6.801/6.866: Machine Vision, Lecture 21

Professor Berthold Horn, Ryan Sander, Tadayuki Yoshitake
MIT Department of Electrical Engineering and Computer Science
Fall 2020

These lecture summaries are designed to be a review of the lecture. Though I do my best to include all main topics from the lecture, the lectures will have more elaborated explanations than these notes.

1 Lecture 21: Relative Orientation, Binocular Stereo, Structure from Motion, Quadrics, Camera Calibration, Reprojection

Today, we will continue discussing **relative orientation**, one of our main problems in photogrammetry. We concluded our previous lecture with a discussion of critical surfaces - that is, surfaces which make solving the relative rotation problem difficult. In general, these critical surfaces are **quadric surfaces**, which take the form:

$$\pm\left(\frac{x}{a}\right)^2 \pm \left(\frac{y}{b}\right)^2 \pm \left(\frac{z}{c}\right)^2 = 1$$

This form is simple: (i) Centered at the origin, (ii) Axes line up. In general, the form of these quadric surfaces may be more complicated, with both linear and constant terms. The signs of these qudratic terms determine the shape, as we saw in the previous lecture:

1. $+++$ (ellipsoid)
2. $++-$ (hyperboloid of one sheet) - this is a class of critical surfaces
3. $+--$ (hyperboloid of two sheets)
4. $---$ (imaginary ellipsoid)

For the solutions to the polynomial $R^2 + \cdots - R = 0$ (the class of **hyperboloids of one sheets**), the solutions tell us why these critical surfaces create solution ambiguity:

1. $\mathbf{R} = \mathbf{0}$: I.e. the solution is on the surface - the origin of the righthand system is on the surface. Two interpretations from this; the first is for binocular stereo, and the second is for Structure from Motion (SfM):
   (a) **Binocular Stereo**: The righthand system lies on the surface.
   (b) **Structure from Motion**: In Structure from Motion, we "move ont the surface" in the next time step.
2. $\mathbf{R} = -\mathbf{b}$: This solution is the origin of the lefthand system (we move the solution left along the baseline from the righthand system to the lefthand system). Two interpretations from this; the first is for binocular stereo, and the second is for Structure from Motion (SfM):
   (a) **Binocular Stereo**: "The surface has to go through both eyes".
   (b) **Structure from Motion**: We must start and end back on the surface.
3. $\mathbf{R} = k\mathbf{b}$: Because this system (described by the polynomial in $\mathbf{R}$, then we have no scaling, giving us a solution. This suggests that the entire baseline in fact lies on the surface, which in turn suggests:
   - This setting/surface configuration is rare.
   - This surface is **ruled** - we can draw lines inscribed in the surface. This suggests that our critical surface is a hyperboloid of one sheet. It turns out that we have two rulings for the hyperboloid at one sheet (i.e. at every point on this surface, we can draw two non-parallel lines through the surface that cross at a point).

Hyperboloids of one sheet, as we saw in the previous lecture, are not the only types of critical surfaces. **Intersections of planes** are another key type of critical surface, namely because they are much more common in the world than hyperboloids of one sheet. Analytically, the intersection of planes is given by the product of planes:

$$(a_1X + b_1Y + c_1Z + d_1)(a_2X + b_2Y + c_2Z + d_2) = 0$$

This analytic equation describes the intersection of two planes - gives us a quadric surface.

In order for this intersection of planes to not have any constants, one of the planes must be the **epipolar plane**, whose image is a line. The second plane is arbitrary and can take any form.

Practically, we may not run into this problem in this exact analytic form (i.e. a surface that is an exact instance of a hyperboloid of one sheet), but even as we venture near this condition, the noise amplification factor increases. As it turns out, using a higher/wider Field of View (FOV) helps to mitigate this problem, since as FOV increases, it becomes less and less likely that the surface we are imaging is locally similar to a critical surface.

One way that we can increase the FOV is through the use of **spider heads**, which consist of 8 cameras tied together into a rigid structure. Though it requires calibration between the cameras, we can "stitch" together the images from the 8 cameras into a single "mosaic image".

