

Libsynexens3 SDK instructions

Version	Revised Date	Adapt SDK version	Description	Editor
v1.0	July4,2022	v0.1.2	Initial Version	Kai
v1.1	July5,2022	v0.1.3	Add interface description	Kai
v1.2	July11,2022	v0.1.3	Add ubuntu environment configuration	Kai
v1.3	February 17,2023	v0.7.3	Add some new interface	Kai

Contents

1.	Descriptions	1
2.	Windows environment configuration.....	2
2.1.	Windows environment configuration (vs2017 as example)	2
2.2.	Ubuntu environment configuration (cmake as example)	4
2.3.	Calling process diagram.....	7
3.	API Reference	7
3.1.	Global interface.....	7
3.2.	Basic interface.....	10
3.3.	SENSOR control interface	19
3.4.	Algorithm interface	24
3.5.	Date type	25
4.	Sample code	30
4.1.	Get depth frame.....	30
4.2.	Get align.....	31

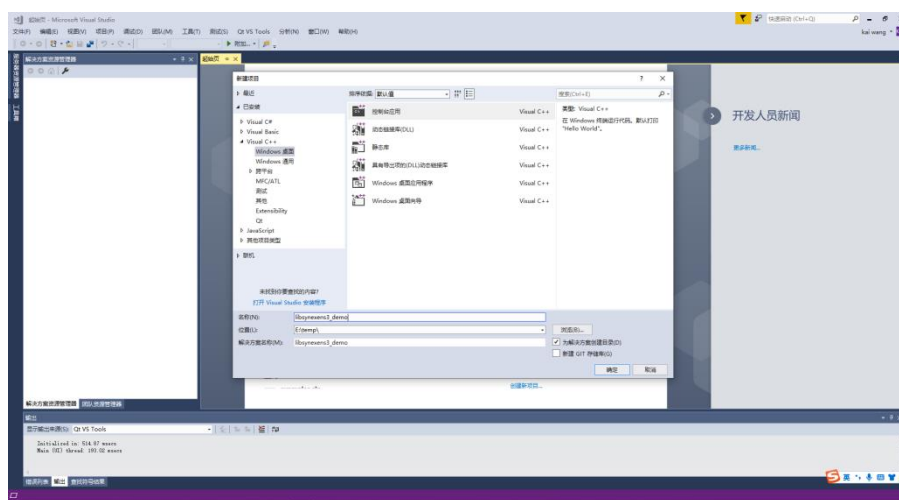
1. Descriptions

This document introduces code interfaces and demo code for users, the SDK adapt for CS series camera.

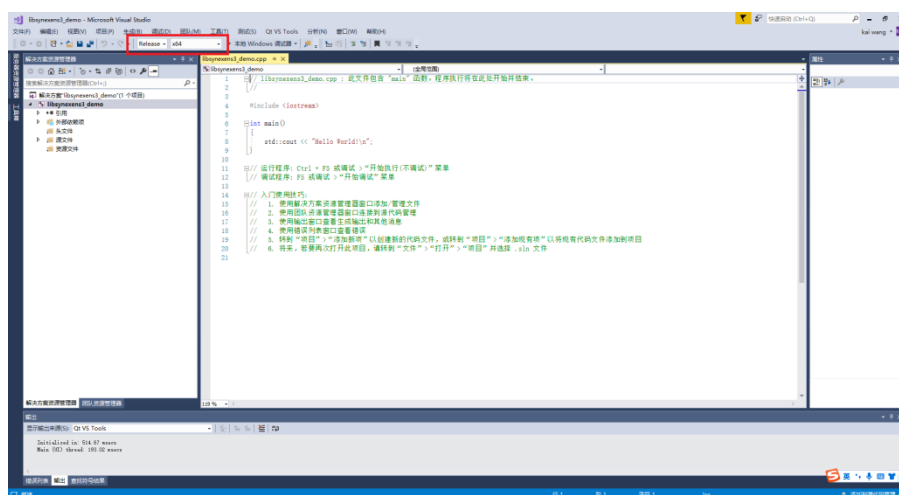
2. Windows environment configuration

2.1. Windows environment configuration (vs2017 as example)

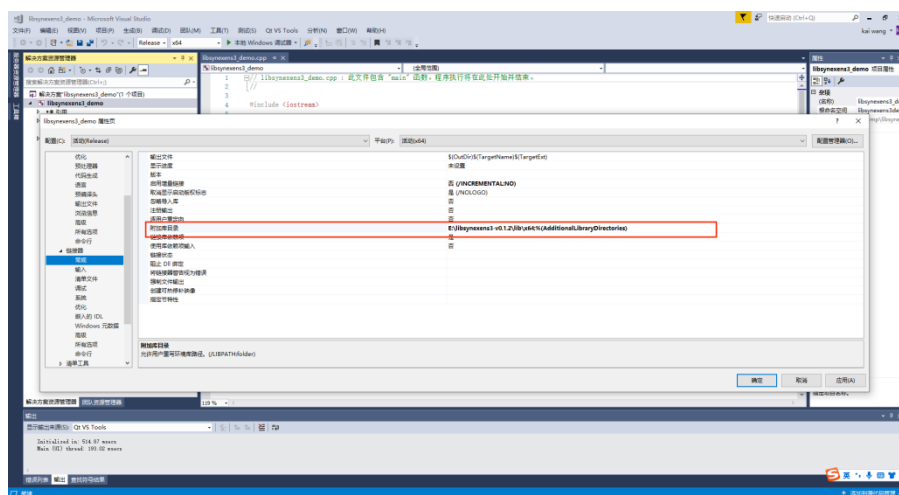
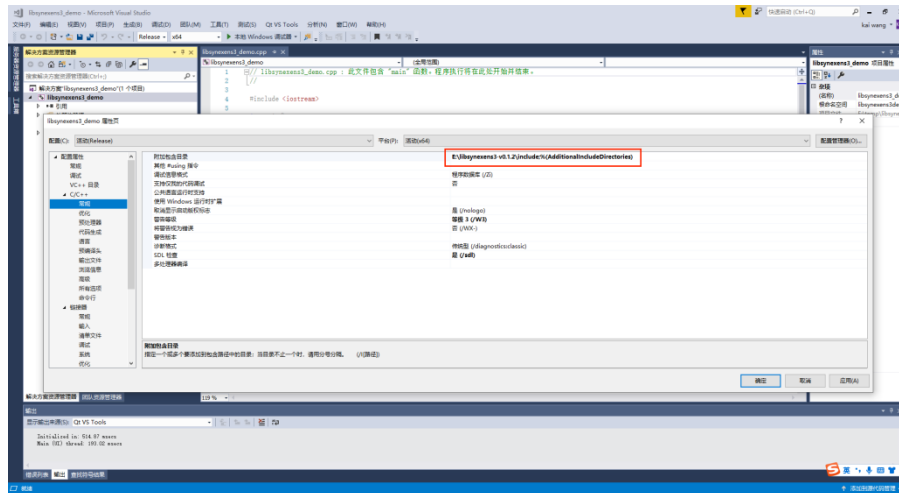
2.1.1. Create VS project.

