CS231a - Computer Vision Cheatsheet

Tim Reinhart - rtim@stanford.edu

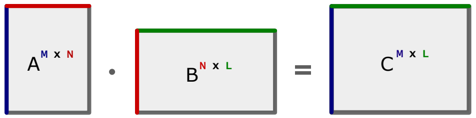
Version: May 6, 2024

Matrices

Matrix Multiplication

Matrices can be multiplied with each other in the following manner:

$$\boxed{A \cdot B = C \quad \Rightarrow \quad c_{ik} = \sum_{j=1}^{n} a_{ij} \cdot b_{jk}}$$



Associative & Distributive Laws:

$$(A \cdot B) \cdot C = A \cdot (B \cdot C)$$
$$(A + B) \cdot C = A \cdot C + B \cdot C$$
$$A \cdot (C + D) = A \cdot C + A \cdot D$$

Warning! The commutative law does not apply! Generally, $A \cdot B \neq B \cdot A$.

Transpose

The transpose of a matrix is obtained by "mirroring" it along its diagonal.

Example: $$\begin{pmatrix} a & b \\ c & d \\ e & f \end{pmatrix}^T = \begin{pmatrix} a & c & e \\ b & d & f \end{pmatrix}$$

Calculation Rules:

$$(A + B)^T = A^T + B^T \qquad (A^T)^{-1} = (A^{-1})^T$$
$$(A \cdot B)^T = B^T \cdot A^T \qquad rank(A^T) = rank(A)$$
$$(c \cdot A)^T = c \cdot A^T \qquad det(A^T) = det(A)$$
$$(A^T)^T = A \qquad eig(A^T) = eig(A)$$

Inverse

The inverse A^{-1} of A reverses a multiplication with A. When you multiply A with A^{-1}, you get the identity matrix.

Properties:

- Only square matrices can be invertible.
- An invertible matrix is called **regular**, a non-invertible one **singular**.
- The inverse is unique.
- A is invertible if and only if A has full rank.
- A is invertible if and only if A^T is invertible.
- A is symmetric if and only if A^{-1} is symmetric.
- A is a triangular matrix if and only if A^{-1} is a triangular matrix.
- A is invertible if and only if $\det(A) \neq 0$.
- A is invertible if and only if no eigenvalue $\lambda = 0$.
- A and B are invertible implies AB is invertible.

Calculation rules:

$$I^{-1} = I \qquad (A^T)^{-1} = (A^{-1})^T$$
$$(A^{-1})^{-1} = A \qquad rang(A^{-1}) = rang(A)$$
$$(A^k)^{-1} = (A^{-1})^k \qquad det(A^{-1}) = det(A)^{-1}$$
$$(c \cdot A)^{-1} = c^{-1} \cdot A^{-1} \qquad eig(A^{-1}) = eig(A)^{-1}$$
$$(A \cdot B)^{-1} = B^{-1} \cdot A^{-1}$$

Eigenvalues and Eigenvectors

$$\boxed{\text{Eigenvalues of A: } \det(A - \lambda \cdot I) = 0}$$

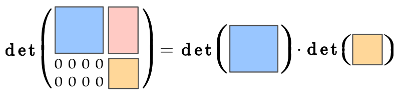
Verify Computation

- Trace(A) $= a_{11} + a_{22} + \cdots + a_{nn} = \sum \lambda_i$
- det(A) = product of λ_i

Eigenvectors: Kernel of the matrix $A - \lambda_i \cdot I$, where λ_i is the eigenvalue corresponding to the eigenvector.

Determinant

Block Sentence for Determinant Computation



Basic Spaces of a Matrix

Null Space (Kernel): Set of all vectors v such that $Av = 0$.
Column Space (Range): Set of all vectors that can be expressed as Av for some v.
Row Space: Set of all vectors that can be expressed as vA for some v, equivalent to the column space of A^T.

Special Matrices

Identity Matrix I: Diagonal matrix with ones on the diagonal.
Triangular Matrix: All elements above (upper) or below (lower) the diagonal are zero.
Orthogonal Matrix Q: Satisfies $Q^T Q = QQ^T = I$.

