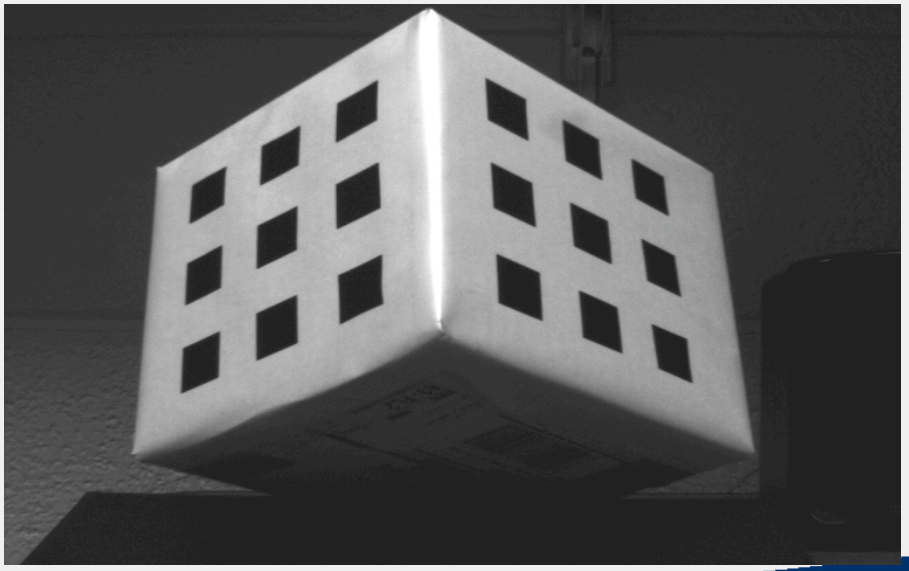
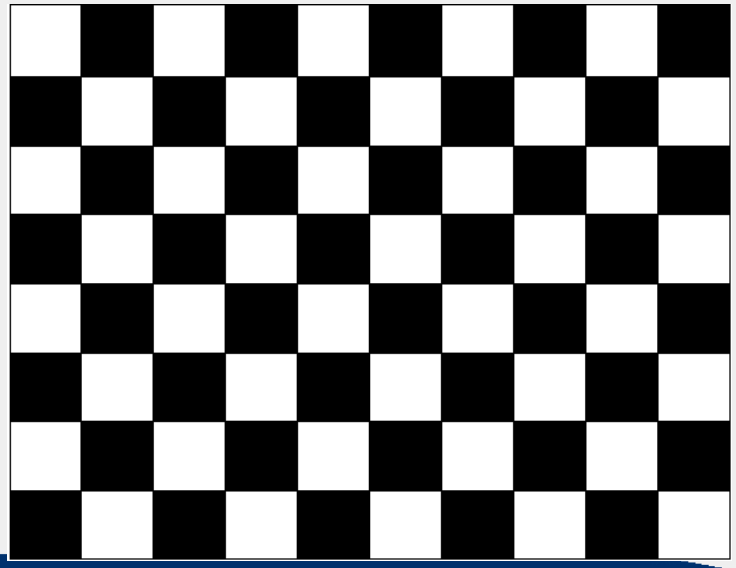
Camera Calibration - Homography

Calibration Object

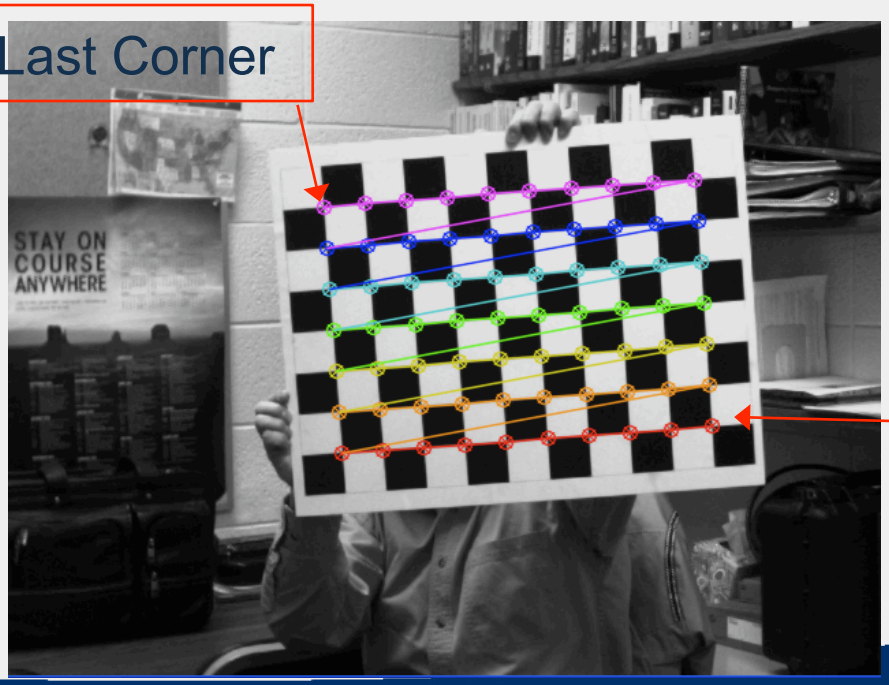
- It is difficult to make a good 3-D calibration object.
- Direct or indirect method is sensitive to noise or corner inaccuracy.
- Optical center is usually not very accurate.
- OpenCV uses multiple views of a planar object rather than one view of a 3-D object (J. Y. Bouguet : www.vision.caltech.edu/bouguetj/calib_doc).