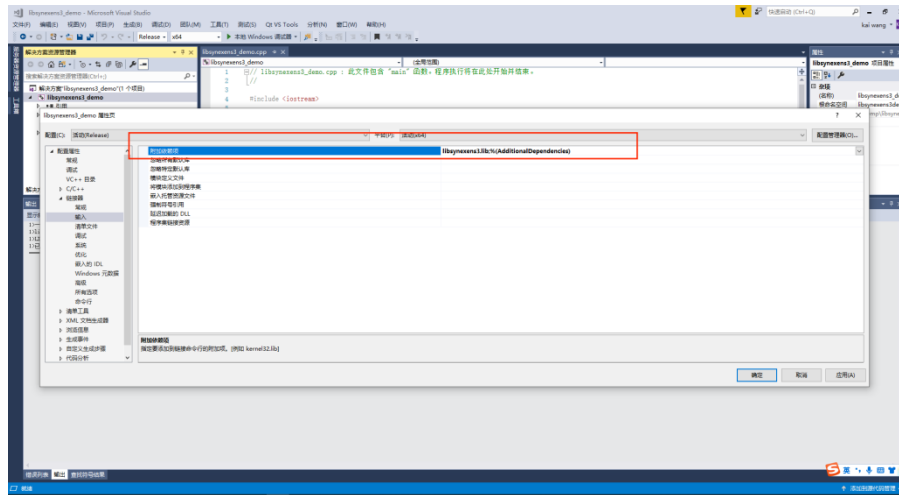


2.1.2. Select the solution and platform corresponding to the SDK library.



2.1.3. Configure the header file path and library path of the SDK in the project properties.





2.1.4. After the configuration is completed, you can enter.

2.2. Ubuntu environment configuration (cmake as example)

2.2.1. Installation Dependencies

```
sudo apt-get install libusb-1.0-0-dev
sudo apt-get install libudev-dev
```

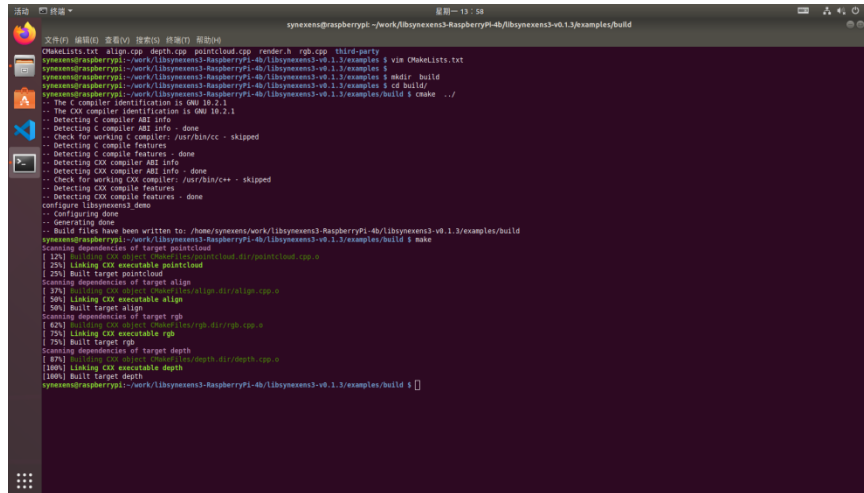
2.2.2. Compile CmakeLists.txt, the process requires familiarity with cmake syntax

```
synexens@raspberrypi: ~/work/libsynexens3-RaspberryPI-4b/libsynexens3-v0.1.3/examples
1 cmake_minimum_required(VERSION 3.4.1)
2 project("examples")
3
4 set(TARGET_NAME libsynexens3_demo)
5
6 message(STATUS "${TARGET_NAME}")
7 set_property(GLOBAL PROPERTY CXX_FLAGS -std=c++11)
8
9 include_directories(${CMAKE_CURRENT_SOURCE_DIR}/../include)
10 link_directories(${CMAKE_CURRENT_SOURCE_DIR}/../lib/armv64)
11
12 include_directories(${CMAKE_CURRENT_SOURCE_DIR}/third-party/opencv-4.4.0/include)
13 link_directories(${CMAKE_CURRENT_SOURCE_DIR}/third-party/opencv-4.4.0/lib/armv64/shared)
14
15
16 [FEATURES]
17 set(OpenCV_LIBS
18     opencv_core
19     opencv_imgproc
20     opencv_imgcodecs
21     opencv_highgui
22     opencv_video
23     opencv_calib3d)
24
25 if(UNIX)
26     set(OpenCV_LIBS
27         opencv_core
28         opencv_imgproc
29         opencv_imgcodecs
30         opencv_highgui
31         opencv_video
32         opencv_calib3d)
33     endif()
34
35 set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -std=c++11 -pthread -fopenmp")
36 set(CMAKE_SKIP_BUILD_RPATH FALSE)
37
38 add_executable(depth depth.cpp)
39 target_link_libraries(depth PUBLIC synexens3)
40 target_link_libraries(depth PRIVATE synexens3 ${OpenCV_LIBS})
41 set_property(TARGET depth PROPERTY FOLDER Samples)
42
43 add_executable(rgb rgb.cpp)
44 target_link_libraries(rgb PRIVATE synexens3 ${OpenCV_LIBS})
45 set_property(TARGET rgb PROPERTY FOLDER Samples)
46
47
48 add_executable(align align.cpp)
49 target_link_libraries(align PRIVATE synexens3 ${OpenCV_LIBS})
50 set_property(TARGET align PROPERTY FOLDER Samples)
51
52
53 add_executable(pointcloud pointcloud.cpp)
54 target_link_libraries(pointcloud PRIVATE synexens3 ${OpenCV_LIBS})
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

2.2.3. Create Compiled Project File

