```
// basic include
#include <iostream>
#include <stdlib.h>
#include <string>
#include <sstream>
#include <math.h>
#include <vector>
#include <list>
#include <algorithm>
#include <time.h>

// programm includes
//#include "global_variables.h"
#include "programm_verbosity.h"
#include "user_interaction.h"
#include "time_stamp_manager.h"
#include "list_of_pixel_positions.h"
#include "pointing_direction.h"
#include "star_in_image.h"
#include "intrinsic_camera_parameter.h"
#include "simple_image.h"
#include "sccan_image.h"
#include "ueye_camera.h"
#include "sccan_analysis_point.h"
#include "sccan_point_pair.h"
#include "mirror.h"
#include "reflector.h"
#include "snapshot.h"
#include "sccan_point_pair_handler.h"
#include "quick_align.h"
#include "sccan_point_analysis.h"
#include "verbosity_handler.h"
#include "main_menu.h"
#include "star_recognition_test_environment.h"
```

open CV


```
//========================================================================
int main(){
    int system_call_return_value;
    system_call_return_value = system("clear");
    system_call_return_value = system("clear");

    //========================================================================
    // star camera
    //========================================================================

        //========================================================================
        // intrinsic paramters for star camera
```

```
    //=================================================================
intrinsic_camera_parameter parameters_for_star_camera;

/*
parameters_for_star_camera.set_names(
"ueye 5MPx CMOS",
"Carl Zeiss Flektogon F2.4 / f35mm");

parameters_for_star_camera.set_FoV_to_pixel_mapping(3.34375E-3);
*/
parameters_for_star_camera.set_names(
"ueye 5MPx CMOS","Flektogon F1.8 / f50mm");

parameters_for_star_camera.set_FoV_to_pixel_mapping(0.002427534);

parameters_for_star_camera.
set_coefficients_for_radiometric_correction_plane(
-1.2185,
1.2021,
0.99303
);

ueye_camera star_camera(13,parameters_for_star_camera);
//=====================================================================
// reflector camera
//=====================================================================

    //=================================================================
    // intrinsic paramters for reflector camera
    //=================================================================
intrinsic_camera_parameter parameters_for_reflector_camera;

parameters_for_reflector_camera.set_names(
"Thor Labs 1.3MPx CCD",
"M12 the imageing source F2.0 / f4mm"
);

parameters_for_reflector_camera.
set_coefficients_for_radiometric_correction_plane(
-1.1527,
1.0283,
-0.18637
);

ueye_camera reflector_camera(42,parameters_for_reflector_camera);


star_camera.display_camera_information();
reflector_camera.display_camera_information();
//=====================================================================
// hanldes
//=====================================================================

sccan_point_pair_handler sccan_handle;
sccan_handle.set_cameras(&star_camera,&reflector_camera);

//sccan_handle.acquire_sccan_points(5);
```

```
    snapshot snap;
    snap.add_camera(&star_camera);
    snap.add_camera(&reflector_camera);

    reflector reflector_instance(&reflector_camera);
    quick_align quick(&reflector_instance,&sccan_handle);

    //tester
    star_recognition_test_environment test_environment;

    sccan_point_analysis analysis(
    &sccan_handle,&reflector_instance//,&star_camera
    );

    verbosity_handler verbosity_interaction(
    &global_time_stamp_manager_instance,
    &star_camera,
    &reflector_camera,
    &reflector_instance,
    &snap,
    &sccan_handle,
    &quick,
    &analysis
    );

    main_menu menu(
    &snap,
    &reflector_instance,
    &sccan_handle,
    &quick,
    &analysis,
    &verbosity_interaction,
    &star_camera,
    &reflector_camera,
    &test_environment);

    menu.interaction();

    return 0;
}
```

UEYE camera

```
//==================================
// include guard
#ifndef __UEYE_CAMERA_H_INCLUDED__
#define __UEYE_CAMERA_H_INCLUDED__

//==================================
// forward declared dependencies

//==================================
// included dependencies
#include "programm_verbosity.h"
#include "intrinsic_camera_parameter.h"
#include "sccan_image.h"
#include <cmath>
//============================================================================
```

UEYE camera by IDS imaging.
Manual:

```
#include <ueye.h>
//============================================================================

//==================================
class ueye_camera                :public programm_verbosity{
private:
    sccan_image latest_image;
    HIDS        ueye_camera_handle;
    CAMINFO     ueye_camera_info;
    SENSORINFO  ueye_sensor_info;
    int         ueye_camera_id;
    int         ueye_camera_sensor_number_of_pixels_in_width;
    int         ueye_camera_sensor_number_of_pixels_in_hight;
    int         ueye_color_mode;
    int         ueye_number_of_coulor_channels;
    int         ueye_bits_per_coulor_channel;
    int         ueye_bits_per_pixel;
    double      ueye_exposure_time_in_ms;
    uint        ueye_pixel_clock_min_in_MHz;
    uint        ueye_pixel_clock_max_in_MHz;
    uint        ueye_pixel_clock_increment_in_MHz;
    uint        ueye_current_pixel_clock_in_MHz;
    uint        ueye_default_pixel_clock_in_MHz;
    double      ueye_current_framerate_in_fps;
    double      ueye_default_framerate_in_fps;
    bool        initialization_succesfull;
    bool        flag_long_time_exposure;
    std::stringstream out;

    //intrinsic_camera_parameter
    intrinsic_camera_parameter intrinsic;
public:
//============================================================================
ueye_camera(int camera_ID_to_initialize,
intrinsic_camera_parameter new_intrinsic);
//============================================================================
void set_camera_ID(int camera_ID_to_initialize);
//============================================================================
uint get_camera_ID();
//============================================================================
~ueye_camera();
//============================================================================
bool initialize();
//============================================================================
bool acquire_image(double *pointer_to_desired_exposure_time_in_ms);
//============================================================================
bool acquire_image(double *pointer_to_desired_exposure_time_in_ms,
double desired_relative_maximal_camera_response);
//============================================================================
bool long_time_exposure(bool long_time_exposure);
//============================================================================
```

```
void display_camera_information();
//=========================================================================
sccan_image get_latest_image();
//=========================================================================
void export_latest_image(std::string filename_prefix);
//=========================================================================
double get_current_exposure_time_in_ms();
//=========================================================================
void disp_latest_image();
//=========================================================================
bool camera_status();
//=========================================================================
bool is_initialized();
//=========================================================================
void toggle_verbosity();
//=========================================================================
cv::Size get_sensor_size()const;
//=========================================================================
};

#endif // __UEYE_CAMERA_H_INCLUDED__
```