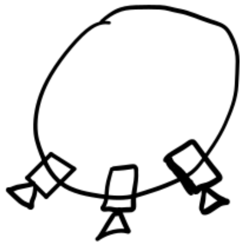


Figure 1: A spider heads apparatus, which is one technique for increasing the FOV, thereby reducing the probability of surfaces resembling critical surfaces.

Next, we will switch our discussion to another of our photogrammetry problems - interior orientation.

1.1 Interior Orientation

Recall that interior orientation was another one of the 4 photogrammetry problems we study, and involves going from 3D (world coordinate) to 2D (image coordinates). Specifically, we focus on **camera calibration**.

Have we already touched on this? Somewhat - with **vanishing points**. Recall that the goal with vanishing points was to find (x_0, y_0, f) using calibration objects. This method:

- Is not very accurate, nor general across different domains/applications
- Does not take into account radial distortion, which we need to take into account for high-quality imaging such as aerial photography.

What is radial distortion? We touch more on this in the following section.

1.1.1 Radial Distortion

This type of distortion leads to a discrepancy between what should be the projected location of an object in the image, and what it actually projects to. This distortion is radially-dependent.

This type of distortion becomes more apparent when we image lines/edges. It manifests from having a **center of distortion**. The image of an object does not appear exactly where it should, and the error/discrepancy depends on the radius of the point in the image relative to the center of distortion. This radial dependence is typically approximated as a polynomial in the radius r:

$$\begin{aligned}
r &= ||\mathbf{r}||_2 \\
\delta_x &= x(k_1r^2 + k_2r^4 + \cdots) \\
\delta_y &= y(k_1r^2 + k_2r^4 + \cdots)
\end{aligned}$$

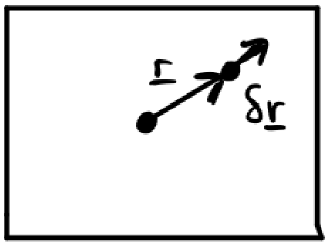
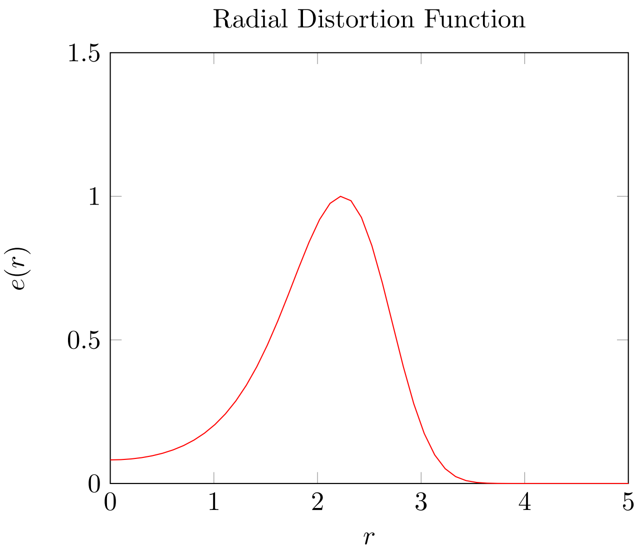


Figure 2: Radial distortion is in the direction of the radius emanating from the center of projection to the point in the image plane where a point projects to in the absence of radial distortion.

Note that the error vector $\delta\mathbf{r}$ is parallel to the vector $\mathbf{r}$ - i.e. the distortion is in the direction of the radial vector.

Many high-quality lenses have a defined radial distortion function that gives the distortion as a function of r, e.g. the figure below:



These functions, as we stated above, are typically approximated as polynomial functions of r. **How do we measure radial distortion?** One way to do so is through the use of **plumb lines**. These lines are suspended from the ceiling via weights, which allows us to assume that they are straight and all parallel to one another in the 3D world. Then, we take an image of these lines, with the image centered on the center lines. We can estimate/assess the degree of radial distortion by identifying the curvature, if any, of the lines as we move further away from the center lines of the image.

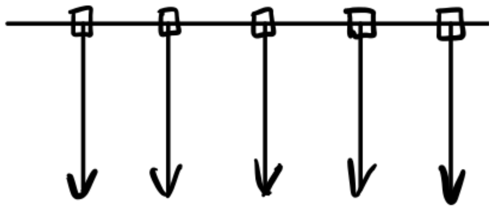


Figure 3: Plumb lines are a useful way of estimating radial distortion, because they allow us to estimate the degree of curvature between lines that should be straight and parallel.

