Introduction to Computer Vision
(to appear as Fundamentals of Image Processing and Interpretation)

Shaifali Parashar (CNRS Research Scientist)

Computer Vision (Image interpretation)

To interpret the world seen through cameras

Observations:

Semantic

- green, blue, white, gray, orange, red (colors)
- window, cars, wheels, trees, grass, sky, cloud, shadow
- garden, building, parking with cars
  (edges+contours+ image features + learning to recognise geometric features and patterns)

Computer Vision (Image interpretation)

To interpret the world seen through cameras

Observations:

Semantic

- green, blue, white, gray, orange, red (colors)
- window, cars, wheels, trees, grass, sky, cloud, shadow
- garden, building, parking with cars
  (edges+contours+ image features + learning to recognise geometric features and patterns)

Metric

- distance (multiple-view geometry + 3D reconstruction)

Computer Vision (Image interpretation)

To interpret the world seen through cameras

Observations:

Semantic

- green, blue, white, gray, orange, red (colors)
- window, cars, wheels, trees, grass, sky, cloud, shadow
- garden, building, parking with cars
  (edges+contours+ image features + learning to recognise geometric features and patterns)

Metric

- distance (multiple-view geometry + 3D reconstruction)

Interpretation (by a non-familiar person):
A building with a parking lot and green spaces, less than 50 metres away

Computer Vision (Image interpretation)

To interpret the world seen through cameras

Observations:

Semantic

- green, blue, white, gray, orange, red (colors)
- window, cars, wheels, trees, grass, sky, cloud, shadow
- garden, building, parking with cars
  (edges+contours+ image features + learning to recognise geometric features and patterns)

Metric

- distance (multiple-view geometry + 3D reconstruction)

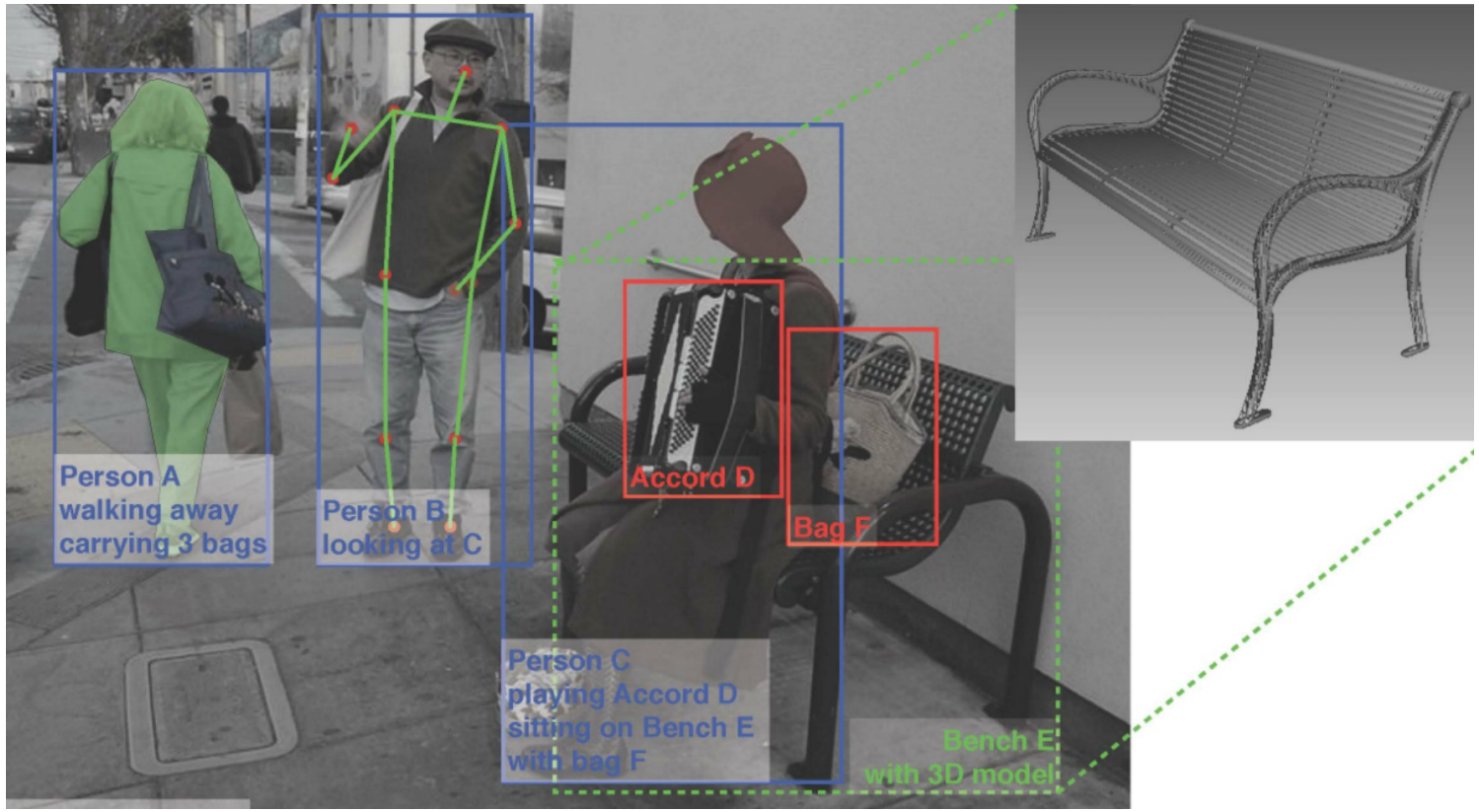
Interpretation (by a non-familiar person):
A building with a parking lot and green spaces, less than 50 metres away