```
synexens@raspberrypi: ~/work/libsynexens3-RaspberryPI-4b/libsynexens3-v0.1.3/examples/build
synexens@raspberrypi:~/work/libsynexens3-RaspberryPI-4b/libsynexens3-v0.1.3/examples $ mkdir build
synexens@raspberrypi:~/work/libsynexens3-RaspberryPI-4b/libsynexens3-v0.1.3/examples $ cd build/
synexens@raspberrypi:~/work/libsynexens3-RaspberryPI-4b/libsynexens3-v0.1.3/examples/build $ cmake ../
-- The C compiler identification is GNU 10.2.1
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working C compiler: /usr/bin/cc - skipped
-- Detecting C compile features
-- Detecting C compile features - done
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Check for working CXX compiler: /usr/bin/c++ - skipped
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Configuring done
-- Generating done
-- Build files have been written to: /home/synexens/work/libsynexens3-RaspberryPI-4b/libsynexens3-v0.1.3/examples/build
synexens@raspberrypi:~/work/libsynexens3-RaspberryPI-4b/libsynexens3-v0.1.3/examples/build $
```

2.2.4. make

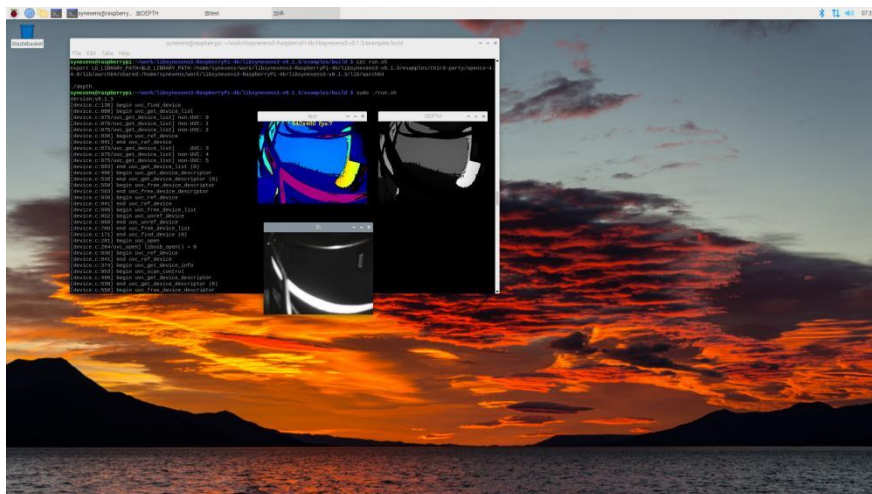