Plumb lines allow us to estimate two types of radial distortion: (i) **Barrel Distortion** and (ii) **Pincushion Distortion**, depicted below. The radius of curvature from these lines allows us to measure k_1 (the first polynomial coefficient for r^2):

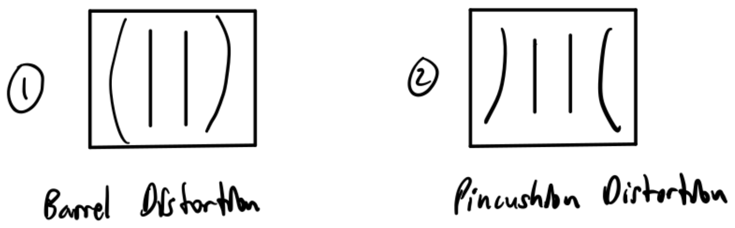


Figure 4: Barrel and pincushion distortion, which we can absorb from plumb lines.

A more subtle question: is it better to go from)a) **undistorted** $\rightarrow$ **distorted** coordinates, or from (b) **distorted** $\rightarrow$ **undistorted** coordinates?

It is often desirable to take approach (b) (**distorted** $\rightarrow$ **undistorted**) because we can measure the distorted coordinates. Wse can use **series inversion** to relate the distorted and undistorted coodinates. This affects the final coordinate system that we do optimization in, i.e. we optimize error in the image plane.

1.1.2 Tangential Distortion and Other Distortion Factors

There are other types of distortion that we can analyze and take into account:

1. **Tangential Distortion**: This is a problem we have solved, but to contrast this with radial distortion, rather than the distortion acting radially, the distortion acts tangentially (see figure below). With tangential distortion, our offsets take the form:

$$\delta_x = -y(\epsilon_1 r^2 + \epsilon_2 r^4 + \cdots)$$
$$\delta_y = x(\epsilon_1 r^2 + \epsilon_2 r^4 + \cdots)$$

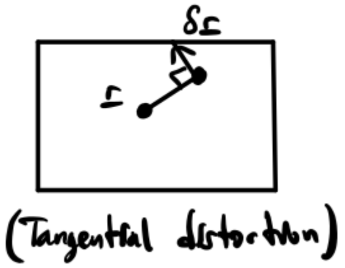


Figure 5: Tangential distortion acts tangentially relative to the radius - in this case, in a counterclockwise fashion (as we rotate θ).

2. **Decentering**: If the center of distortion is not the principal point (center of the image with perspective projection), then we get an offset that depends on position. This offset is typically small, but we still need to take it into account for high-quality imaging such as aerial photography.

3. **Tilt of Image Plane**: If the image plane is tilted (see the figure below), then magnification will not be uniform and the focus will not be perfect. In practice, to fix this, rather than changing the tilt of the image plane, you can instead insert a **compensating element** to offset this tilt.

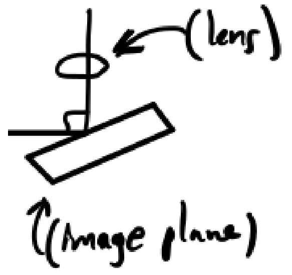


Figure 6: Tilt of the image plane is another factor that can affect distortion of images, as well as variable magnification/scaling across the image. Oftentimes, it can be corrected via a compensating element that offsets this tilt, rather than realigning the image plane.

Taking these effects into account allows for creating more sophisticated distortion models, but sometimes at the expense of overfitting and loss of generalizability (when we have too many degrees of freedom, and our estimates for them become so precariously configured that our model is not applicable for a slight change of domain).

These distortion models can also be used for nonlinear optimization protocols such as Levenberg-Marquadt (LM) or Gauss-Newton (GN) (see lecture 20 for a review of these).

1.2 Tsai's Calibration Method

This calibration method relies on using a calibration object for which we have precise knowledge of the points on our 3D points of the object.