Interpretation (by a familiar person):
A view of Nautibus from the Lyon 1 tram stop

Applications: Recognition, Reorganisation and Reconstruction

What do images tell?

- Objects
- Activities
- Scenes
- Locations
- Text/writing
- Faces
- Gestures
- Motions
- Emotions …



1. Mathematical Fundamentals

Matrix analysis

Vector

collection of ordered numbers that represent something: pixel brightness, functions describing location, motion, etc.

$\mathbf{v} \in \mathbb{R}^{N \times 1}$ column vector (default)

$\mathbf{v}^\top \in \mathbb{R}^{1 \times N}$ row vector

$$\mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \\ \dots \\ v_n \end{pmatrix} = \begin{pmatrix} v_1 & v_2 & \dots & v_n \end{pmatrix}^\top$$

T = transpose flips the vector, row to column and vice versa

```
>> v = [1;2;3]

v =

     1
     2
     3

>> v'

ans =

     1     2     3
```

In matlab, ' is the transpose operator

Vector

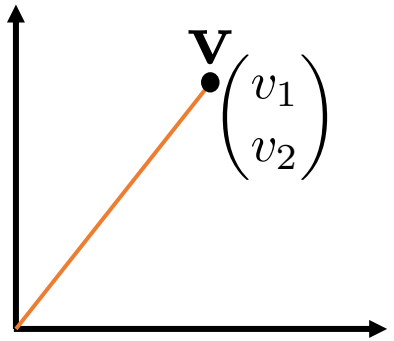
collection of ordered numbers that represent something: pixel brightness, functions describing location, motion, etc.

$\mathbf{v} \in \mathbb{R}^{N \times 1}$ column vector (default)

$\mathbf{v}^\top \in \mathbb{R}^{1 \times N}$ row vector

$$\mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \\ \dots \\ v_n \end{pmatrix} = \begin{pmatrix} v_1 & v_2 & \dots & v_n \end{pmatrix}^\top$$

T = transpose flips the vector, row to column and vice versa



Norm/length of vector

$$\|\mathbf{v}\| = \sqrt{v_1^2 + v_2^2}$$

Length = 1 -> unit (or normalised) vector

Location or offset (n-dimensional)

Point is vector through origin

Vector

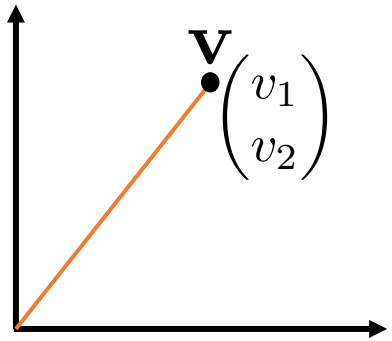
collection of ordered numbers that represent something: pixel brightness, functions describing location, motion, etc.