Find Chessboard Corners

bool findChessboardCorners(const Mat& image, Size patternSize, vector<Point2f>& corners, int flags=CV_CALIB_CB_ADAPTIVE_THRESH +CV_CALIB_CB_NORMALIZE IMAGE);



Corners are detected and recorded starting from the lower right corner of the chessboard

Find Subpixel Corners

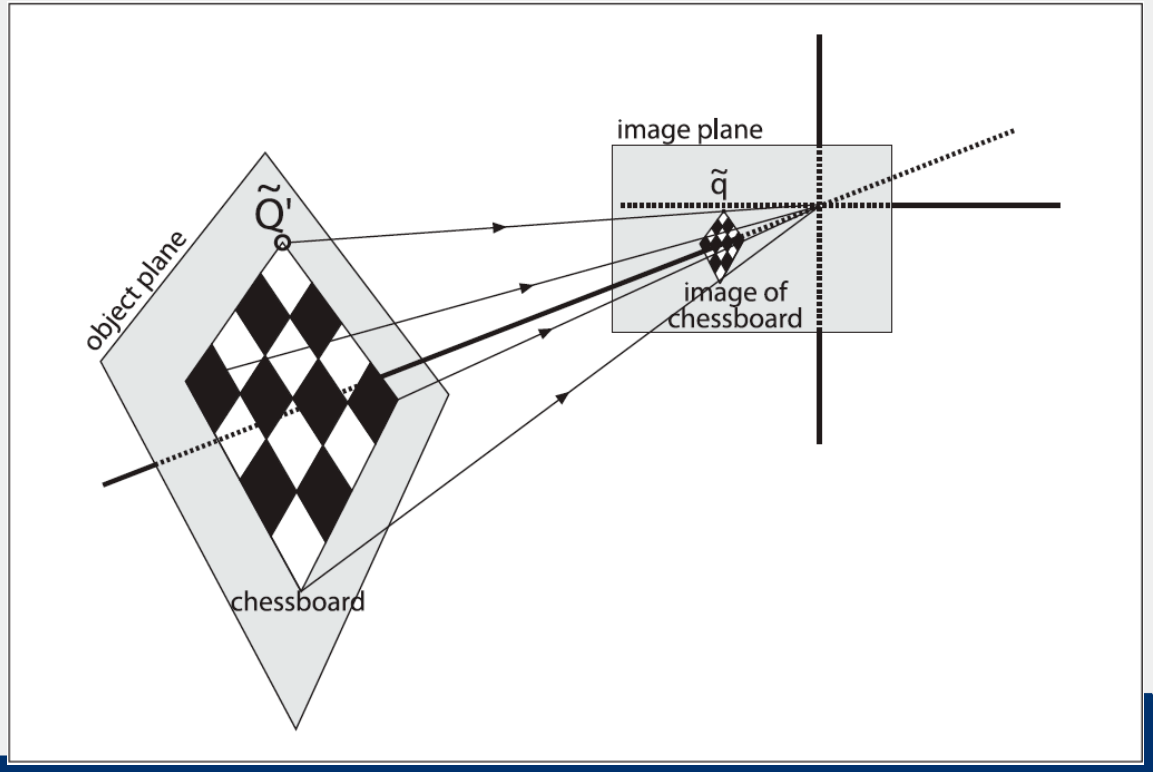
Find internal chessboard corners (where black squares touch each other) from grayscale image. Use cornerSubPix() to refine corner locations and drawChessboardCorners() to draw found corners on a color image (convert single channel to 3 channels).

void cornerSubPix(const Mat& image, vector<Point2f>& corners, Size winSize, Size zeroZone, TermCriteria criteria);

void drawChessboardCorners(Mat& image, Size patternSize, const Mat& corners, bool patternWasFound);

Homography

- Planar homography is defined as a projective mapping from one plane (object plane) to another (image plane).