In camera calibration, **correspondences** are defined between **points in the image** and **points on the calibration object**.

Here, we encounter the same reason that vanishing point-based methods are difficult - it is hard to directly relate the camera to calibration objects. When we take **exterior orientation** into account, we generally perform much better and get higher-performing results. Exterior orientation (2D $\rightarrow$ 3D) seeks to find the calibration object in space given a camera image.

Adding **exterior orientation** also increases the complexity of this problem:

- **Interior orientation** has 3 Degrees of Freedom
- **Exterior orientation** has 6 Degrees of Freedom (translation and rotation)

Combined, therefore our problem now possesses 9 DOF. We can start solving this problem with perspective projection and interior orientation.

1.2.1 Interior Orientation

Beginning with perspective projection, we have:

$$\frac{x_I - x_O}{f} = \frac{X_c}{Z_c}$$
$$\frac{x_I - x_O}{f} = \frac{X_c}{Z_c}$$

Where:

1. $(X_c, Y_c, Z_c)^T$ are the camera coordinates (world coordinate units)
2. (x_I, y_I) denote image position (row, column/grey level units)
3. (x_O, y_O, f) denotie interior orientation/principal point (column/grey level and pixel units)

Can we modify the measurements so we are able to take radial distortion into account?

We need a good initial guess, because, numerical, iterative approaches that are used to solve this problem precisely exhibit multiple mimima, and having a good initial guess/estimate to serve as our "prior" will help us to find the correct minima.

From here, we can add exterior orientation.

1.2.2 Exterior Orientation

As we are solving for translation and rotation, our equation for exterior orientation equation becomes (written as a matrix-vector problem):

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X_s \\ Y_s \\ Z_s \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$
$$\mathbf{X}_c = \mathbf{R}\mathbf{X}_s + \mathbf{t}$$

The figure below illustrates the desired transformation between these two reference frames:

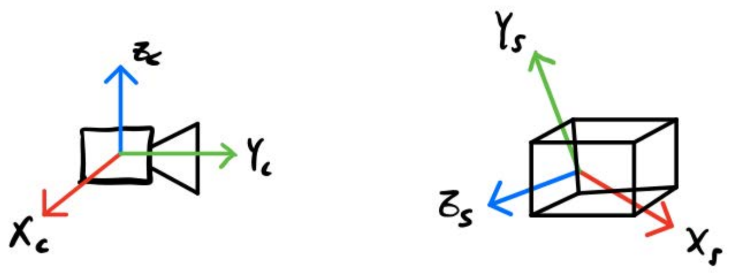


Figure 7: Exterior orientation seeks to find the transformation between a camera and a calibration object.

Where the matrix **R** and the translation **t** are our unknowns. From this, we can combine these equations with our equations for interior orientation to find our solutions.

1.2.3 Combining Interior and Exterior Orientation

Combining our equations from interior and exterior orientation:

- **Interior Orientation**:

$$\frac{x_I - x_O}{f} = \frac{X_c}{Z_c}$$
$$\frac{x_I - x_O}{f} = \frac{X_c}{Z_c}$$

- **Exterior Orientation**:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X_s \\ Y_s \\ Z_s \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$
$$\mathbf{X}_c = \mathbf{R}\mathbf{X}_s + \mathbf{t}$$

Then we can combine these equations into:

$$\frac{x_I - x_O}{f} = \frac{X_c}{Z_c} = \frac{r_{11}X_s + r_{12}Y_s + r_{13}Z_s + t_x}{r_{31}X_s + r_{32}Y_s + r_{33}Z_s + t_z}$$

$$\frac{y_I - y_O}{f} = \frac{Y_c}{Z_c} = \frac{r_{21}X_s + r_{22}Y_s + r_{23}Z_s + t_y}{r_{31}X_s + r_{32}Y_s + r_{33}Z_s + t_z}$$

Written this way, we can map directly from **image coordinates** to **calibration objects**.

How can we find f and t_z, as well as mitigate radial distortion?

Let's start by looking at this problem in polar coordinates. Radial distortion only changes radial lengths (r), and not angle (θ). Changing f or Z_c changes only radius/magnification. Let's use this factor to "forget" about the angle.