$\mathbf{v} \in \mathbb{R}^{N \times 1}$ column vector (default)

$\mathbf{v}^\top \in \mathbb{R}^{1 \times N}$ row vector

$$\mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \\ \dots \\ v_n \end{pmatrix} = \begin{pmatrix} v_1 & v_2 & \dots & v_n \end{pmatrix}^\top$$

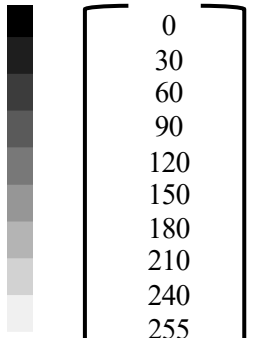
T = transpose flips the vector, row to column and vice versa



Norm/length of vector

$$\|\mathbf{v}\| = \sqrt{v_1^2 + v_2^2}$$

Length = 1 -> unit (or normalised) vector



Location or offset (n-dimensional)
Point is vector through origin

Data values, no geometric interpretation

Matrix

$\mathbf{M} \in \mathbb{R}^{N \times P}$ matrix with N rows and P columns

m_{ij} - elements of the matrix, i[th] row and j[th] column

$$\mathbf{M} = \begin{pmatrix} m_{11} & m_{12} & \dots & m_{1P} \\ m_{21} & m_{22} & \dots & m_{2P} \\ \vdots & \vdots & \vdots & \vdots \\ m_{N1} & m_{N2} & \dots & m_{NP} \end{pmatrix}$$

x

y

Note that when matrix represent images, their coordinates [r,c] are not cartesian, [y=-1,x=1] $=m_{11}$

Matrix

$\mathbf{M} \in \mathbb{R}^{N \times P}$ matrix with N rows and P columns

m_{ij} - elements of the matrix, i^{th} row and j^{th} column

Transpose= flip a matrix along diagonal

exchange between rows and columns

$$\begin{pmatrix} 0 & 1 \\ 2 & 3 \\ 4 & 5 \end{pmatrix}^{\top} = \begin{pmatrix} 0 & 2 & 4 \\ 1 & 3 & 5 \end{pmatrix}$$

Trace: sum of diagonals $tr(\mathbf{M}) = \sum m_{ii}$

Norm (Frobenius norm): $\|\mathbf{M}\| = \sqrt{\sum_{i=1}^{M} \sum_{j=1}^{P} m_{ij}^2} = \sqrt{tr(\mathbf{M}^{\top}\mathbf{M})}$

```
>> m = randi(5,5,5)

m =

     1     5     4     1     3
     2     3     4     3     3
     4     3     2     4     1
     3     5     3     5     2
     2     2     1     1     1

>> trace(m)

ans =

    12

>> norm(m)

ans =

   14.3592

>> norm(m,'fro')

ans =

   15.0997

>> sqrt(trace(m'*m))

ans =

   15.0997

>>
```

Basic operations

Addition/Subtraction: possible only with matrices/vectors of same dimension or a scalar

Multiplication: With scalars, element-wise multiplication $3 \times \mathbf{M} \rightarrow 3 \times m_{ij}$

With Vectors/Matrix: possible when columns (c) in left matrix = rows (r) in right matrix

$$\mathbf{C} = \mathbf{AB} = \sum_{k=1}^{n} a_{ik} b_{kj}, \ \mathbf{A} \in \mathbb{R}^{M \times N}, \mathbf{B} \in \mathbb{R}^{N \times P}$$

Associative : $\mathbf{ABC} = \mathbf{A}(\mathbf{BC}) = (\mathbf{AB})\mathbf{C}$

Distributive: $\mathbf{A}(\mathbf{B} + \mathbf{C}) = \mathbf{AB} + \mathbf{BC}$

Non-commutative (in general): $\mathbf{AB} \neq \mathbf{BA}$

Special Products

Inner product (dot product) on vectors with same length:

$$\mathbf{v} \cdot \mathbf{w} = \mathbf{v}^\top \mathbf{w} = \sum_i v_i w_i = \|\mathbf{v}\| \|\mathbf{w}\| \cos\theta$$

If $\mathbf{w}$ is a unit vector, then $\mathbf{v} \cdot \mathbf{w}$ gives the length of $\mathbf{v}$ which lies in the direction of $\mathbf{w}$

Theta is the angle between vectors

Orthogonal vectors: $\mathbf{v} \cdot \mathbf{w} = 0 \quad \theta = 90°$

Cross producton vectors with same length: $\mathbf{v} \times \mathbf{w} = \|\mathbf{v}\| \|\mathbf{w}\| \sin\theta$