Homography

- Using homogeneous coordinates the relationship between two planes can be expressed as

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = sH \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

- *s is an arbitrary scale factor because homography is only defined up to a scale factor.*
- x_i *and* y_i are in image pixels
- *H includes two parts: the physical transformation that locates the object plane and the projection that has the camera intrinsic parameters*

Homography

- Remember the projection matrix M (transformation between world and image

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = M \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = M_{\text{int}} M_{ext} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = \begin{bmatrix} fs_x & 0 & o_x \\ 0 & fs_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & T_x \\ r_{21} & r_{22} & r_{23} & T_y \\ r_{31} & r_{32} & r_{33} & T_z \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

Since object is a planar surface and it can be chosen as the world coordinate system, Z_w of all corner points can be set to zero.

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = sH \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = sM_{\text{int}} M_{ext} \begin{bmatrix} X_w \\ Y_w \\ 0 \\ 1 \end{bmatrix} = s \begin{bmatrix} fs_x & 0 & o_x \\ 0 & fs_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & T_x \\ r_{21} & r_{22} & T_y \\ r_{31} & r_{32} & T_z \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ 1 \end{bmatrix}$$

Homography

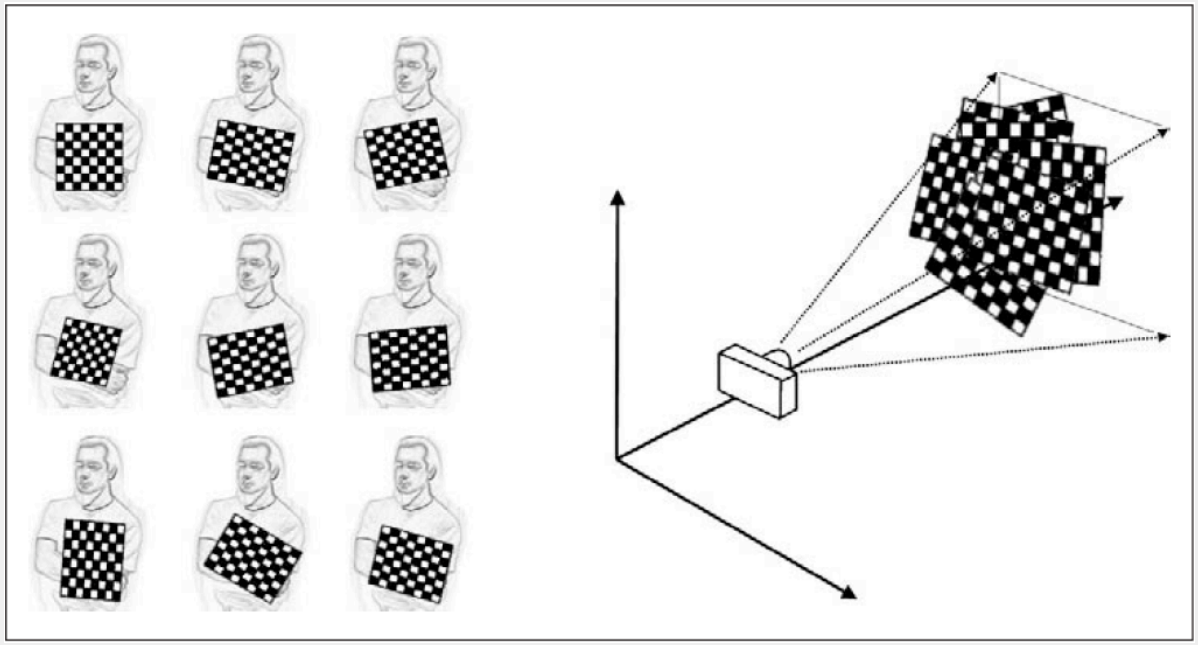
- The homography matrix H that maps a planar object's points onto the imager is described completely by a 3x3 matrix.