```
synexens@raspberrypi:~/work/libsynexens3-Raspberrypi-4b/libsynexens3-v0.1.3/examples/build
$ make
CMakeLists.txt align.cpp depth.cpp pointcloud.cpp render.h rgb.cpp third-party
synexens@raspberrypi:~/work/libsynexens3-Raspberrypi-4b/libsynexens3-v0.1.3/examples $ cd build/
synexens@raspberrypi:~/work/libsynexens3-Raspberrypi-4b/libsynexens3-v0.1.3/examples $ cd build/
synexens@raspberrypi:~/work/libsynexens3-Raspberrypi-4b/libsynexens3-v0.1.3/examples/build $ cmake ../
-- The C compiler identification is GNU 18.2.1
-- The CXX compiler identification is GNU 18.2.1
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working C compiler: /usr/bin/cc - skipped
-- Detecting C compile features
-- Detecting C compile features - done
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Check for working CXX compiler: /usr/bin/c++ - skipped
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Configuring done
-- Generating done
-- Build files have been written to: /home/synexens/work/libsynexens3-Raspberrypi-4b/libsynexens3-v0.1.3/examples/build
synexens@raspberrypi:~/work/libsynexens3-Raspberrypi-4b/libsynexens3-v0.1.3/examples/build $ make
Scanning dependencies of target pointcloud
[12%] Building CXX executable pointcloud
[25%] Built target pointcloud
Scanning dependencies of target align
[37%] Building CXX executable align
[50%] Built target align
Scanning dependencies of target rgb
[62%] Building CXX executable rgb
[75%] Built target rgb
Scanning dependencies of target depth
[87%] Building CXX executable depth
[100%] Built target depth
synexens@raspberrypi:~/work/libsynexens3-Raspberrypi-4b/libsynexens3-v0.1.3/examples/build $
```

2.2.5. Execute the executable file to test the effect

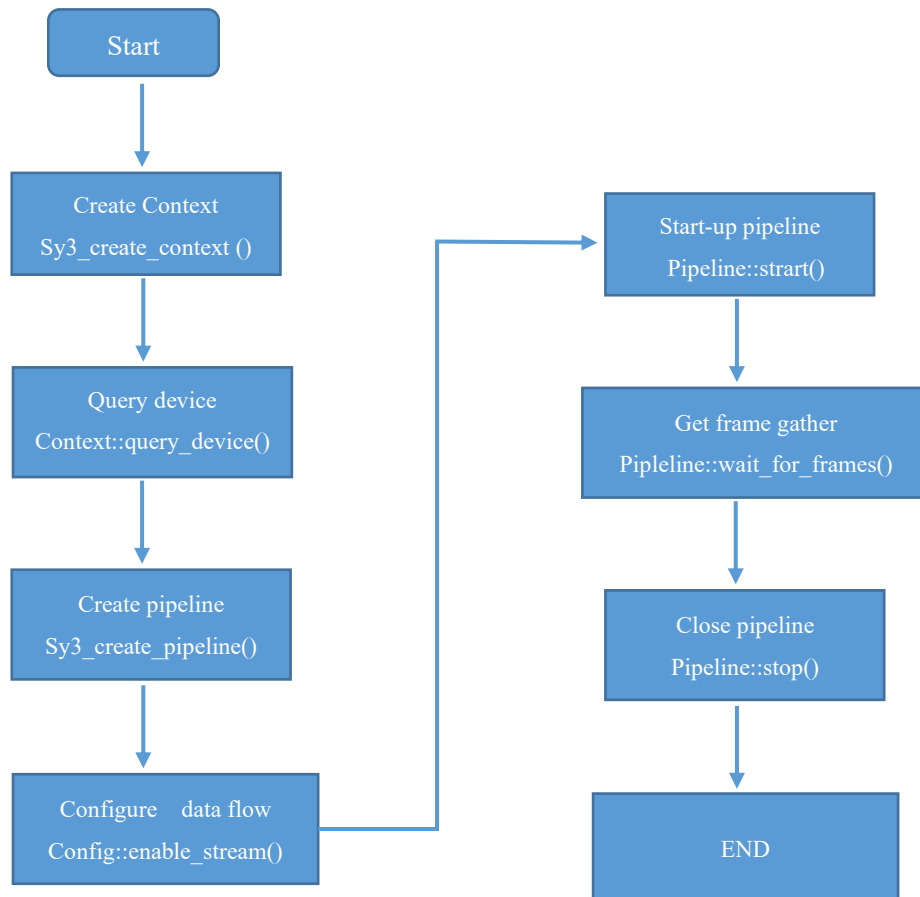
LD_LIBRARY_PATH needs to be configured before executing the program, in order to find the library files that the program depends on, the example has written run.sh script to facilitate program execution.

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:../lib/../../third-party/ubuntu18.04_x64/opencv-4.4.0/lib/x64/shared/
../depth
```



2.3. Calling process diagram

Currently, only polling mode calls are supported



3. API Reference

3.1. Global interface

3.1.1. get_device_info

Description: Obtain devices information

Grammar:

```
const device_info *get_device_info(sy3_error &error) const;
```

Parameters:

Parameters name	Descriptions	Input/Output
error	Function execution status	Output

The return value:

Return value	Description
device_info	Device information

3.1.2. query_device

Descriptions: query device

Grammar:

```
device *query_device(sy3_error &error) const;
```

Parameters:

Parameters name	Description	Input/Output
error	Function execution status	Output

The return value:

Return value	Description
device	Device pointer

3.1.3. sy3_create_pipeline

Description: Create routes

Grammar:

```
pipeline *sy3_create_pipeline (const context *ctx,sy3_error &error);
```

Parameters:

Parameters name	Description	Input/Output
ctx	Context currently in use	Input
error	Function execution status	Output

The return value:

Return value	Description
pipeline	Pipeline pointer

3.1.4. sy3_create_config

Description: Creates a configuration parameter pointer

Grammar:

```
config *sy3_create_config(sy3_error &error)
```

Parameters:

Parameters name	Description	Input/Output
error	Function execution status	Output

The return value:

Return value	Description
config	Configuration parameter pointer

3.1.5. sy3_get_device_info

Description: Get device information

Grammar:

```
const char* sy3_get_device_info(const device* device,sy3_camera_info info, sy3_error &error);
```

Parameters:

Parameters name	Description	Input/Output
device	Device pointer	Input
info	Info Enumeration	Input
error	Function execution status	Output

The return value:

Return value	Description
const char*	Corresponding device information string

Note: The call is invalid until the device is found successfully

3.2. Basic interface

3.2.1. device::get_sensor

Description: Get the sensor pointer

Grammar:

```
const sensor *get_sensor(sy3_error &error) const;
```

Parameters:

Parameters	Description	Input/Output
error	Function execution status	Output

The return value:

Return value	Description
sensor *	sensor pointer

3.2.2. device::get_type

Description: Get device type

Grammar:

```
const sy3_device_type get_type(sy3_error &error) const;
```

Parameters:

Parameters	Description	Input/Output
error	Function execution status	Output

The return value:

Return value	Description
sy3_device_type *	Device type

3.2.3. device::get_support_stream

Description: Get the list of data streams supported by the device

Grammar:

```
const std::vector<sy3_stream> get_support_stream(sy3_error &error) const
```

Parameters:

Parameters name	Description	Input/Output
error	Function execution status	Output

The return value:

Return value	Description
const std::vector<sy3_stream>	Data flow list

3.2.4. device::get_support_format

Description: Get supported data formats

Grammar:

```
const std::vector<sy3_format> get_support_format(sy3_error &error) const;
```

Parameters:

Parameters name	Description	Input/Output
error	Function execution status	Output

The return value:

Return value	Description
const std::vector<sy3_format>	Data flow list

3.2.5. device::get_support_format

Description: Get the format supported by the specified data stream of the device

Grammar:

```
const std::vector<sy3_format> get_support_format(sy3_stream stream,sy3_error &error) const;
```

Parameters:

Parameters name	Description	Input/Output
stream	Data flow type	Input
error	Function execution status	Output

The return value:

Return value	Description
const std::vector<sy3_format>	Data format list

3.2.6. config::enable_stream

Description: Enables the specified data flow

Grammar:

```
void enable_stream(sy3_stream stream, uint16_t width, uint16_t height, sy3_error &error);
```

Parameters:

Parameters name	Description	Input/Output
stream	Data type to enable	Input
width	Width of data stream image	Input
height	Height of data stream image	Input
error	Function execution status	Output

Return value: None

Note: This function is only called valid before the pipeline is started. It is invalid when called during the pipeline running.

