

\*/

int32\_t EOL\_remove\_outliers( Smc\_Cal\_T\* const pSmc,

EOL\_str\_avm\_intrin\* const avm\_cam\_bev ,

P\_EOL\_Param const p\_Eol\_param,

EOL\_str\_world\_model\* const world\_model)

(3) EOL\_pose\_estimator函数为优化器函数入口，主要完成相机内外参优化功能。

/\*

Function Name: EOL\_pose\_estimator

Function Function : Use LM algorithm to optimize camera inner and external params

Input :

calib\_data\_buff : Buffer for saving optimized result

pSmc : pSmc

avm\_cam\_bev : Camera intrinsic param

p\_Eol\_param : p\_Eol\_param

world\_model: Describe the relationship between vehicle coordinate system and world

coordinate system

Return : error code

\*/

EOL\_Calib\_Diag\_Status\_T EOL\_pose\_estimator(EOL\_Buffer\_Mng\_T\* calib\_data\_buff,

Smc\_Cal\_T \*pSmc,

EOL\_str\_avm\_intrin\* avm\_cam\_bev,

const P\_EOL\_Param p\_Eol\_param,

EOL\_str\_world\_model\* world\_model )

(4) EOL\_draw\_corners函数将检测到的角点及测量的世界坐标点经优化后相机参数重投影至图像中，并分别用圈和十字画在原图上，用以直观观测优化结果的准确性。

/\*

Function Name : EOL\_draw\_corners

Function Function : Draw circle onto src image's corners

Input:

rgb\_img: Src image

calib\_data\_buff : The optimized camera extrinsic params and camera center(if optimized)

pSMC: The SMC information

avm\_cam\_bev: Camera intrinsic params

world\_model: The transformation between vehicle coordinate system and world

coordinate system

pattern: Detected pattern array

pattern\_num: The valid pattern num in current camera

camid: The camera id

Return: Error code

\*/

int32\_t EOL\_draw\_corners( IplImage\* rgb\_img,

EOL\_Buffer\_Mng\_T\* calib\_data\_buff,

Smc\_Cal\_T\* pSMC,

EOL\_str\_avm\_intrin\* avm\_cam\_bev,

EOL\_str\_world\_model\* world\_model,

const chessBoard\_pattern\* const pattern,

int pattern\_num,

int camid)

(5) EOL\_draw\_world\_coordinate函数将世界坐标系中点按1pixel = 10cm的比例画至鸟瞰图中。

/\*

Function Name : EOL\_draw\_world\_coordinate

Function Function : Draw world coordinates' distribution on an image

Input:

p\_Eol\_param : p\_Eol\_param

Return : Error code

\*/

int EOL\_draw\_world\_coordinate(P\_EOL\_Param p\_Eol\_param)

### 角点检测模块重要函数说明

该小节对角点检测模块中的主要函数进行介绍。

(1) cvAdaptiveThreshold函数主要完成图像的自适应二值化，后续角点检测算法以二值化后图像为基础进行操作。

/\*

Function Name : cvAdaptiveThreshold

Function Function : Finish gray image binarylization

Input:

srcIm : source gray image

dstIm: binarylized image

maxValue: pixel upper limit

adaptive\_method: CV\_ADAPTIVE\_THRESH\_MEAN\_C

type: CV\_THRESH\_BINARY or CV\_THRESH\_BINARY\_INV

blockSize: kernel diameter

delta: Gaussian sigma

Return: Error code

\*/

void cvAdaptiveThreshold( const IplImage\* srcIm,

IplImage\* dstIm,

double maxValue,

int adaptive\_method,

int type,

int blockSize,

double delta)

(2) cvFindChessboardCorners3函数为角点检测类最为核心的函数，函数主要完成单块棋盘格角点检测功能。

/\*

Function Name: cvFindChessboardCorners3

Function Function : Detect corners corners of non 1x1 pattern

Input :

img : The input gray image

Return : Error code

\*/

int chessBoard\_corner\_detector::cvFindChessboardCorners3( const IplImage\* img )

(3) cvDilate函数按照指定核类型完成对二值图的腐蚀操作，目前支持矩形、十字形、一字型等14中核类型。

/\*

Function Name: cvDilate

Function Function : Detect corners corners of non 1x1 pattern

Input :

img : The input gray image

Return : Void

\*/

void cvDilate( const IplImage\* src, IplImage\* dst, int blockSize, int mode, int iterations)

(4) cvErode函数按照指定核类型完成对二值图的腐蚀操作，目前支持矩形、十字形、一字型等14中核类型。

/\*

Function Name: cvErode

Function Function : Detect corners corners of non 1x1 pattern

Input :

img : The input gray image

Return : Void

\*/

void cvErode ( const IplImage\* src, IplImage\* dst, int blockSize, int mode, int iterations)