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = sH \begin{bmatrix} X_w \\ Y_w \\ 1 \end{bmatrix}$$

- *OpenCV uses this equation to compute a homography matrix for each view (we have enough points).*
- *We will have one set of new extrinsic parameters (define homography) for each view. However, the intrinsic parameters stay the same.*
- *Computing multiple H's from multiple views can lead us to a unique set of intrinsic parameters.*

Calibration

double calibrateCamera(const vector<vector<Point3f> >&
objectPoints, const vector<vector<Point2f> >& imagePoints,
Size imageSize, Mat& cameraMatrix, Mat& distCoeffs,
vector<Mat>& rvecs, vector<Mat>& tvecs,
int flags=0);



Find Homography

Mat findHomography(const Mat& srcPoints, const Mat& dstPoints, Mat& status, int method=0, double ransacReprojThreshold=3);

Mat findHomography(const Mat& srcPoints, const Mat& dstPoints, vector<uchar>& status, int method=0, double ransacReprojThreshold=3);

Mat findHomography(const Mat& srcPoints, const Mat& dstPoints, int method=0, double ransacReprojThreshold=3);

Method:

0 a regular method using all the points (regular least squares error)

CV_RANSAC RANSAC-based robust method (RANdom SAmple Consensus)

CV_LMEDS Least-Median Squares robust method

Only eight free parameters in H, the entire matrix H can be normalized by dividing the entire matrix by a scale factor to make $H_{33} = 1$.

Camera Parameters

Intrinsic parameters: – 3-D geometry

(O_x, O_y) is the image center in pixels

S_x and S_y are the pixel size in pixels per mm or inches

f is the focal length in mm or inches

σ is the aspect ratio S_y/S_x (usually 1.0 for a square pixel camera)

$$o_x,\ o_y,\ fs_x,\ fs_y$$

Distortion parameters:

Tied to 2-D geometry

$$k_1,\ k_2,\ k_3,\ p_1,\ p_2$$

Extrinsic parameters:

Rotation and translation – 3-D geometry

$$\mathbf{T}=[T_x \quad T_y \quad T_z]^{\mathrm{T}}$$

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

Calibration – Distortion Parameters

- Three corner points (3 x's and 3 y's) from one pattern of one view is sufficient to solve for 5 distortion parameters.
- More points yield an optimal solution (minimum error).
- More points will reduce effects from noise

Calibration – Extrinsic Parameters

- Rotation can be represented by rotations about three individual axes.

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\phi & 0 & \sin\phi & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\phi & 0 & \cos\phi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\psi & -\sin\psi & 0 & 0 \\ \sin\psi & \cos\psi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- We need to solve for 6 parameters, θ, ϕ, and Ψ for rotation and T_x, T_y, and T_z for translation

How many corners & how many views?

- *Size of chessboard is not relevant for calibration, why?*
- *N corners and K images of the chessboard (in different positions).*
- *K images of the chessboard provide 2NK constraints (2 because x and y coordinate).*
- Ignoring the distortion parameters for the moment
- Need to solve for 4 intrinsic parameters and 6*K extrinsic parameters (the intrinsic parameters stay the same)*
- *2NK must be* $\geq 6K + 4$ *(or, equivalently,* $(N - 3)\,K \geq 2$*) to solve for these parameters*
- If $N = 5$ *then we need only* $K = 1$ *image*
- *K* must be more than 1. At lease two images to fit a homography matrix.

How many corners & how many views?

- A homography can yield at most eight parameters from four *(x, y) pairs.*
- *This is because only four points are needed to express everything that a planar* perspective view can do.
- So, no matter how many corners we detect on a plane, we only get four corners' worth of information.
- Per chessboard view, then, the equation can give us only four corners of information or $(4 - 3)\ K > 1$, *which means* $K > 1$.
- *This implies that two views of a 3-by-3 chessboard internal corners* are the minimum that could solve our calibration problem.
- More images of a larger chessboard to achieve numerical stability.
- In practice, for high-quality results, we'll need at least 10 images of a 7x8 or larger chessboard (and that's only if we move the chessboard enough between images to obtain a "rich" set of views).

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = H \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = sM_{\text{int}} M_{ext} \begin{bmatrix} X_w \\ Y_w \\ 0 \\ 1 \end{bmatrix} = s \begin{bmatrix} fs_x & 0 & o_x \\ 0 & fs_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & T_x \\ r_{21} & r_{22} & T_y \\ r_{31} & r_{32} & T_z \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ 1 \end{bmatrix}$$

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \end{bmatrix} = s \begin{bmatrix} fs_x & 0 & o_x \\ 0 & fs_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & T_x \\ r_{21} & r_{22} & T_y \\ r_{31} & r_{32} & T_z \end{bmatrix} = sM_{(\text{int})} \begin{bmatrix} R_1 & R_2 & T \end{bmatrix}$$