We can also remove additional DOF by **dividing our combined equations** by one another:

$$\frac{x_I - x_O}{y_I - y_0} = \frac{\frac{r_{11}X_s + r_{12}Y_s + r_{13}Z_s + t_x}{r_{31}X_s + r_{32}Y_s + r_{33}Z_s + t_z}}{\frac{r_{21}X_s + r_{22}Y_s + r_{23}Z_s + t_y}{r_{31}X_s + r_{32}Y_s + r_{33}Z_s + t_z}}$$

$$= \frac{r_{11}X_s + r_{12}Y_s + r_{13}Z_s + t_x}{r_{21}X_s + r_{22}Y_s + r_{23}Z_s + t_y}$$

We can see that combining these constraints removes our DOF t_z, giving us 8 constraints rather than 9. Cross-multiplying and term consolidation gives:

$$(X_S y_I')r_{11} + (Y_S y_I')r_{12} + (Z_S y_I')r_{13} + y_I' t_x - (X_S x_I')r_{21} - (Y_S x_I')r_{22} - (Z_S x_I')r_{23} - x_I' t_y = 0$$

A few notes on this consolidated homogeneous equation:

- Since our unknowns are the r_{ij} values, we have a linear combination/equation of our unknowns! The coefficients they are multiplied by are either our observed image positions or the positions of the correspondences/keypoints on the calibration object.
- Note that we subtract out an estimate of the center of projection. The estimation error of this center of projection affects adjacent points the most, as a slight miscalculation can greatly affect the calculations above - and as a result we throw out all correspondences that are ϵ-close (for some $\epsilon > 0$ to the center of the image.
- Each correspondence between image coordinates and the points on the calibration object gives one of these equations: "This point in the image corresponds to another point on the calibration object".

As we have asked for the other frameworks and systems we have analyzed: **How many correspondences do we need?** Let's consider which unknowns appear in the equations, and which do not:

- DOF/unknowns in the equations: $r_{11}, r_{12}, r_{13}, r_{21}, r_{22}, r_{23}, t_x, t_y$ (8 unknowns)
- DOF/unknowns not in the equations: $r_{31}, r_{32}, r_{33}, t_z, f$ (5 unknowns)

Can this inform of how we can solve this problem? Let's consider the following points:

- We are not yet enforcing orthonormality of the rotation matrices, i.e. $\mathbf{R}^T\mathbf{R} = \mathbf{I}$.
- We can solve for the last row of the rotation matrix containing the unknowns r_{31}, r_{32}, r_{33} through finding the cross product of the first two rows of this matrix.
- Having 8 unknowns suggests that we need at least 8 correspondences, but because this equation is homogeneous, there is **scale factor ambiguity** (i.e. doubling all the variables does not change the righthand side value of the equation). The solution to account for this scale factor ambiguity is to set/fix one of these variables to a constant value. Fixing an unknown means we now have 7 DOF (rather than 8), and therefore, **we actually need 7 correspondences**. Solving with 7 (or more) correspondences gives us a multiple of the solution. Solving with exactly 7 correspondences gives us 7 equations and 7 unknowns, which we can solve with, for instance, Gauss-Jordan elimination. More correspondences are desirable because they allow us to estimate both solutions and the error associated with these solutions.
- These solutions give us values for the 8 unknowns in the equation above: $r_{11}', r_{12}', r_{13}', r_{21}', r_{22}', r_{23}', t_x, t_y = 1$ (where t_y is the fixed parameter to account for scale ambiguity). However, as aforementioned these values are scaled versions of the true solution, and we can compute this scale factor by noting that because the rows of our (soon to be) orthonormal rotation matrix must have unit length - therefore the scale factor is given by computing the length of the rows of this matrix:

  $$s = \frac{1}{\sqrt{{r_{11}'}^2 + {r_{12}'}^2 + {r_{13}'}^2}}$$
  OR
  $$s = \frac{1}{\sqrt{{r_{21}'}^2 + {r_{22}'}^2 + {r_{23}'}^2}}$$

  A good sanity check: these two variables should evaluate to approximately the same value - if they do not, this is often one indicator of correspondence mismatch.