(5) icvGenerateQuads函数用于检测图像中满足范围要求的四边形。

/\*

Function Name: icvGenerateQuads

Function Function : generate quad based on binary image

Input :

output\_quads : The detected quad group

out\_corners : The detected corner structure

storage : The storage will be released when the object is released

image : binary image

flags : Not used

dilation : Not used

firstRun : Whether this function is called for the first time in the dilation loop

Return : The found quad count

\*/

int chessBoard\_corner\_detector::icvGenerateQuads( CvCBQuad \*\*out\_quads,

CvCBCorner \*\*out\_corners,

CvMemStorage \*storage,

IplImage \*image,

int flags,

int dilation,

bool firstRun )

(6) mrFindQuadNeighbors2函数用于将顶点相距较近的棋盘格角点互相标记为邻居。

/\*

Function Name: mrFindQuadNeighbors2

Function Function : Label quads whose corner near with annother as neighbors

Input :

quads: The quads

quad\_count: valid quads count

dilation: dilation times

Return : Void

\*/

void chessBoard\_corner\_detector::mrFindQuadNeighbors2(CvCBQuad \*quads,

int quad\_count,

int dilation)

(7) 函数将棋盘格聚类为若干类。

/\*

Function Name: icvGenerateQuads

Function Function : generate quad based on binary image

Input :

quad : The quad group

quad\_count : Valid quads count

out\_group : The quads group

group\_idx : The current group id

storage : Make sure malloced storage will be release when object isdeconstructed

dilation : Current dilation time

Return : The found quad count

\*/

int chessBoard\_corner\_detector::icvFindConnectedQuads( CvCBQuad \*quad,

int quad\_count,

CvCBQuad \*\*out\_group,

int group\_idx,

CvMemStorage\* storage,

int dilation )

(8) icvCleanFoundConnectedQuads函数完成移除当前组中干扰方块。

/\*

Function Name: icvCleanFoundConnectedQuads

Function Function : If we found disturbed connected quads, remove them

Input :

quad\_count : Detected quad num

quad\_group : Detected quad group

pattern\_size : the size of pattern required

Return :

quand\_count : The valid quad count after outliers removing

\*/

int chessBoard\_corner\_detector::icvCleanFoundConnectedQuads( int quad\_count,

CvCBQuad \*\*quad\_group,

CvSize pattern\_size )

(9) removeInvalidQuads函数进一步移除当前组中干扰方块及干扰组。

/\*

Function Name: removeInvalidQuads

Function Function : Remove invalid pattern

Input :

quad\_group : The detected quad group

board\_size : The pattern size need to be detected

count : Valid quad num in quad\_group

Return : Void

Note : This function remove quad by the flowing criterion.

1. 0.25\*height < y center of the pattern < 0.75\*height;

2. if pattern is bigger than 2 \* 2, 0.18\*width < x center of the pattern

< 0.82\*width;

3. if pattern is 2\*2, 0.35\*width < x center of the pattern || x center of the

pattern < 0.65\*width;

4. if pattern is 2\*2, the ave x of left corners < the ave x of right corners

5. if pattern is 2\*2, the ave x of up corners < the ave y of down corners

6. the max y of up corners < the min y of down corners

7. add two 2\*2 pattern do not at the same side selection on 2017/3/21

\*/

void chessBoard\_corner\_detector::removeInvalidQuads( CvCBQuad\*\* quad\_group,

CvSize board\_size,

int& count)

(10) mrLabelQuadGroup函数给当前包含方格数目最多的组赋予标签。

/\*

Function Name: mrLabelQuadGroup

Function Function : Label corners with row and column

Input :

count : Detected quad num

quad\_group : Detected quad group

pattern\_size : The size of pattern required

firstRun : Whether this is the first time running

Return : Void

Note :

\*/

void chessBoard\_corner\_detector::mrLabelQuadGroup( CvCBQuad \*\*quad\_group,

int count,

CvSize pattern\_size,

bool firstRun )

(11) mrCopyQuadGroup函数给满足要求的棋盘格拷贝至待输出结构体。

/\*

Function Name: mrCopyQuadGroup

Function Function : Copies all necessary information of every quad of the largest found

group into a new Quad struct array.