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \end{bmatrix} = sM_{(\text{int})} \begin{bmatrix} R_1 & R_2 & T \end{bmatrix}$$

$$R_1 = \frac{1}{s} M^{-1} h_1 \qquad R_2 = \frac{1}{s} M^{-1} h_1 \qquad \mathrm{T} = \frac{1}{s} M^{-1} h_3$$

$$R_1^T R_2 = (\frac{1}{s} M^{-1} h_1)^T (\frac{1}{s} M^{-1} h_2) = 0 \quad \text{orthonormal}$$
$$\Rightarrow h_1^T (M^{-1})^T M^{-1} h_2 = 0$$

$$\|R_1\| = \|R_2\| = R_1^T R_1 = R_2^T R_2 \quad \text{orthonormal (same length)}$$
$$\Rightarrow h_1^T (M^{-1})^T M^{-1} h_1 = h_2^T (M^{-1})^T M^{-1} h_2$$

$$\text{Let } B = (M^{-1})^T M^{-1} = \begin{bmatrix} \frac{1}{f_x^2} & 0 & \frac{-c_x}{f_x^2} \\ 0 & \frac{1}{f_y^2} & \frac{-c_y}{f_y^2} \\ \frac{-c_x}{f_x^2} & \frac{-c_y}{f_y^2} & \frac{c_x}{f_x^2} + \frac{c_y}{f_y^2} + 1 \end{bmatrix} = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{12} & B_{22} & B_{23} \\ B_{13} & B_{23} & B_{33} \end{bmatrix}$$

B is a symmetrical matrix

$$h_i^T B h_j = \begin{bmatrix} h_{i1}h_{j1} \\ h_{i1}h_{j2} + h_{i2}h_{j1} \\ h_{i2}h_{j2} \\ h_{i3}h_{j1} + h_{i1}h_{j3} \\ h_{i3}h_{j2} + h_{i2}h_{j3} \\ h_{i3}h_{j3} \end{bmatrix}^T \begin{bmatrix} B_{11} \\ B_{12} \\ B_{22} \\ B_{13} \\ B_{23} \\ B_{33} \end{bmatrix} = \mathrm{v}_{ij}^{\mathrm{T}} \mathrm{b}$$

$$h_1^T (M^{-1})^T M^{-1} h_2 = h_1^T B h_2 = v_{12}^T b = 0$$
$$h_1^T (M^{-1})^T M^{-1} h_1 = h_2^T (M^{-1})^T M^{-1} h_2$$
$$\Rightarrow h_1^T B h_1 - h_2^T B h_2 = v_{11}^T b - v_{22}^T b = 0$$

$$\begin{bmatrix} v_{12}^T \\ (v_{11}^T - v_{22}^T) \end{bmatrix} b = 0$$

Collect k images of chessboards together, we can stack k of

$$\begin{bmatrix} v_{12}^T \\ (v_{11}^T - v_{22}^T) \end{bmatrix} b = 0 \text{ togrther} \Rightarrow Vb = 0 \quad \text{V is a } 2\text{K} \times 6 \text{ matrix}$$

- $B=[B_{11}, B_{12}, B_{13}, B_{22}, B_{23}, B_{33}]^T$ can be solved.
- Camera intrinsic parameters can be extracted from B

$$\begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{12} & B_{22} & B_{23} \\ B_{13} & B_{23} & B_{33} \end{bmatrix} = \begin{bmatrix} \frac{1}{f_x^2} & 0 & \frac{-c_x}{f_x^2} \\ 0 & \frac{1}{f_y^2} & \frac{-c_y}{f_y^2} \\ \frac{-c_x}{f_x^2} & \frac{-c_y}{f_y^2} & \frac{c_x}{f_x^2} + \frac{c_y}{f_y^2} + 1 \end{bmatrix}$$

Intrinsic Parameters

- Remember a scale factor (λ=1/s) was cancelled out and must be restored.

$$\begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{12} & B_{22} & B_{23} \\ B_{13} & B_{23} & B_{33} \end{bmatrix} = \begin{bmatrix} \frac{1}{f_x^2} & 0 & \frac{-c_x}{f_x^2} \\ 0 & \frac{1}{f_y^2} & \frac{-c_y}{f_y^2} \\ \frac{-c_x}{f_x^2} & \frac{-c_y}{f_y^2} & \frac{c_x}{f_x^2} + \frac{c_y}{f_y^2} + 1 \end{bmatrix}$$

$$f_x = \sqrt{\lambda / B_{11}} \qquad f_y = \sqrt{\lambda B_{11} / (B_{11}B_{22} - B_{12}^2)}$$

$$c_x = -B_{13} f_x^2 / \lambda \qquad c_y = (B_{12}B_{13} - B_{11}B_{23}) / (B_{11}B_{22} - B_{12}^2)$$

$$\lambda = B_{33} - (B_{13}^2 + c_y(B_{12}B_{13} - B_{11}B_{23})) / B_{11}$$

Extrinsic Parameters

$$R_1 = \lambda M^{-1} h_1 \qquad R_2 = \lambda M^{-1} h_1 \qquad R_3 = R_1 \times R_2 \qquad \mathrm{T} = \lambda M^{-1} h_3$$