Zero for parallel vectors

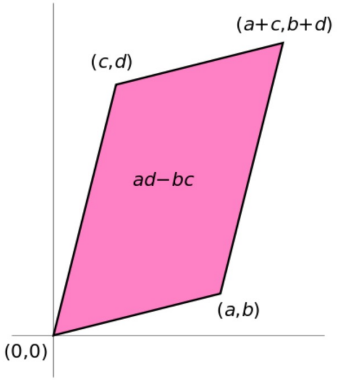
Outer product (exterior product) on vectors that can be of different length:

$$\mathbf{v} \otimes \mathbf{w} = \mathbf{v}\mathbf{w}^\top$$

Determinant

Determinant: Represents area (2D) and volume (3D) of the parallelogram described by the vectors in the rows of the matrix

$$\det \mathbf{M}_{2\times 2} = m_{11}m_{22} - m_{12}m_{21}$$



Area: ad-bc

Special property:

$$\det \mathbf{AB} = \det \mathbf{BA}$$
$$\det \mathbf{A}^\top = \det \mathbf{A}$$
$$\det \mathbf{A}^{-1} = \frac{1}{\det \mathbf{A}}$$

If $\det \mathbf{M} = 0$, $\mathbf{M}$ is singular -> no area/volume, non-invertible

Special matrices

$\mathbf{M} \in \mathbb{R}^{N \times P}$ If N=P, it is a square matrix

Identity matrix: $m_{ij} = 1 \quad i = j \quad$, 0 otherwise usually denoted as $\mathbf{I}$

$$\mathbf{AI} = \mathbf{IA} = \mathbf{A}$$

Diagonal matrix: $m_{ij} = d \quad i = j, \quad d \in \mathbb{R} \quad$, 0 otherwise $\mathbf{AM} = \mathbf{MA}$

diagonal elements can be different

Symmetric matrix: $m_{ij} = m_{ji} \qquad \mathbf{M} = \mathbf{M}^\top$

Skew-symmetric matrix: $m_{ij} = -m_{ji} \quad \mathbf{M} = -\mathbf{M}^\top$

Orthogonal matrix: $\mathbf{M}\mathbf{M}^\top = \mathbf{M}^\top\mathbf{M} = \mathbf{I}, \quad \det \mathbf{M} = 1$

Matrix Inverse (only for square matrices)

Given $\mathbf{M}$, $\mathbf{M}^{-1}$ is such that $\mathbf{M}\mathbf{M}^{-1} = \mathbf{M}^{-1}\mathbf{M} = \mathbf{I}$ $\quad \mathbf{M}^{-1} = \frac{1}{\det \mathbf{M}} adj\mathbf{M}$

$$\mathbf{M} = \begin{pmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{pmatrix}, \quad \mathbf{M}^{-1} = \frac{1}{\det \mathbf{M}} \begin{pmatrix} m_{22} & -m_{12} \\ -m_{21} & m_{22} \end{pmatrix}$$

If $\mathbf{M}$ is singular, $\mathbf{M}^{-1}$ does not exist.

$$(\mathbf{AB})^{-1} = \mathbf{B}^{-1}\mathbf{A}^{-1}$$
$$(\mathbf{M}^{-1})^{-1} = \mathbf{M}$$
$$(\mathbf{M}^{-1})^{\top} = (\mathbf{M}^{\top})^{-1} = \mathbf{M}^{-\top}$$

```
>> m = randi(5,2,2)

m =

     4     3
     2     1

>> det(m)

ans =

    -2

>> inv(m)

ans =

   -0.5000    1.5000
    1.0000   -2.0000

>> pinv(m)

ans =

   -0.5000    1.5000
    1.0000   -2.0000
```

Always use pseudoinverses for numerical computations

Matrix Rank

Linear dependence of vectors: If a vector can be expressed as a linear combination of other vectors, it is linear dependent on them otherwise it is linearly independent.

Given$(\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3)$ $\mathbf{v}_3 = \alpha_1 \mathbf{v}_1 + \alpha_2 \mathbf{v}_2$, $\mathbf{v}_3$ is linearly dependent on others

Matrix Rank

Linear dependence of vectors: If a vector can be expressed as a linear combination of other vectors, it is linear dependent on them otherwise it is linearly independent.

Given$(\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3)$ $\mathbf{v}_3 = \alpha_1 \mathbf{v}_1 + \alpha_2 \mathbf{v}_2$, $\mathbf{v}_3$ is linearly dependent on others

If all 3 vectors are orthogonal, they are linearly independent. In that case,

$$\alpha_1 \mathbf{v}_1 + \alpha_2 \mathbf{v}_2 + \alpha_3 \mathbf{v}_3 = 0 \iff \alpha_{1,2,3} = 0$$

Matrix Rank

Linear dependence of vectors: If a vector can be expressed as a linear combination of other vectors, it is linear dependent on them otherwise it is linearly independent.