3.2.7. config::disable_stream

Description: Disables the specified data flow

Grammar:

```
void disable_stream(sy3_stream stream, sy3_error &error);
```

Parameters:

Parameters name	Description	Input/Output
stream	Type of data flow to disable	Input
error	Function execution status	Output

Return value: None

3.2.8. config::disable_all_streams

Description: Disable all data flows

Grammar:

```
void disable_all_streams(sy3_error &error);
```

Parameters:

Parameters name	Description	Input/Output
error	Function execution status	Output

Return value: None

3.2.9. pipeline::start

Description: Start pipeline

Grammar:

```
void start(const config *cfg,sy3_error &error);
```

Parameters:

Parameters name	Description	Input/Output
cfg	Pipeline configure	Input
error	Function execution status	Output

Return value: None

3.2.10. pipeline::get_process_engin

Description: Get the algorithm instance pointer

Grammar:

```
process_engine* get_process_engin(sy3_error &error);
```

Parameters:

Parameters name	Description	Input/Output
error	Function execution status	Output

The return value:

Return value	Description
process_engine*	Algorithm instance pointer

3.2.11. pipeline::stop

Description: stop pipeline

Grammar:

```
void stop(sy3_error &error);
```

Parameters:

Parameters name	Description	Input/Output
error	Function execution status	Output

Return value: None

3.2.12.pipeline::wait_for_frames

Description: Get pipeline frame set

Grammar:

```
frameset *wait_for_frames(unsigned int timeout_ms, sy3_error &error);
```

Parameters:

Parameters name	Description	Input/Output
timeout_ms	overtime	Input
error	Function execution status	Output

The return value:

Return value	Description
frameset *	Frame gather

3.2.13.pipeline::get_device

Description: Get the current device

Grammar:

```
const device *get_device(sy3_error &error);
```

Parameters:

Parameters name	Description	Input/Output
error	Function execution status	Output

The return value:

Return value	Description
const device *	Device pointer

3.2.14.frameset::get_depth_frame

Description: Get depth frame

Grammar:

```
depth_frame *get_depth_frame();
```

Parameters: none

The return values

Return value	Description
--------------	-------------

depth_frame *	Depth frame data pointer
---------------	--------------------------

3.2.15.frameset::get_ir_frame

Description: Get ir frame

Grammar:

```
ir_frame *get_ir_frame();
```

Parameters: none

The return values

Return value	Description
ir_frame *	Ir frame data pointer

3.2.16.frameset::get_raw_frame

Description: Get device information

Grammar:

```
raw_frame *get_raw_frame();
```

Parameters: none

The return value

Return value	Description
raw_frame *	raw frame data pointer

Note: At present, raw_ Frame data is only open to internal calibration, and external users cannot obtain raw data

3.2.17.frame::get_width

Description: Get the width of image

Grammar:

```
const int get_width();
```

Parameters: none

The return value:

Return value	Description
const int	Width

3.2.18.frame::get_height

Description: Get the height of image

Grammar:

```
const int get_height() ;
```

Parameters: none

The return value:

Return value	Description
const int	Height

3.2.19.frame::get_type

Description: Get the type of frame

Grammar:

```
const sy3_stream get_type() ;
```

Parameters: none

The return value:

Return value	Description
const sy3_stream	Frame type

3.2.20.depth_frame::get_data

Description: Obtain the frame image array, store the data in the form of uint16 type two-dimensional array (uint16 [h] [w]), and return the array pointer to the caller as the return value. The specific data format is as follows:

depth[0][0]	depth[0][1]	...	depth[0][width-2]	depth[0][width-1]
depth[1][0]	depth[1][1]	...	depth[1][width-2]	depth[1][width-1]
...				
...				
depth[height-1][0]	depth[height-1][1]	...	depth[height-1][width-2]	depth[height-1][width-1]

Grammar:

```
void *get_data();
```

Parameters: none

The return value:

Return value	Description
--------------	-------------

void *	Frame data pointer
--------	--------------------

Note: The return value type is void *, and you need to manually convert it to uint16*

3.2.21.depth_frame::apply_colormap

Description: Obtain the data mapped rgb (BGR format), store the data in the form of uint8 type 3D array (uint8 [h] [w] [3]), and return the array pointer to the caller as the return value. The specific data format is as follows:

rgb[0][0][(b)0]	rgb[0][0][(g)1]	rgb[0][0][(r)2]			...	rgb[0][w-1][0]	rgb[0][w-1][1]	rgb[0][w-1][2]
rgb[1][0][0]	rgb[1][0][1]	rgb[1][0][2]			...	rgb[1][w-1][0]	rgb[1][w-1][1]	rgb[1][w-1][2]
...								
rgb[h-1][0][0]	rgb[h-1][0][1]	rgb[h-1][0][2]			...	rgb[h-1][w-1][0]	rgb[h-1][w-1][1]	rgb[h-1][w-1][2]

Grammar:

```
uint8_t *apply_colormap();
```

Parameters: none

The return value:

Return value	Description
uint8_t *	rgb image numeric pointer

3.2.22.frame::get_profile

Description: Get frame configure

Grammar:

```
const stream_profile *get_profile() const;
```

Parameters: none

The return values

Return value	Description
const stream_profile *	Frame configuration pointer

3.2.23.frame::dump

Description: Save frames locally for debugging

Grammar:

```
int dump(const char *filenam) ;
```

Parameters:

Parameters name	Description	Input/Output
filenam	File name to save	Input

The return value:

Return value	Description
int	execution result

Note: This function will not automatically create folders. If there are folders in the path that have not been created, please create them manually.