$$\lambda = 1 \Big/ \left\| M^{-1} h_1 \right\| \qquad \because R_1^T R_1 = I$$

Same as the direct method

$$[\mathrm{r}_{31}\ \mathrm{r}_{32}\ \mathrm{r}_{33}] = [\mathrm{r}_{11}\ \mathrm{r}_{12}\ \mathrm{r}_{13}] \times [\mathrm{r}_{21}\ \mathrm{r}_{22}\ \mathrm{r}_{23}] = \begin{bmatrix} \mathrm{i} & \mathrm{j} & \mathrm{k} \\ \mathrm{r}_{11} & \mathrm{r}_{12} & \mathrm{r}_{13} \\ \mathrm{r}_{21} & \mathrm{r}_{22} & \mathrm{r}_{23} \end{bmatrix}$$

$$= [\mathrm{r}_{12}\mathrm{r}_{23} - \mathrm{r}_{13}\mathrm{r}_{22} \quad \mathrm{r}_{11}\mathrm{r}_{23} - \mathrm{r}_{13}\mathrm{r}_{21} \quad \mathrm{r}_{11}\mathrm{r}_{22} - \mathrm{r}_{12}\mathrm{r}_{21}]$$

Question – How can this possible? We only have one camera. Is it possible to solve for a unique R and T?

Orthogonality

Again, due to the noise the resulting rotation matrix may not meet the orthogonality requirement.

$$RR^T = I$$

Use SVD to enforce this by

replacing D with the 3×3 identity matrix

$$\hat{\mathrm{R}} = \mathrm{UDV}^{\mathrm{T}} \qquad R = U \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} V^T$$

Distortion Parameters

If the pinhole camera were perfect and there were no distortion

$$\begin{bmatrix} x_{perfect} \\ y_{perfect} \end{bmatrix} = \begin{bmatrix} f_x \, X_w / Z_w + c_x \\ f_y \, Y_w / Z_w + c_y \end{bmatrix}$$

We made this assumption and use this equation to calculate initial intrinsic and extrinsic parameters.

$$\begin{bmatrix} x_{perfect} \\ y_{perfect} \end{bmatrix} = (1 + k_1^2 + k_2^4 + k_3^6) \begin{bmatrix} x_{distorted} \\ y_{distorted} \end{bmatrix} + \begin{bmatrix} 2p_1 x_{distorted} y_{distorted} + p_2 (r^2 + 2x_{distorted}^2) \\ p_1 (r^2 + 2y_{distorted}^2) + 2p_2 x_{distorted} y_{distorted} \end{bmatrix}$$