Given$(\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3)$ $\mathbf{v}_3 = \alpha_1\mathbf{v}_1 + \alpha_2\mathbf{v}_2$, $\mathbf{v}_3$ is linearly dependent on others

If all 3 vectors are orthogonal, they are linearly independent. In that case,

$$\alpha_1\mathbf{v}_1 + \alpha_2\mathbf{v}_2 + \alpha_3\mathbf{v}_3 = 0 \iff \alpha_{1,2,3} = 0$$

(Column/row) Rank of a matrix: number of independent columns/rows in a matrix. Apparently, they are column rank = row rank

Full rank: for $\mathbf{M} \in \mathbb{R}^{M\times N}$, $rank(\mathbf{M}) \leq min(M, N)$, in case of equality, it is full rank

A singular matrix (non-invertble) is not full rank.

Change of Basis

Cartesian coordinates 2D: basis $\mathbf{B} = \begin{pmatrix} \mathbf{e}_1 & \mathbf{e}_2 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$

Given a point **x** = (1,1) in 2D, it is with respect to **B**

Suppose we change the basis, the new basis $\overline{\mathbf{B}} = \begin{pmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix}$

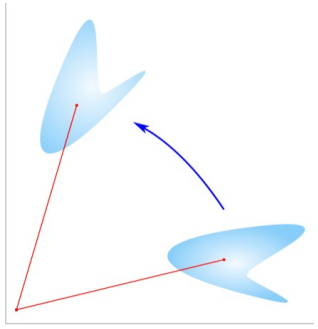
The transformed **x** according to this basis is

$$\overline{\mathbf{x}} = \overline{\mathbf{B}}^{-1}\mathbf{x}$$

Matrix Transformations: Rotation

In 2D: degrees of freedom = 1

$$\mathbf{R} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$



In 3D: degrees of freedom = 3

$$\mathbf{R_x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}$$

$$\mathbf{R_y} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

$$\mathbf{R_z} = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

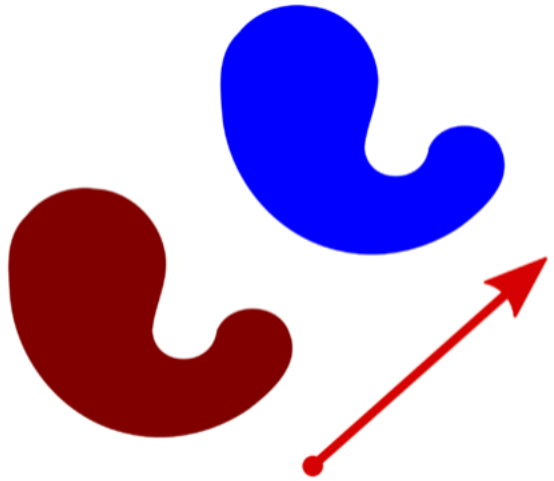
$$\mathbf{R} = \mathbf{R_z}\mathbf{R_y}\mathbf{R_x} = \begin{bmatrix} \cos\theta\cos\psi & -\cos\phi\sin\psi + \sin\phi\sin\theta\cos\psi & \sin\phi\sin\psi + \cos\phi\sin\theta\cos\psi \\ \cos\theta\sin\psi & \cos\phi\cos\psi + \sin\phi\sin\theta\sin\psi & -\sin\phi\cos\psi + \cos\phi\sin\theta\sin\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix}$$

Rotation is an orthogonal transformation, det(**R**) = 1
Reflection is like rotation, but det(**R**) = -1
Axis-angle representations: Euler angles, Quaternions, Rodrigue's formula