QR Decomposition

Decomposition of a matrix A into an orthogonal matrix Q and an upper triangular matrix R:

$$A = Q \cdot R$$

Singular Value Decomposition (SVD)

Decomposition of a matrix A into U, Σ, and V^T where U and V are orthogonal, and Σ contains singular values:

$$\mathbf{M} = \mathbf{U} \cdot \mathbf{\Sigma} \cdot \mathbf{V}^T$$

- The diagonal entries $\sigma_i = \Sigma_{ii}$ of $\mathbf{\Sigma}$ are uniquely determined by $\mathbf{M}$ and are known as the singular values of $\mathbf{M}$.
- The number of non-zero singular values is equal to the rank of $\mathbf{M}$.
- The columns of $\mathbf{U}$ and the columns of $\mathbf{V}$ are called left-singular vectors and right-singular vectors respectively.
- The columns of $\mathbf{U}$ and the columns of $\mathbf{V}$ form two sets of orthonormal bases $\mathbf{u}_1, \ldots, \mathbf{u}_m$ and $\mathbf{v}_1, \ldots, \mathbf{v}_n$
- If they are sorted so that the singular values σ_i with value zero are all in the highest-numbered columns (or rows), the singular value decomposition can be written as

  $$\mathbf{M} = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^*,$$

  where $r \leqslant \min\{m, n\}$ is the rank of $\mathbf{M}$

Relation to the Four Fundamental Subspaces

- The first r columns of $\mathbf{U}$ are a basis of the column space of $\mathbf{M}$.
- The last $m - r$ columns of $\mathbf{U}$ are a basis of the null space of $\mathbf{M}^T$.
- The first r columns of $\mathbf{V}$ are a basis of the column space of $\mathbf{M}^T$ (the row space of $\mathbf{M}$ in the real case).
- The last $n - r$ columns of $\mathbf{V}$ are a basis of the null space of $\mathbf{M}$.

Solving Homogeneous Linear Equations

- Equation form: $\mathbf{Ax} = \mathbf{0}$.
- Goal: Find non-zero $\mathbf{x}$ that satisfies the equation.
- $\mathbf{x}$ is a right null vector of $\mathbf{A}$.
- Characterization: $\mathbf{x}$ is a right-singular vector for a zero singular value of $\mathbf{A}$.
- If $\mathbf{A}$ has no zero singular values, there is no non-zero solution. ($\rightarrow$ full rank)
- Multiple zero singular values allow for solutions that are linear combinations of corresponding right-singular vectors.
- Left null vector: $\mathbf{x}^*\mathbf{A} = \mathbf{0}$ where $\mathbf{x}^*$ is the conjugate transpose of $\mathbf{x}$.

Total Least Squares Minimization

- Objective: Minimize the 2-norm of $\mathbf{Ax}$ with $\|\mathbf{x}\| = 1$.
- Solution: Right-singular vector of $\mathbf{A}$ corresponding to the smallest singular value.

Range, Null Space, and Rank

- SVD gives explicit representation of matrix's range and null space.
- Null space: Spanned by right-singular vectors for zero singular values of $\mathbf{M}$.
- Range: Spanned by left-singular vectors for non-zero singular values of $\mathbf{M}$.
- Rank: Number of non-zero singular values, matching non-zero diagonal elements in $\mathbf{\Sigma}$.
- Effective rank: Determined by singular values, considering numerical errors that might lead to small non-zero singular values.

Low-Rank Matrix Approximation

- Problem: Approximate $\mathbf{M}$ with $\tilde{\mathbf{M}}$ of specific rank r, minimizing the Frobenius norm of their difference.
- Solution via SVD:

  $$\tilde{\mathbf{M}} = \mathbf{U}\tilde{\mathbf{\Sigma}}\mathbf{V}^*,$$

  where $\tilde{\mathbf{\Sigma}}$ has only the r largest singular values, others set to zero.

Homogeneous Coordinates

- Definition: Extends traditional coordinates by adding an extra dimension.
- Example: Point (x, y) in Cartesian coordinates becomes (wx, wy, w) in homogeneous coordinates, where $w \neq 0$.

- Applications: Used in computer graphics and computer vision for handling affine and perspective transformations.
- Conversion: From homogeneous to Cartesian coordinates by dividing each component by the last coordinate (if non-zero).
- **Point at infinity:** $x_\infty = \begin{bmatrix} x \\ y \\ 0 \end{bmatrix}$

Types of Geometric Transformations

This section outlines four major types of geometric transformations, each preserving different properties of shapes and spaces. Each type includes an example of how a point in homogeneous coordinates is transformed.

