Camera Calibration and 3D information

Goals

- Learning how to calibrate a camera using OpenCV.
- Learning how to use calibration to augment reality.

Minimum requirements (6 points)

1. Using the calibration opencv calibration program, calibrate the camera that recorded the video file provided.
2. Using the file with the camera parameters, create a program to augment reality. The idea is to create a program that reads the video recorded with the calibrated camera and superimpose information on it.

For each video frame, the program should proceed as follows:

1. Detect the board using cv::findChessboardCorners, and refine the corners with cv::cornerSubPix.
2. Estimate the camera pose with respect to the board using cv::solvePnP.
3. Drawing on the image a simple 3D scene. The 3D axis placed in the center of the reference system in colors red (X axis), green (Y axis) and blue (Z axis). The size of the axis should be the same as the board squares. You can use the function cv::projectPoints and cv::line for that.

The program should be used as:

```
augReal rows cols size intrisics.yml <input video-file|cam-idx>
```

The input parameters are the intrinsic parameters stored in the .yml file generated with the previous program, the board geometry (row col size), and a video sequence.

Optional requirements (4 points)

Show an image or video over the board. In order to map an image over another you must use cv::getPerspectiveTransform which obtains the mapping matrix and then cv::warpPerspective to map the virtual image over the real one using the previous matrix.

This will add the command line parameter
`[-i <img|video>]` -i for overlaying another image or video to the program

Example reading video

```
#include <opencv2/highgui.hpp>
#include <iostream>
int main(int argc,char **argv){
   std::string pathToVideo=argv[1];
   cv::VideoCapture video(pathToVideo);

   cv::Mat image;
   while(video.grab()){

        video.retrieve(image);
        cv::imshow("image",image);
        cv::waitKey(10);
   }

}
```