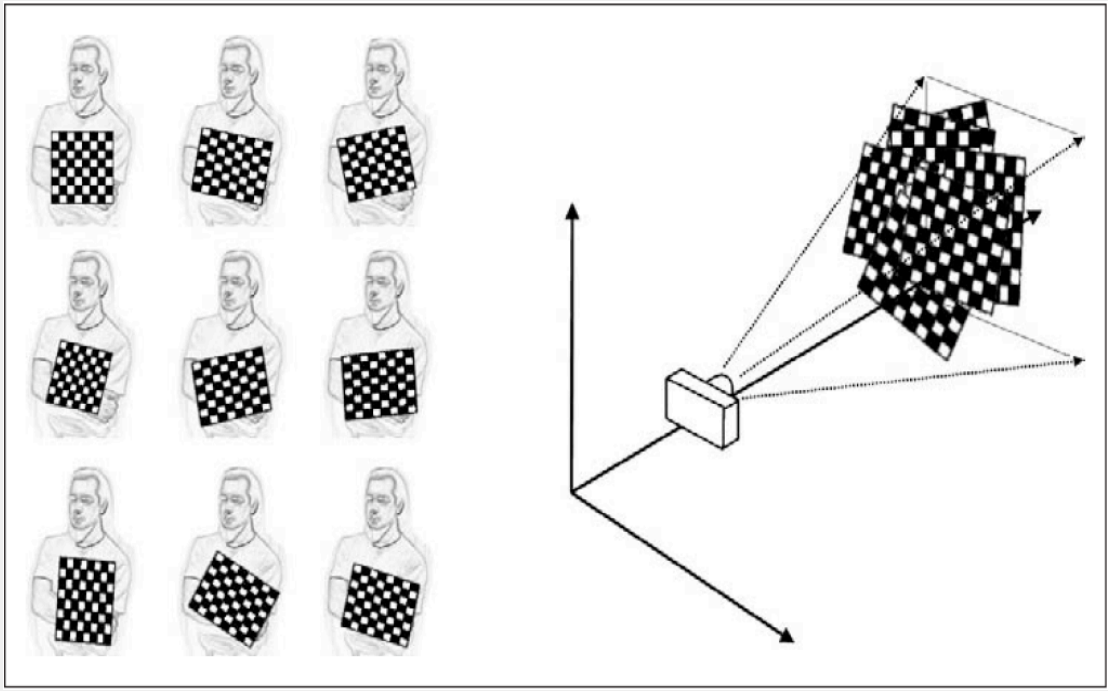
Distortion Parameters

- We assumed that there was no lens distortion to calculate the intrinsic and extrinsic parameters.
- We need to use the calculated intrinsic and extrinsic parameters to obtain the distortion parameters
- We then use the distortion parameters to refine the intrinsic and extrinsic parameters.

Calibration

double calibrateCamera(const vector<vector<Point3f> >&
objectPoints, const vector<vector<Point2f> >& imagePoints,
Size imageSize, Mat& cameraMatrix, Mat& distCoeffs,
vector<Mat>& rvecs, vector<Mat>& tvecs, int flags=0);

Object points are for 3D corner points on the chessboard measured in real units (mm or inches) with all z's = 0.
These measurements can have an arbitrary scale factor for single camera calibration.
The exact measurements of these corner points are critical for stereo calibration.



- We get unique intrinsic and distortion parameters.
- We get rotation and translation parameters for each image view.
- Using a single camera, we cannot get depth information. All we get is a line in 3-D to which a given image point must correspond.
- We do not get a 3x3 rotation matrix. We get a counterclockwise angle of rotation of each axis.
- Rodrigues() is used to convert these angles to a 3x3 rotation matrix.

Distortion Correction

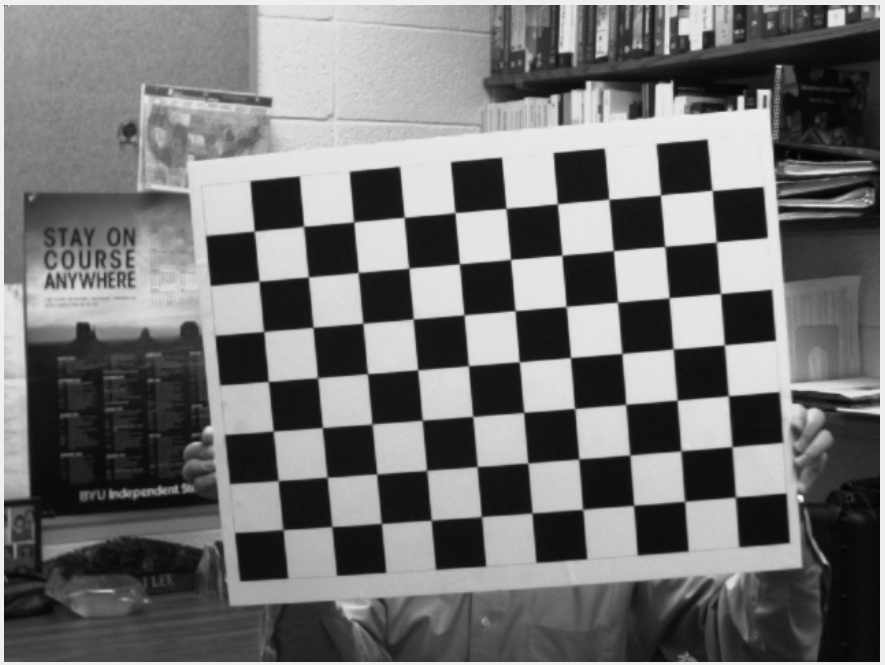
- With the distortion parameters, input image can be undistorted.
- Calculate the map once and remap the image