Isometric Transformations

- Property Preserved: Distances between points.
- Also known as rigid transformations, include rotations, translations, and reflections.
- Example:

$$p' = \begin{bmatrix} \cos\theta & -\sin\theta & t_x \\ \sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

for rotation by θ and translation by (t_x, t_y).

Similarity Transformations

- Property Preserved: Shape of figures → ratio of lengths and angles
- Encompasses isometric transformations along with scaling; angles and relative proportions are maintained.
- Example:

$$\begin{bmatrix} s\cos\theta & -s\sin\theta & t_x \\ s\sin\theta & s\cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} SR & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

where s is the scale factor, $S = \begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix}$ and θ the rotation angle.

Affine Transformations

- Property Preserved: Points, straight lines, parallelism of lines.
- Includes translations, scaling, rotations, and shearing; more general than similarity transformations.
- Maps **points at infinity** to points at infinity!
- Example:

$$p' = \begin{bmatrix} a & b & t_x \\ c & d & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

where matrix elements $a, b, c,$ and d define the linear transformation.

Projective Transformations / Homographies

- Property Preserved: Collinearity of points (lines are preserved).
- Can map parallel lines to a converging point, typically used in perspective projections.
- Generally maps **points at infinity** to points no longer at infinity!
- Example:

$$p' = \begin{bmatrix} a & b & t_x \\ c & d & t_y \\ e & f & g \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

where the elements $e, f,$ and g introduce the projective distortion.

Geometry

2D Lines with homogeneous coordinates

Given $x = [x\,y\,1]^T$ and $l = [a\,b\,c]^T$, then the line eqation is given by:

$$l^T x = 0$$

Slope: $-\frac{a}{b}$, y-intercept: $-\frac{c}{b}$

Finding l Given points x_1 and x_2, we find l by:

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \times \begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix}$$

Finding intersection Given l_1 and l_2, we find x by:

$$\begin{bmatrix} wx \\ xy \\ w \end{bmatrix} = \begin{bmatrix} a_1 \\ b_1 \\ c_1 \end{bmatrix} \times \begin{bmatrix} a_2 \\ b_2 \\ c_2 \end{bmatrix}$$

Parallel Lines For parallel lines l and l', we have the same slope, i.e. $\frac{a}{b} = \frac{a'}{b'}$. Their intersection is given by:

$$l \times l' \propto \begin{bmatrix} b \\ -a \\ 0 \end{bmatrix} = x_\infty$$

All parallel lines (same slope) pass through the same point at infinity (**ideal point**).

Camera Matrix Model

Parameters:

- Focal length f
- Translation c_x, c_y: Image plane and digital image coordinates can differ by a translation
- Change of units: k and l change the units from cm to pixels for each axis of the image plane. $\alpha = f \cdot k$ and $\beta = f \cdot l$

Mapping from 3D point P in camera reference frame to image coordinates P':

$$P' = \begin{bmatrix} \alpha\frac{x}{z} + c_x \\ \beta\frac{y}{z} + c_y \end{bmatrix} P$$

Or in homogeneous coordinates:

$$P' = \begin{bmatrix} \alpha & 0 & c_x & 0 \\ 0 & \beta & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \underbrace{\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}}_{=P}$$

$$= \underbrace{\begin{bmatrix} \alpha & 0 & c_x \\ 0 & \beta & c_y \\ 0 & 0 & 1 \end{bmatrix}}_{=K} \begin{bmatrix} I_{3\times 3} & 0 \end{bmatrix} P$$

If the world reference frame is different from camera reference frame, we can compute the camera coordinates from world point P_w by:

$$P = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} P_w$$

Where R is the rotation matrix and T is the translation vector (**extrinsic parameters**). Full camera model:

$$P = K \begin{bmatrix} R & T \end{bmatrix} P_w = M P_w$$

Single View Metrology

Vanishing Points and Lines

Given the direction of a set of parallel 3D lines $d = [a\,b\,c]^T$. The vanishing point v is given by:

$$v = Kd$$

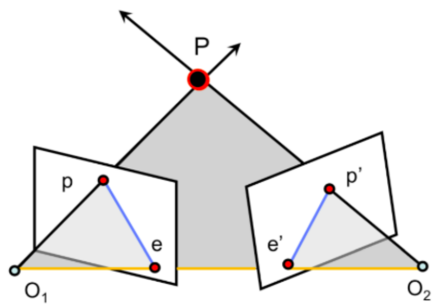
The direction can be computed by:

$$d = \frac{K^{-1}v}{||K^{-1}v||}$$

The horizon line l_{horiz} is the projective transformation of a line at infinity l_∞(line that passes through a set of points at infinity). The normal n of a plane in 3D can be computed using

$$n = K^T l_{\text{horiz}}$$

Epipolar Geometry



- Epipoles: e, e'
- Epipolar Lines: $\vec{pe}, \vec{p'e'}$

If you assume that the world reference system is associated with the first camera, then the camera projection matrices are:

$$M = K \begin{bmatrix} I_{3\times 3} & 0 \end{bmatrix} \quad M' = K' \begin{bmatrix} R^T & -R^T T \end{bmatrix}$$

Essential Matrix

Given the rotation R and translation T from the first camera reference frame to the second. The location of p' in the first camera reference frame is:

$$p'_1 = Rp'_2 + T$$

The Essential Matrix E is defined for **canonical cameras**, $K = K' = I$. $E \in \mathbb{R}^{3\times 3}$ has 5 DoF and is defined as:

$$E = [T_\times]R$$

And the **epipolar constraint** is

$$p^T E p' = 0$$

Epipolar Lines:

- in image plane of camera **2**: $l' = E^T p$
- in image plane of camera **1**: $l = Ep'$

Dot product with epipoles: $E^T e = Ee' = 0$

Fundamental Matrix

For **non canonical cameras**. We must define the location of p in the camera reference frame:

$$p_c = K^{-1}p \quad p'_c = K'^{-1}p'$$

The Fundamental Matrix $F \in \mathbb{R}^{3\times 3}$ has 7 DoF and is defined as:

$$F = K'^{-T}[T_\times]RK^{-1}$$

Epipolar Lines:

- in image plane of camera **2**: $l' = F^T p$
- in image plane of camera **1**: $l = Fp'$
- The epipole e lies at the intersection of all epipolar lines l.

Other properties:

- F has **rank 2**
- if x and x' are corresponding image points, then $x'^T F x = 0$
- Epipoles: $Fe = 0$ and $F^T e' = 0$

Normalized Eight-Point Algorithm

- W is ill-conditioned for SVD, due to the large image coordinate values in modern cameras. If the image correspondences are all in a small region of the image, then all p_i and p'_i will be very similar, therefore one singular value of W will be very large and the others very small. For SVD to work properly, only one singular value should be near zero.
- Solution: Apply a transformation and scaling on the image coordinates.
- Origin should be located at the centroid of image points (translation), and the mean squared distance of the transformed image points should be 2 pixels.

For **each camera** we define a transformation T. The scaling factor s is given by:

$$s = \sqrt{\frac{2N}{\sum_i^N ||x_i - \bar{x}||}}$$

where $\bar{x} = \frac{1}{N}\sum_i^N x_i$. Then the transformation is given by:

$$T = \begin{bmatrix} s & 0 & -s\cdot\bar{x}_1 \\ 0 & s & -s\cdot\bar{x}_2 \\ 0 & 0 & 1 \end{bmatrix}$$

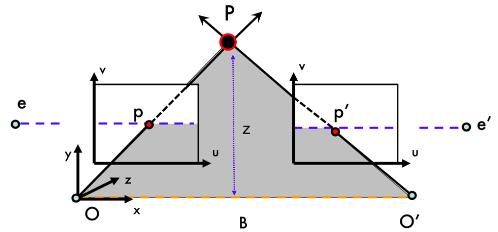
The points are normalized by:

$$q_i = Tp_i \quad q'_i = T'p'_i$$

And the final Fundamental Matrix is given by:

$$F = T'^T F_q T$$

Disparity



$$p = \begin{bmatrix} p_u \\ p_v \\ 1 \end{bmatrix} \quad p' = \begin{bmatrix} p'_u \\ p'_v \\ 1 \end{bmatrix}$$

Disparity d is defined as:

$$d = p_u - p'_u \propto \frac{B\cdot f}{z}$$

Structure from Motion (SfM)

- m cameras with camera matrix M_i
- n 3D point measurements X_j
- location x_{ij} is the projection of X_j in the image plane of camera i

Goal is to recover the m projection matrices M_i (motion) and the n 3D points X_j (structure).

Tomasi and Kanade Factorization Method

Solves the affine structure from motion problem: Assuming a weak perspective transformation M.