Matrix Transformations: Translation

In 2D: degrees of freedom = 2

In 3D: degrees of freedom = 3



Matrix Transformations: Scaling

In 2D: $\mathbf{S} = \begin{pmatrix} s_x & 0 \\ 0 & s_y \end{pmatrix}$

Isotropic: $s_x = s_y$

Anisotropic: $s_x \neq s_y$

degrees of freedom (isotropic): 1

degrees of freedom (anisotropic): 2

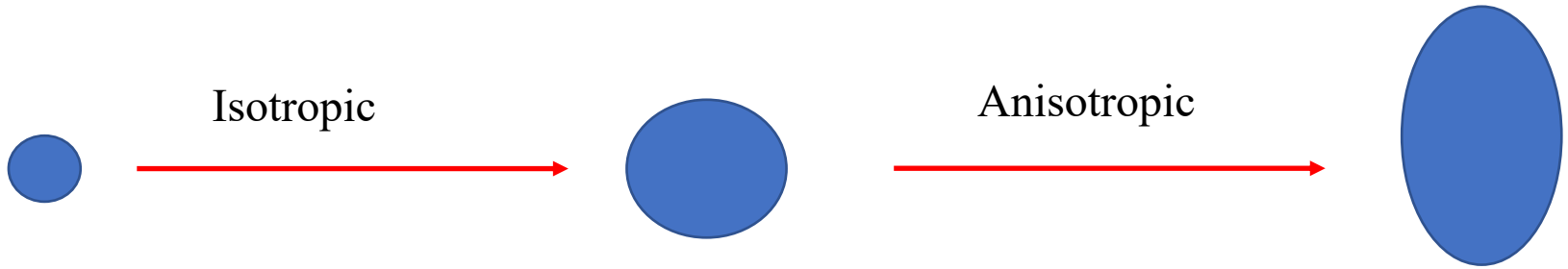
In 3D: $\mathbf{S} = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & s_z \end{pmatrix}$

$s_x = s_y = s_z$

$s_x \neq s_y \neq s_z$

degrees of freedom (isotropic): 1

degrees of freedom (anisotropic): 3



Matrix Transformations: Shear

In 2D:

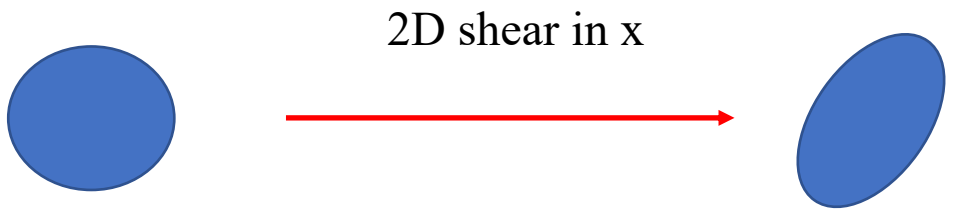
$$\mathbf{S} = \begin{pmatrix} 1 & s_x \\ s_y & 1 \end{pmatrix}$$

degrees of freedom: 2

In 3D:

$$\mathbf{S} = \begin{pmatrix} 1 & s_{x1} & s_{x2} \\ s_{y1} & 1 & s_{y2} \\ s_{z1} & s_{z2} & 1 \end{pmatrix}$$

degrees of freedom: 6



Euclidean Transformations

Also known as rigid transformations or isometries

Preserve lengths, angles and areas

A combination of rotations, reflections and translations

Euclidean Transformations

Also known as rigid transformations or isometries

Preserve lengths, angles and areas

A combination of rotations, reflections and translations

In 2D: degrees of freedom: 1(rotation) + 2(translation) = 3 $[\mathbf{R}|\mathbf{t}]_{2\times3}$

In 3D: degrees of freedom: 3(rotation) + 3(translation) = 6 $[\mathbf{R}|\mathbf{t}]_{3\times4}$

Similarity Transformations

Preserve ratio of lengths, angles

A combination of scale, rotations, reflections and translations

In 2D: degrees of freedom: 1(scale) + 1(rotation) + 2(translation) = 4 $s[\mathbf{R}|\mathbf{t}]_{2\times 3}$

In 3D: degrees of freedom: 1(scale) + 3(rotation) + 3(translation) = 6 $s[\mathbf{R}|\mathbf{t}]_{3\times 4}$

Affine Transformations

Preserve ratio of lengths, ratio of areas, parallelism

A combination of shear, scale, rotations, reflections and translations

In 2D: degrees of freedom: 2(shear) + 2(scale) + 1(rotation) + 2(translation) = 6

In 3D: degrees of freedom: 6(shear) + 3(scale) + 3(rotation) + 3(translation) = 12

Affine Transformations

Preserve ratio of lengths, ratio of areas, parallelism

A combination of shear, scale, rotations, reflections and translations

In 2D: degrees of freedom: 2(shear) + 2(scale) + 1(rotation) + 2(translation) = 6

In 3D: degrees of freedom: 6(shear) + 3(scale) + 3(rotation) + 3(translation) = 12

Degrees of freedom (dof) can’t be more than matrix size.