With this scale factor computed we can find the true values of our solution by multiplying the scaled solutions with this scale factor:

$$s(r'_{11}, r'_{12}, r'_{13}, r'_{21}, r'_{22}, r'_{23}, t_x, t_y = 1) \rightarrow r_{11}, r_{12}, r_{13}, r_{21}, r_{22}, r_{23}, t_x, t_y$$

Note that we also have not (yet) enforced orthonormality of the first two rows in the rotation matrix, i.e. our goal is to have (or minimize):

$$r'_{11}r'_{21} + r'_{12}r'_{22} + r'_{13}r'_{23} = 0$$

To accomplish this, we will use the idea of "squaring up".

1.2.4 "Squaring Up"

Squaring up is one method for ensuring that vectors are numerically/approximately orthogonal to one another. This technique answers the question of "Given a set of orthogonal vectors, what is the nearest set of orthogonal vectors?"

The figure below highlights that given vectors $\mathbf{a}$ and $\mathbf{b}$, we can find the nearest set of rotated vectors $\mathbf{a}'$ and $\mathbf{b}'$ using the following, which states that adjustments to each vector in the pair are made in the direction of the other vector:

$$\begin{aligned}\mathbf{a}' &= \mathbf{a} + k\mathbf{b} \\ \mathbf{b}' &= \mathbf{b} + k\mathbf{a}\end{aligned}$$

Since we want this dot product to be zero, we can solve for the scale k by taking the dot product of $\mathbf{a}'$ and $\mathbf{b}'$ and setting it equal to zero:

$$\begin{aligned}\mathbf{a}' \cdot \mathbf{b}' &= (\mathbf{a} + k\mathbf{b})(\mathbf{b} + k\mathbf{a}) = 0 \\ &= \mathbf{a} \cdot \mathbf{b} + (\mathbf{a} \cdot \mathbf{a} + \mathbf{b} \cdot \mathbf{b})k + k^2 + \mathbf{a} \cdot \mathbf{b} = 0\end{aligned}$$

This equation produces a quadratic equation in k, but unfortunately, as we approach our solution point, this sets the second and third terms, which k depends on, to values near zero, creating numerical instability and consequently making this a poor approach for solving for k. A better approach is to use the approximation:

$$k \approx -\left(\frac{\mathbf{a} \cdot \mathbf{b}}{2}\right)$$

It is much easier to solve for this and iterate over it a couple of times. I.e. instead of using the approach above, we use:

$$x = \frac{2c}{-b \pm \sqrt{b^2 - 4ac}}$$

Where we note that the "standard" quadratic solution is given by:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

It is important to consider both of these approaches because of floating-point precision accuracy - in any case, as we approach the solution point, one of the two forms of these solutions will not perform well, and the other form will perform substantially better.

With this, we are able to iteratively solve for k until our two vectors have a dot product approximately equal to zero. Once this dot product is approximately zero, we are done! We have enforced orthonormality of the first two rows of the rotation matrix, which allows us to find an orthonormal third row of the rotation matrix by taking a cross product of the first two rows. Therefore, we then have an orthonormal rotation matrix.

1.2.5 Planar Target

Planar target is an example calibration object that possesses desirable properties, namely, as we will see below, that it reduces the number of correspondences we need from 7 to 5. Additionally, planar targets are generally easy to make, store, and allow for high camera calibration accuracy. They are typically mounted on the side of wheels so that they can be rotated. They are typically given analytically by the plane:

$$Z_s = 0, \text{ i.e. the XY-plane where Z is equal to 0.}$$

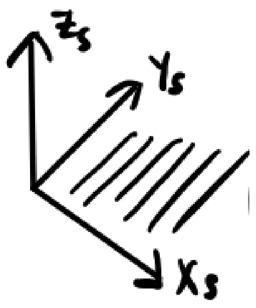


Figure 8: A geometric view of a planar target, which is a calibration target whose shape is geometrically described by a plane.