Step 1: Data Centering. For each image i, subtract the centroid $\bar{x}_i$ from image coordinates.

Step 2: Build a measurement matrix $D \in \mathbb{R}^{2m\times n}$. Use the SVD of $D = U\Sigma V^T$. We know that $rank(D) = 3$, therefore only three singular values will be nonzero.

Robust Factorization: $M = U_3\sqrt{\Sigma_3}$ $S = \sqrt{\Sigma_3}V_3^T$
Where $M \in \mathbb{R}^{2m\times 3}$ and $S \in \mathbb{R}^{3\times n}$

$$M = \begin{bmatrix} A_1 \\ \vdots \\ A_m \end{bmatrix} \quad S = \begin{bmatrix} X_1 & \dots & X_n \end{bmatrix}$$

Where A_i uses the affine camera model:

$$x = \begin{bmatrix} m_1 X \\ m_2 X \end{bmatrix} = \begin{bmatrix} A & b \end{bmatrix} X$$

Ambiguities in Reconstruction

Any invertible matrix $A \in \mathbb{R}^{3\times 3}$ may be inserted into the Factorization:

$$D = MS = MAA^{-1}S = (MA)(A^{-1}S) = M'S'$$

Therefore, the solution has **affine ambiguity**, which means that parallelism is preserved, but the metric scale is unknown.

Similarity ambiguity: Occurs when a Reconstruction is correct up to a similarity transform - rotation, translation, scaling. Also known as metric Reconstruction. For calibrated cameras, this is the only ambiguity.

Perspective SfM

In the general case, M has 11 DoF, as it is defined up to scale.

Fitting and Matching

Least Squares Method

Model 1: $y_i - mx_i - b = 0$ Error:

$$E = \sum_i^n (y_i - mx_i - b)^2$$

Solution:

$$h = \begin{bmatrix} m \\ b \end{bmatrix} = (X^T X)^{-1} X^T Y$$

Where:

$$X = \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix} \quad Y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}$$

Issues: Fails completely for vertical line!

Model 2: $ax_i + by_i + d = 0$
Distance between points $(x, y, 1)$ and line (a, b, d) is given by $ax + by = d$. Find line to minimize sum of squared perpendicular distances:

$$E = \sum_i^n (ax_i + by_i + d)^2$$

Find h s.t. $Ah = 0$. Minimize $||Ah||$ subject to $||h|| = 1$. SVD, h is the last column of V.
Least squares is **not Robust** to outliers!

RANSAC

Robust to outliers and missing data!

Steps for Eight-Point Algorithm

1. Randomly select the minimum number of points needed to fit a model. Line: 2, 8PA: 8, Homography: 4 correspondences
2. Fit model to random sample set
3. Use model to compute the inlier set from the entire dataset

Repeat for M iterations, maximize the size of the inlier set

Volumetric Stereo

Space Carving

- Requires knowing the camera intrinsics and extrinsics
- Produces conservative 3D Reconstructions (no smaller than the actual 3D shape)
- Method to find the silhouette of the object in each view is needed
- The result is voxels instead of a point cloud, and the degree of accuracy depends on the number of voxels we choose to use.

Steps

1. Define a working volume, e.g. entire space enclosed by cameras
2. Divide volume into small units: **voxels**
3. Project each voxel into each of the views
4. If the voxel is not contained by the silhouette in a view, it is discarded.

Limitations

- Scales linearly with number of voxel, which increases cubically.
- Accuracy depends heavily on the silhouette.
- Incapable of modeling certain concavities of an object.

Shadow Carving

- uses self shadows: shadows that an object projects on itself
- can estimate concavities better than space carving
- produces a conservative volume estimate

Steps

1. Begins with initial voxel grid
2. In each view, each light in the array is turned on and off
3. Identify the shadow in the image plane
4. find voxels on the surface that are in the visual cone of the Shadow
5. Use surface voxels allow us to make new visual cone
6. a voxel that is part of both visual cones cannot be part of the object

Limitations

- takes $n + 1$ times longer than space carving (n is nr. of lights)
- cannot handle cases where

Voxel coloring

- Uses color consistency instead of contour consistency in space carving
- Gives a colored Reconstruction
- Object must be Lambertian: the perceived luminance of any part of the objcet does not change with viewpoint location or produces
- Voxels need to be processed in a certain order $\rightarrow$ cameras cannot be in certain locations