3.2.24.points::get_points

Description: Obtain 3D point cloud data, which is stored in the form of float type 3D array float [h] [w] [3]. The array pointer is returned to the caller as a return value. The specific data format is as follows:

points[0][0][(x)0]	points[0][0][(y)1]	points[0][0][(z)2]	...	points[0][w-1][0]	points[0][w-1][1]	points[0][w-1][2]
points [1][0][0]	points[1][0][1]	points[1][0][2]	...	points[1][w-1][0]	points[1][w-1][1]	points[1][w-1][2]
points [h-1][0][0]	points[h-1][0][1]	points[h-1][0][2]	...	points[h-1][w-1][0]	points[h-1][w-1][1]	points[h-1][w-1][2]

Grammar:

```
float *get_points();
```

Parameters: none

The return value:

Return value	Description
float *	Point cloud array pointer

3.2.25.points::get_length

Description: Get point cloud length

Grammar:

```
int get_length();
```

Parameters:

Parameters name	Description	Input/Output
-----------------	-------------	--------------

error	Function execution status	Output
-------	---------------------------	--------

The return value:

Return value	Description
int	Point cloud length

3.3. SENSOR control interface

3.3.1. sensor::set_option

Description: configure sensor function attribute value

Grammar:

```
int set_option(sy3_option option, uint16_t value, sy3_error &error);
```

Parameters:

Parameters name	Description	Input/Output
option	Function Item Enumeration	Input
value	Set value	Input
error	Execution status	Output

The return value:

Return value	Description
int	whether or not success

3.3.2. sensor::set_option

Description: Configure the extreme value of the sensor function item

Grammar:

```
int set_option(sy3_option option, uint16_t max, uint16_t min, sy3_error &error);
```

Parameters:

Parameters name	Description	Input/Output
option	Function Item Enumeration	Input
max	Max	Input
min	Min	Input
error	Function execution status	Output

The return value:

Return value	Description
--------------	-------------

int	whether or not success
-----	------------------------

3.3.3.sensor::get_option

Description: Get the extreme value range of sensor function attribute

Grammar:

```
int get_option(sy3_option option, uint16_t &max, uint16_t &min,sy3_error &error);
```

Parameters:

Parameters name	Description	Input/Output
option	Function Item Enumeration	Input
max	Max	Input
min	Min	Input
error	Function execution status	Output

The return value:

Return value	Description
int	whether or not success

3.3.4.sensor::get_option

Description: Get the extreme value of sensor function attribute

Grammar:

```
int get_option(sy3_option option, uint16_t &value,sy3_error &error);
```

Parameters:

Parameters name	Description	Input/Output
option	Function Item Enumeration	Input
value	Attribute Value	Input
error	Function execution status	Output

The return value:

Return value	Description
int	whether or not success

3.3.5.sensor::set_filter_value

Description: filter function

Grammar:

```
set_filter_value(FilterType filter_type, FILTER_THRESHOLD threshold_value, int num)
```

Parameters:

Parameters name	Description	Input/Output
filter_type	Filter type	Input
threshold_value	Filter type	Input
num	Filter parameters length	Input

3.3.6.sensor::get_filter_value

Description: filter function

Grammar:

```
set_filter_value(FilterType filter_type, FILTER_THRESHOLD threshold_value, int num)
```

Parameters:

Parameters name	Description	Input/Output
filter_type	Filter type	
threshold_value	Filter type	
num	Filter parameters length	

Filter function parameters setting description:

Sample: Amplitude filter AMPLITITUD

```
FILTER_THRESHOLD threshold_value{ 0 };
threshold_value[0] = 10; // amplitud_threshold
set_filter_value(FilterType::AMPLITITUD, threshold_value, 1);
int num = 0;
get_filter_value(FilterType::AMPLITITUD, threshold_value, num);
std::cout << "set amplitud_threshold = " << threshold_value[0] << std::endl;
```

Sample: Median filter MEDIAN

```
FILTER_THRESHOLD threshold_value{ 0 };
threshold_value[0] = 3; // median_ksize
threshold_value[1] = 1; // median_iterations
set_filter_value(FilterType::MEDIAN, threshold_value, 2);
int num = 0;
get_filter_value(FilterType::MEDIAN, threshold_value, num);
std::cout << "set median_ksize = " << threshold_value[0] << "    median_iterations = " <<
threshold_value[1] << std::endl;
```

Sample: Gauss filter GAUSS

```
FILTER_THRESHOLD threshold_value{ 0 };
threshold_value[0] = 3;// median_ksize
threshold_value[1] = 1;// median_iterations
set_filter_value(FilterType::GAUSS, threshold_value, 2);
int num = 0;
get_filter_value(FilterType::GAUSS, threshold_value, num);
std::cout << "set gauss_ksize = " << threshold_value[0] << "    gauss_iterations = " <<
threshold_value[1] << std::endl;
```

Sample: Edge filter EDGE

```
sy3::FILTER_THRESHOLD threshold_value{ 0 };
threshold_value[0] = 50;//edge_threshold
set_filter_value(sy3::FilterType::EDGE, threshold_value, 1);
int num = 0;
get_filter_value(sy3::FilterType::EDGE, threshold_value, num);
std::cout << "set edge_threshold = " << threshold_value[0] << std::endl;
```