As a result of this, we no longer need to determine the rotation matrix coefficients r_{13}, r_{23}, and r_{33} (i.e. the third column of the rotation matrix, which determines how Z_s affects X_c, Y_c, and Z_c in the rotated coordinate system. Mathematically:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X_s \\ Y_s \\ 0 \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

As a result of this, our equations become:

$$\frac{x_I - x_O}{f} = \frac{X_c}{Z_c} = \frac{r_{11}X_s + r_{12}Y_s + t_x}{r_{31}X_s + r_{32}Y_s + t_z}$$

$$\frac{y_I - y_O}{f} = \frac{Y_c}{Z_c} = \frac{r_{21}X_s + r_{22}Y_s + t_y}{r_{31}X_s + r_{32}Y_s + t_z}$$

And, as we saw before, dividing these equations by one another yields a simplified form relative to our last division for the general case above:

$$\frac{x_I - x_O}{y_I - y_0} = \frac{\frac{r_{11}X_s + r_{12}Y_s + t_x}{r_{31}X_s + r_{32}Y_s + t_z}}{\frac{r_{21}X_s + r_{22}Y_s + t_y}{r_{31}X_s + r_{32}Y_s + t_z}}$$

$$= \frac{r_{11}X_s + r_{12}Y_s + t_x}{r_{21}X_s + r_{22}Y_s + t_y}$$

As we did before for the general case, cross-multiplying and rearranging gives:

$$(X_S y_I')r_{11} + (Y_S y_I')r_{12} + y_I' t_x - (X_S x_I')r_{21} - (Y_S x_I')r_{22} - x_I' t_y = 0$$

Now, rather than having 8 unknowns, we have 6: $r_{11}, r_{12}, r_{21}, r_{22}, t_x, t_y$. Because this is also a homogeneous equation, we again can account for scale factor ambiguity by setting one of DOF/parameters to a fixed value. This in turn reduces the DOF from 6 to 5, which means we only need 5 correspondences to solve for now (compared to 7 in the general case). As before, if we have more than 5 correspondences, this is still desirable as it will reduce estimation error and prevent overfitting.

One potential issue: if the parameter we fix, e.g. t_y, evaluates to 0 in our solution, this can produce large and unstable value estimates of other parameters (since we have to scale the parameters according to the fixed value). If the **fixed parameter** is close to 0, then we should fix a different parameter.

1.2.6 Aspect Ratio

This is no longer a common issue in machine vision now, but when aspect ratios were set differently in analog with photo-diode arrays for stepping in the x and y directions, this required setting an additional parameter to analytically describe the scale of x relative to y, and this parameter cannot be solved without a planar target.

1.2.7 Solving for t_z and f

Now that we have solved for our other parameters, we can also solve for t_z and f:

- These unknowns do not appear in other equations, but we can use our estimates from our other parameters to solve for t_z and f.
- We can again use the same equations that combine interior and exterior orientation:

$$(r_{11}X_s + r_{12}Y_s + r_{13}Z_s + t_x)f - x_I' t_z = (r_{31}X_s + r_{32}Y_s + r_{33}Z_s)x_I'$$
$$(r_{21}X_s + r_{22}Y_s + r_{23}Z_s + t_y)f - y_I' t_z = (r_{31}X_s + r_{32}Y_s + r_{33}Z_s)y_I'$$

  Since we only have two unknowns (since we have found estimates for our other unknowns at this point), this now becomes a much simpler problem to solve. With just **1 correspondence**, we get both of the **2 equations** above, which is sufficient to solve for our **2 unknowns** that remain. However, as we have seen with other cases, using more correspondences still remains desirable, and can be solved via least squares to increase robustness and prevent overfitting.

- One problem with this approach: We need to have **variation in depth**. Recall that perspective projection has multiplication by f and division by Z; therefore, doubling f and Z results in no change, which means we can only determine t_z and f as a ratio $\frac{f}{t_z}$, rather than separately. To remove this consequence of scale ambiguity, we need **variation in depth**, i.e. the calibration object, such as a plane, **cannot be orthogonal to the optical axis**.

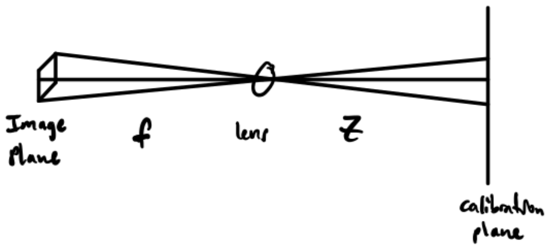


Figure 9: In order to obtain f and t_z separately, and not just up to a ratio, it is crucial that the calibration object does not lie completely orthogonal to the optical axis - else, these parameters exhibit high noise sensitivity and we can determine them separately.