Input :

temp\_quad\_group : Source quad group

for\_out\_quad\_group : The output quad group

count : The valid quad num

Return : Void

\*/

void chessBoard\_corner\_detector::mrCopyQuadGroup( CvCBQuad \*\*temp\_quad\_group,

CvCBQuad \*\*for\_out\_quad\_group,

int count )

(11) mrWriteCorners函数对带输出棋盘格进行检验，若合格则输出否则则清零。

/\*

Function Name: mrWriteCorners

Function Function : Write corners to file

Input :

output\_quads : The detected quad group

pattern\_size : The pattern size need to be detected

count : Valid quad num in quad\_group

min\_number\_of\_corners : The min valid corner num

Return : Whether desired corners are found

\*/

int chessBoard\_corner\_detector::mrWriteCorners( CvCBQuad \*\*output\_quads,

int count,

CvSize pattern\_size,

int min\_number\_of\_corners )

### 优化器模块主要函数说明

(1) EOL\_absolute\_estimator\_driver函数为优化器的核心函数，坐标系变换及LM算法调用均在此函数中完成。

/\*

Function Name: EOL\_absolute\_estimator\_driver

Function Function : Use ground coordinate and the detected corner coordinate to calculate

the initial value of camera extrinsic param

Input :

avm\_pose : Camera extrinsic param

camera\_center : Camera center coordinate

bev\_cam : Camera intrinsic param

p\_Eol\_param : p\_Eol\_param

world\_model: Describe the relationship between vehicle coordinate system and world

coordinate system

Return : Error code

Note :

\*/

static EOL\_Calib\_Diag\_Status\_T EOL\_absolute\_estimator\_driver(

float64\_t avm\_pose[4][6] ,

float64\_t camera\_center[4][2],

EOL\_str\_cam\_intrin \*bev\_cam[4] ,

const P\_EOL\_Param p\_Eol\_param,

const EOL\_str\_world\_model\* world\_model )

(2) EOL\_avm\_ext\_params\_model函数为LM算法的误差函数，该函数内定义了误差的评判标准。

/\*

Function Name: EOL\_avm\_ext\_params\_model

Function Function : The error function used in LM algorithm

Input :

params : The params need to be optimized

num\_errs : Valid sample point's num

data : NULL, not used

evec : to store the re-projection error of each point sample

inform : return lm status

Return : void

\*/

static void EOL\_avm\_ext\_params\_model( float64\_t \*params,

int32\_t num\_errs,

const void \*data,

float64\_t \*evec,

int32\_t \*inform )

(3) levmarqt\_minimizer 函数为LM算法的核心函数，该函数内实现了LM算法，在此函数中四相机参数进行联合优化。

/\*

Function Name: levmarqt\_minimizer

Function Function : generate quad based on binary image

Input :

n: variable dims

x : variables tobe optimized

m : sample num

data: void, not used

(\*evaluate): pointer to error judgment function

C: LM configuration information

S: LM return status

Return : LM return status

\*/

lM\_OPTIMIZOR\_STATUS\_T levmarqt\_minimizer( int32\_t n,

float64\_t \*x,

int32\_t m,

const void \*data,

void (\*evaluate),

const levmarqt\_config\_struct \*C,

levmarqt\_status\_struct \*S)

# 其他

## 配置项位置

(1) 棋盘格配置信息位置

棋盘格配置对应图1.1中的标定板设计，每种类型的标定工位在使用前均需对该项进行配置，该配置写在cvcalibinit3.h文件中，新增一组工位配置时，EOL\_init函数中亦须添加case支持并做对应修改。

// init\_x, init\_y, quad\_height, quad\_width, pattern\_height, pattern\_width, left\_boundary\_to\_axis

static EOL\_pattern\_config pattern\_config\_ofilm[4] =

{

{600, 600, 400, 400, 1600, 1600, 0}, // pattern22\_config

{300, 400, 300, 300, 1600, 900 , 0}, // pattern42\_config

{300, 400, 300, 300, 1600, 1200, 0}, // pattern43\_config

{300, 400, 300, 300, 1600, 1200, 0} // pattern44\_config

};

(2) 优化器中心优化开关

优化器是否对图像中心进行优化可通过宏OPTIMIZE\_CENTER进行控制，该宏可在eol\_pose\_estimator.h中进行开关。

(3) 地面投影误差开关

优化器中目标函数是否采用世界坐标系中投影误差作为评价函数可通过宏RE\_PROJECTION\_ERROR进行控制，该宏可在eol\_pose\_estimator.h中进行开关。

(4) 鱼眼图像角点检测结果保存

是否对角点检测结果图像进行保存可通过SAVE\_CORNER\_IMAGE宏进行配置，该宏定义在eol\_main\_driver.cpp文件中。

(5) log 保存

该选项保存角点检测是否存入log文件中可通过SAVE\_LOG宏进行配置，该宏定义在eol\_main\_driver.cpp文件中，保存路径同样定义在该文件中。

## 移植所需文件

COMM\source\utility\common\smc.h

COMM\source\utility\cv\opencv\_adapter.h

COMM\source\utility\cv\opencv\_adapter.cpp

COMM\source\utility\gpu3d\BevLUT\_OPT.cpp

## 待加入功能

(1) 拼接结果显示函数，该函数可用来直观评估优化后参数的准确性，后续考虑加入。