Sample: Speckle filter SPECKLE

```
// speckle_max_diff threshold_value[0] = 40; speckle_size threshold_value[1] = 100
sy3::FILTER_THRESHOLD threshold_value{ 0 };
threshold_value[0] = 40;// speckle_size
threshold_value[1] = 100;// speckle_max_diff
set_filter_value(sy3::FilterType::SPECKLE, threshold_value, 2);
int num = 0;
get_filter_value(sy3::FilterType::SPECKLE, threshold_value, num);
std::cout << "Set speckle_size = " << threshold_value[0] << "    speckle_max_diff = " <<
threshold_value[1] << std::endl;
```

Sample: Sobel filter SOBEL

```
sy3::FILTER_THRESHOLD threshold_value{ 0 };
threshold_value[0] = 150;// sobel_threshold
set_filter_value(sy3::FilterType::SOBEL, threshold_value, 1);
int num = 0;
get_filter_value(sy3::FilterType::SOBEL, threshold_value, num);
std::cout << "Set sobel_threshold = " << threshold_value[0] << std::endl;
```

Sample: Edge filter 2 EDGE_MAD

```
sy3::FILTER_THRESHOLD threshold_value{ 0 };
threshold_value[0] = 15;// EDGE_MAD_threshold
set_filter_value(sy3::FilterType::EDGE_MAD, threshold_value, 1);
int num = 0;
get_filter_value(sy3::FilterType::EDGE_MAD, threshold_value, num);
```



```
std::cout << "set edge_mad_threshold  = " << threshold_value[0] << std::endl;
```

Sample: Okada filter OKADA

```
sy3::FILTER_THRESHOLD threshold_value{ 0 };
threshold_value[0] = 15;// EDGE_MAD_threshold
set_filter_value(sy3::FilterType::OKADA, threshold_value, 1);
int num = 0;
get_filter_value(sy3::FilterType::OKADA, threshold_value, num);
std::cout << "set okada_diff  = " << threshold_value[0] << std::endl;
```

Parameters range table:

Parameters interface	Parameters1 Min	Parameters1 Max	Parameters1 recommended value	Parameters2 Min	Parameters2 Max	Parameters2 recommended value
AMPLITUD	0	100	6			
MEDIAN	3	5	3	0	5	1
EDGE	20	200	50			
SPECKLE	24	200	40	40	200	
GAUSS	3	5	3	0	5	1
EDGE_MAD	5	100	15			
SOBEL	20	300	150			
OKADA	10	100	10			

Filter call sequence:

CS20: Median , Edge, Speckle, Median
DepthFilter(depth, FilterType::MEDIAN);
DepthFilter(depth, FilterType::EDGE);
DepthFilter(depth, FilterType::MEDIAN);

CS30: SOC run Median, Edge, Median
DepthFilter(depth, FilterType::MEDIAN);
DepthFilter(depth, FilterType::EDGE);
DepthFilter(depth, FilterType::MEDIAN);
Call Speckle , Median again in other platforms
DepthFilter(depth, FilterType::SPECKLE);
DepthFilter(depth, FilterType::MEDIAN);

3.4. Algorithm interface

3.4.1. process_engine::compute_points

Description: Compute point clouds

Grammar:

```
points *compute_points(depth_frame *depth,sy3_error &error) const;
```

Parameters:

Parameters name	Description	Input/Output
depth	Depth frame to be converted to point cloud	Input
error	Function execution status	Output

The return value:

Return value	Description
points	Points cloud pointer

Note: The function has applied for memory for the point cloud internally. If the point cloud data is no longer needed, it needs to be released manually, that is, call delete points

3.4.2. process_engine::align_to_rgb

Description: rgbd alignment

Grammar:

```
sy3::frameset *align_to_rgb(depth_frame *depth,rgb_frame *rgb,sy3_error &error);
```

Parameters:

Parameters name	Description	Input/Output
depth	Depth pointer	Input
rgb	rgb pointer	Input
error	Function execution status	Output

The return values:

Return value	Description
frameset *	Aligned rgb and depth sets

Note: Currently, only mapping from rgb resolution 1920x1080 to depth resolution 640x480 is supported. The align rgb resolution and align depth resolution of the mapping output are 1920x1080 and 1920x1080 respectively.

3.5. Date type

3.5.1. sy3_error

```
enum sy3_error
{
    SUCCESS = 0,
    INVALID_PID,
    INVALID_VID,
    DEVICE_NOT_FOUND,
    INVALID_FORMAT,
    INCONSISTENCY_RES,
    OPEN_FAILED,
    NOT_IMPLEMENTED,
    INVALID_INSTANCE,
}sy3_error;
```

Parameters:

Parameters name

SUCCESS
INVALID_PID
INVALID_VID
DEVICE_NOT_FOUND
INVALID_FORMAT
INCONSISTENCY_RES
OPEN_FAILED
NOT_IMPLEMENTED
INVALID_INSTANCE

3.5.2. sy3_device_type

```
enum sy3_device_type
{
    DEVICE_CS30,
    DEVICE_CS20,
} sy3_device_type;
```

Parameters:

Parameter description
DEVICE_CS30
DEVICE_CS20

3.5.3. sy3_camera_info

```
enum sy3_camera_info
{
    SY3_CAMERA_INFO_NAME,
    SY3_CAMERA_INFO_SERIAL_NUMBER,
    SY3_CAMERA_INFO_FIRMWARE_VERSION,
    SY3_CAMERA_INFO_RECOMMENDED_FIRMWARE_VERSION,
    SY3_CAMERA_INFO_PRODUCT_ID,
    SY3_CAMERA_INFO_COUNT
} sy3_camera_info;
```

Parameters:

Parameters description
SY3_CAMERA_INFO_NAME
SY3_CAMERA_INFO_SERIAL_NUMBER

SY3_CAMERA_INFO_FIRMWARE_VERSION
SY3_CAMERA_INFO_RECOMMENDED_FIRMWARE_VERSION
SY3_CAMERA_INFO_PRODUCT_ID

3.5.4. sy3_stream

```
enum sy3_stream
{
    SY3_STREAM_NONE,
    SY3_STREAM_DEPTH=2,
    SY3_STREAM_RGB,
    SY3_STREAM_IR,
    SY3_STREAM_COUNT,
} sy3_stream;
```

Parameters:

Parameters description
SY3_STREAM_NONE
SY3_STREAM_DEPTH
SY3_STREAM_RGB
SY3_STREAM_IR

3.5.5. sy3_format

```
struct sy3_format {
    sy3_stream stream;
    int width;
    int height;
} sy3_format;
```

Parameters:

Parameters description
stream
width
height

3.5.6. intrinsics

```
struct sy3_intrinsics{
```

```

    int width;
    int height;
    float ppx;
    float ppy;
    float fx;
    float fy;
    float coeffs[5];
} sy3_intrinsics;

```

Parameters:

Parameters description
width
height
ppx
ppy
fx
fy
coeffs[5]

3.5.7. sy3_option

```

typedef enum sy3_option{
    SY3_OPTION_EXPOSURE,
    SY3_OPTION_EXPOSURE_RANGE,
    SY3_OPTION_DISTANCE_RANGE,
    SY3_OPTION_DEFAULT_DISTANCE_RANGE,
    SY3_OPTION_RGB_IMAGE_FLIP,
    SY3_OPTION_RGB_IMAGE_MIRROR,
    SY3_OPTION_TOF_IMAGE_FLIP,
    SY3_OPTION_TOF_IMAGE_MIRROR,
    SY3_OPTION_DEPTH_IMAGE_MIRROR,
    SY3_OPTION_DEPTH_IMAGE_FILTER,
    SY3_OPTION_COUNT,
} sy3_option;

```

Description:

Parameters description	Description
SY3_OPTION_EXPOSURE	Exposure time; unit: us
SY3_OPTION_EXPOSURE_RANGE	Exposure time range
SY3_OPTION_DISTANCE_RANGE	Displayed distance range
SY3_OPTION_DEFAULT_DISTANCE_RANGE	Default distance range
SY3_OPTION_RGB_IMAGE_FLIP	Rgb flip
SY3_OPTION_RGB_IMAGE_MIRROR	Rgb mirror

SY3_OPTION_TOF_IMAGE_FLIP	Tof flip
SY3_OPTION_TOF_IMAGE_MIRROR	Tof mirror
SY3_OPTION_DEPTH_IMAGE_MIRROR	depth mirror
SY3_OPTION_DEPTH_IMAGE_FILTER	depth filter

4. Sample code

4.1. Get depth frame

```
//仅截取关键代码, 详细代码请参阅 samples 源码
#include "libsynexens3/libsynexens3.h"
int main(int argc, char **argv)
{
    sy3::sy3_error e;
    printf("version:%s \n", sy3::sy3_get_version(e));
    sy3::context *ctx = sy3::sy3_create_context(e);
    sy3::device *dev = ctx->query_device(e);
    if (e != sy3::sy3_error::SUCCESS) {
        printf("error:%s \n", sy3::sy3_error_to_string(e));
        return 0;
    }
    sy3::pipeline *pline = sy3::sy3_create_pipeline(ctx, e);
    sy3::config *cfg = sy3_create_config(e);
    cfg->enable_stream(sy3::sy3_stream::SY3_STREAM_DEPTH, 640, 480, e);
    pline->start(cfg, e);
    bool quit = false;
    while (!quit)
    {
        sy3::frameset *frameset = pline->wait_for_frames(SY3_DEFAULT_TIMEOUT, e);
        sy3::depth_frame *depth_frame = frameset->get_depth_frame();
        show_depth_frame(depth_frame);
        delete frameset;
        if (cv::waitKey(1) == 'q') {
            pline->stop(e);
            quit = true;
        }
    }
    system("pause");
    return 0;
}
```


4.2. Get align

```
//仅截取关键代码, 详细代码请参阅 samples 源码
#include "libsynexens3/libsynexens3.h"
int main(int argc, char **argv)
{
    sy3::sy3_error e;
    sy3::context *ctx = sy3::sy3_create_context(e);
    sy3::device *dev = ctx->query_device(e);
    if (e != sy3::sy3_error::SUCCESS) {
        return 0;
    }
    sy3::pipeline *pline = sy3::sy3_create_pipeline(ctx, e);
    sy3::config *cfg = sy3_create_config(e);
    cfg->enable_stream(sy3::sy3_stream::SY3_STREAM_DEPTH, 640, 480, e);
    cfg->enable_stream(sy3::sy3_stream::SY3_STREAM_RGB, 1920, 1080, e);
    pline->start(cfg, e);
    bool quit = false;
    while (!quit)
    {
        switch (cv::waitKey(1)) {
            case 'q': {
                pline->stop(e); quit = true;
            }break;
            case 'e': {
                //设置曝光时间, 单位 us
                dev->get_sensor(e)->set_option(sy3::sy3_option::SY3_OPTION_EXPOSURE, 10,
                e);
            }break;
            default: break;
        }
        sy3::frameset *frameset = pline->wait_for_frames(SY3_DEFAULT_TIMEOUT, e);
        sy3::depth_frame *depth_frame = frameset->get_depth_frame();
        sy3::rgb_frame *rgb_frame = frameset->get_rgb_frame();
        sy3::process_engine *engine = pline->get_process_engin(e)
        sy3::frameset *align_set=engine->align_to_rgb(depth_frame, rgb_frame, e);
        show_depth_frame(align_set->get_depth_frame(), "algin_depth");
        show_rgb_rgb_frame(align_set->get_rgb_frame(), "algin_rgb");
        delete frameset;
        delete align_set;
    }
    return 0;}

```