In 2D, its 2X3 =6 and in 3D, its 3X4=12.

Affine > Similarity > Euclidean

Transformations

To transform **x**, we write (using homogeneous coordinates): $\overline{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{t}$

$$\begin{pmatrix} \overline{\mathbf{x}} \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{A} & t \\ 0_{1\times d}^{\top} & 1 \end{pmatrix} \begin{pmatrix} \mathbf{x} \\ 1 \end{pmatrix}$$

d is dimension

Euclidean: **A** = **R**

Similarity: **A** = s**R**

Affine: **Ax + t**

Projective Transformations

Preserves collinearity

A combination of distortion, shear, scale, rotations, reflections and translations

In 2D: degrees of freedom: 8

In 3D: degrees of freedom: 15

$$\begin{pmatrix} \overline{\mathbf{x}} \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{A} & t \\ \mathbf{v}^{\top}_{1\times d} & v \end{pmatrix} \begin{pmatrix} \mathbf{x} \\ 1 \end{pmatrix}$$

Projective > Affine > Similarity > Euclidean

Always look at the rank of matrix to see how many dimensions does it impact

Eigenvalue and Eigenvectors

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x}, \quad \mathbf{A} \in \mathbb{R}^{N\times N}, \mathbf{x} \in \mathbb{R}^{N\times 1}, \lambda \in \mathbb{R}$$

Given a square matrix **A**, we need to find **x** such that if we transform it with **A**, it ends in scaling without change in direction

If found, **x** is an eigenvector and λ is eigenvalue

We can have up to N eigenvalues and eigenvectors

Eigenvalue and Eigenvectors

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x}, \quad \mathbf{A} \in \mathbb{R}^{N\times N}, \mathbf{x} \in \mathbb{R}^{N\times 1}, \lambda \in \mathbb{R}$$

Given a square matrix **A**, we need to find **x** such that if we transform it with **A**, it ends in scaling without change in direction

If found, **x** is an eigenvector and λ is eigenvalue

We can have up to N eigenvalues and eigenvectors

Find λ by solving $\det(\mathbf{A} - \lambda\mathbf{I}) = 0$

$tr(\mathbf{A}) = \sum \lambda$

$\det \mathbf{A} = \prod \lambda$

Rank = number of non-zero eigenvalues

Singular Value Decomposition (SVD)

Represent a matrix as a product of 3, in order to reveal internal structure

With SVD $\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$, where

U, **V** are orthogonal (rotation/unitary) matrices and

$\boldsymbol{\Sigma}$ is a scaling (diagonal) matrix

A does not need to be a square matrix

Geometrically, it can break down a transformation into simpler components

svd(**A**):

eigenvectors of $\mathbf{A}\mathbf{A}^T$ are columns of **U**, eigenvalues of $\mathbf{A}\mathbf{A}^T$ are sqrt of $\boldsymbol{\Sigma}$ elements

eigenvectors of $\mathbf{A}^T\mathbf{A}$ are columns of **V**, eigenvalues of $\mathbf{A}^T\mathbf{A}$ are sqrt of $\boldsymbol{\Sigma}$ elements

Singular Value Decomposition (SVD)

```
>> [U,S,V]=svd([1 2 3; 4 5 6])

U =

   -0.3863   -0.9224
   -0.9224    0.3863


S =

    9.5080         0         0
         0    0.7729         0


V =

   -0.4287    0.8060    0.4082
   -0.5663    0.1124   -0.8165
   -0.7039   -0.5812    0.4082
```

Quite small, make S(2,2)=0

```
>> U*S*V'

ans =

    1.5745    2.0801    2.5857
    3.7594    4.9664    6.1735
```

Quite similar to original matrix

Singular Value Decomposition (SVD)

```
>> [U,S,V]=svd([1 2 3; 4 5 6])

U =

   -0.3863   -0.9224
   -0.9224    0.3863

S =

    9.5080         0         0
         0    0.7729         0

V =

   -0.4287    0.8060    0.4082
   -0.5663    0.1124   -0.8165
   -0.7039   -0.5812    0.4082
```

```
>> U*S*V'

ans =

    1.5745    2.0801    2.5857
    3.7594    4.9664    6.1735
```

```
>> U*S*V'

ans =

   -0.5745   -0.0801    0.4143
    0.2406    0.0336   -0.1735
```