It turns out this problem comes up in wheel alignment - if machine vision is used for wheel alignment, the calibration object will be placed at an angle of 45 or 60 degrees relative to the optical axis.

1.2.8 Wrapping it Up: Solving for Principal Point and Radial Distortion

To complete our joint interior/exterior orientation camera calibration problem, we need to solve for **principal point** and **radial distortion**.

As it turns out, there is unfortunately no closed-form solution for finding the principal point. Rather, we **minimize image error** using nonlinear optimization (e.g. Levenberg-Marquadt or Gauss-Newton). This error is the 2D error in the (x, y) image plane that describes the error between predicted and observed points in the image plane:

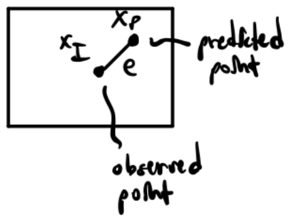


Figure 10: Image error that we minimize to solve for our principal point and radial distortion. We can formulate the total error from a set of observations as the squared errors from each correspondence. Note that x_I refers to the observed point, and x_P refers to the predicted point after applying our parameter estimate transformations to a point on the calibration object corresponding to an observed point.

Ideally, this error is zero after we have found an optimal set of parameters, i.e. for each i:

$$x_{I_i} - x_{P_i} = 0$$
$$y_{I_i} - y_{P_i} = 0$$

Where do x_{P_i} and y_{P_i} come from? They come from applying:

- **Rotation matrix estimate** ($\mathbf{R}$)
- **Translation vector estimate** ($\mathbf{t}$)
- **Principal Point Estimate** (x_0, y_0)
- **Radial Distortion Polynomial** $(k_1, k_2, \cdots)$

Using these equations, we can take a sum of squared error terms and solve for our parameters using nonlinear optimization with Levenberg-Marquadt (LM). This can be implemented using **LMdif** in the **MINPACK** package (Fortran), as well as other packages in other languages. Recall that LM performs nonlinear optimization with L2 regularization.

One small problem with this approach: LM only works for unconstrained optimization, which is not the case when we only use orthonormal rotation matrices. To overcome this, **Tsai's Calibration Method** used **Euler Angles**, and we can use either **Gibb's vector** or **unit quaternions**. Comparing/contrasting each of these 3 rotation representations to solve this problem:

- **Euler Angles**: This representation is non-redundant, but "blows up" if we rotate through 180 degrees.
- **Gibb's Vector**: This representation is also non-redundant, but exhibits a singularity at $\theta = \pi$.
- **Unit Quaternions**: This representation is redundant but exhibits no singularities, and this redundancy can be mitigated by adding an additional equation to our system:

$$\overset{o}{q} \cdot \overset{o}{q} - 1 = 0$$

Finally, although LM optimization finds **local**, and not **global** extrema, we have already developed an initial estimate of our solution through the application of the combined interior/exterior orientation equations above, which ensures that extrema that LM optimization finds are indeed not just locally optimal, but globally optimal.

1.2.9 Noise Sensitivity/Noise Gain of Approach

As we have asked for other systems, it is important to consider how robust this approach is to noise and perturbations. A few notes/concluding thoughts:

- As we saw above, this approach does not produce good estimates for our parameters t_z and f if the calibration object is orthogonal to the optical axis. In addition to not being able to separate the estimates for these parameters, both of these parameters (because they are defined by a ratio) have a very high/infinite noise gain.
- Since these methods are largely based off of numerical, rather than analytical methods, we can best evaluate noise gain using **Monte Carlo** methods, in which we use noise that is well-defined by its statistical properties, e.g. zero-mean Gaussian noise with finite variance denoted by σ^2. We can analyze the estimated noise gain of the system by looking at the parameter distribution in the parameter space after applying noise to our observed values and making solution predictions.
- Higher-order coefficients of radial distortion are highly sensitive to noise/exhibit high noise gain.