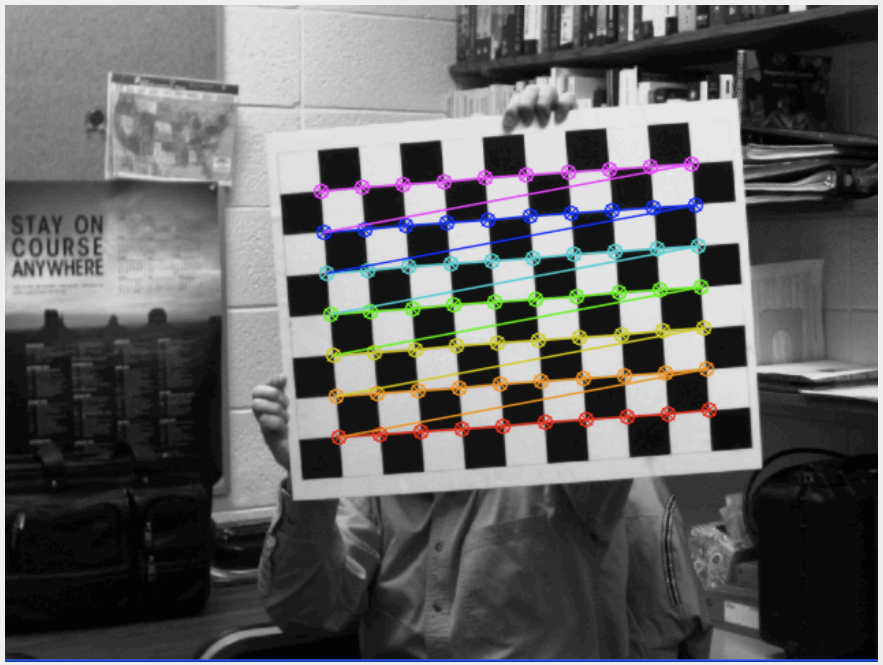
  void initUndistortRectifyMap () ➔void Remap()
- Perform both together (but no reason to calculate the map every time).

  void undistort ()
- Undistort a set of 2-D points (tennis ball location)

  void undistortPoints ()

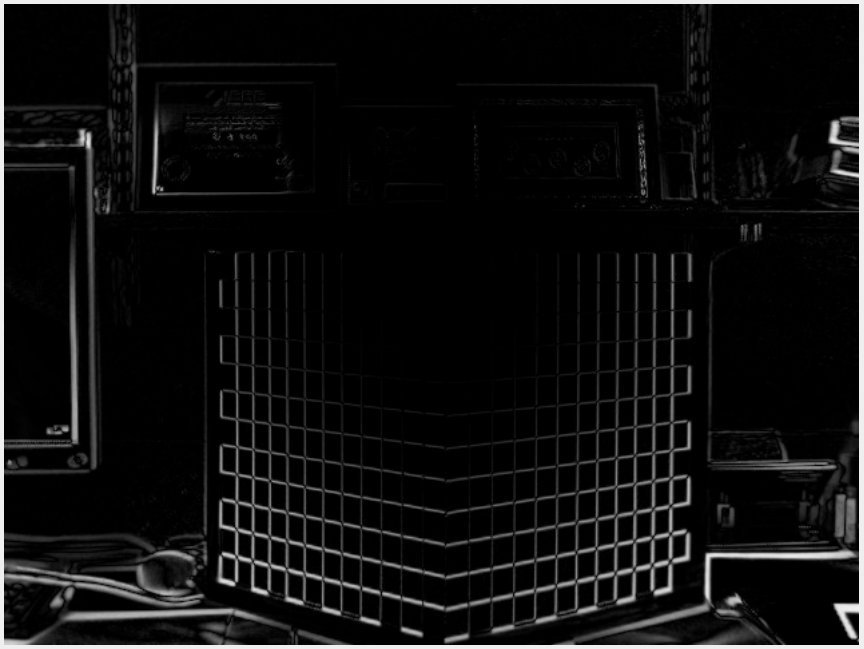
Assignment 2





Assignment 2

Original Image



Absolute difference between the original and the undistorted images

Assignment 2

One unique set of intrinsic parameters

One set of extrinsic parameters per view

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = H \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = sM_{\text{int}} M_{ext} \begin{bmatrix} X_w \\ Y_w \\ 0 \\ 1 \end{bmatrix} = s \begin{bmatrix} fs_x & 0 & o_x \\ 0 & fs_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & T_x \\ r_{21} & r_{22} & T_y \\ r_{31} & r_{32} & T_z \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ 1 \end{bmatrix}$$

- f is the focal length in mm and s_x and s_y are the pixels per mm scale factors.
- $f_x = fs_x$ and $f_y = fs_y$ are focal lengths in pixels and they are all we can get.

Assignment 2

- How do we know if the parameters are correct?
- O_x and O_y should be very close to 324 and 244 (within 30 pixels)
- For example, Point Grey Flea2 camera uses a 1/3" format CMOS sensor that has 648x488 pixels.
- The camera spec sheet says the pixel area size is 7.4µm x 7.4µm
- The actual sensor size is 4.8 mm x 3.6 mm or 6.0 mm diagonal.
- $s_x = s_y = 648/4.8 = 135$ pixels/mm (square pixel)
- For a 16 mm lens, $f_x = f_y = 16$ mm x 135 = 2160 pixels
- For a 12 mm lens, $f_x = f_y = 12$ mm x 135= 1620 pixels
- For a 8.5 mm lens, $f_x = f_y = 8.5$ mm x 135 =1147.5pixels