Blue and orange are two principal components

Singular Value Decomposition (SVD)

Blue and orange are two principal components

1.5745	2.0801	2.5857
3.7594	4.9664	6.1735

+

-0.5745	-0.0801	0.4143
0.2406	0.0336	-0.1735

=

1	2	3
4	5	6

Singular Value Decomposition (SVD)

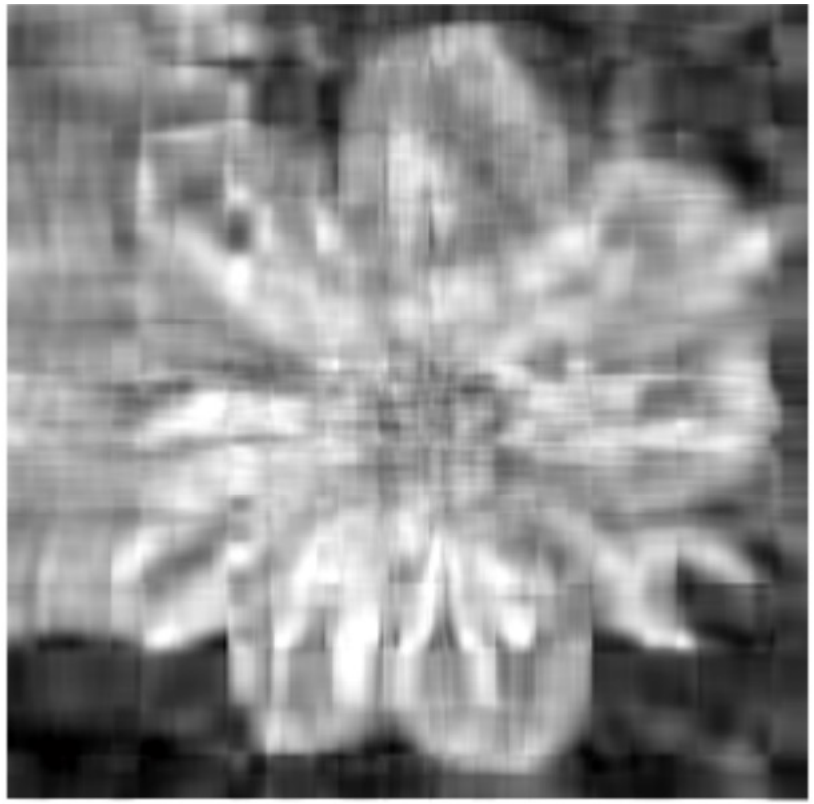


Image compression: By using on 10 principal components instead of 300

Principal Component Analysis (PCA)

Can give you what is relevant information in the data

Raw data samples can be redundant. With PCA, you can represent them as weights to principal components.

It can help with to do data compression, dimensionality reduction

How to do it?

1) Normalise data – zero mean,
2) Compute covariance ($\mathbf{AA}^T$ or $\mathbf{A}^T\mathbf{A}$ whichever is lighter),
3) Compute eigenvalues and eigenvectors

Reading material

https://cs229.stanford.edu/section/cs229-linalg.pdf

Wikipedia

Labwork

- Use matlab tutorial to get acquainted with matlab commands
- Practice today’s learnings
- Use greenspace.m to understand image manipulation using matlab

HW1: P0

Given **R** (a rotation matrix) and **A** (an invertible matrix), solve for **Y**

$$\mathbf{RYA = B}$$

Using Matlab, formulate **R**, **A, Y** and **B** numerically and confirm your obtained solution.

HW1: P1

Convert the following 2D affine transformation, into rotation, scaling, shearing and translation components (whichever applicable). Write a generic decomposition.

Hint: Use SVD

$$\mathbf{T} = \begin{pmatrix} \mathbf{A} & \mathbf{t} \\ 0_{1\times 2} & 1 \end{pmatrix} = \begin{pmatrix} 2 & 3 & 4 \\ 1 & 3 & 5 \\ 0 & 0 & 1 \end{pmatrix}$$

HW1: P2

Use SVD to compress image flower.bmp (convert it to a grayscale image).

What are your observations on compression with k = 10, 50, 100, 150, 200, 250 and 300 eigenvalues?

How would you decide an optimal compression (trade-off between compression accuracy and data storage)?

What is the optimal k, according to you?

What is the reduced data size?