

Contents

- Feature
- Image analysis
 - Projection of images
 - 3D vector geometry
 - Medical imaging
 - Extracting geometric features
- Signal analysis
 - Fourier transform
 - Short time Fourier transform

Introduction (1)

- Data: unstructured, disorganized and jumbled formats
 - various forms: recorded music, photographs, or measured data
 - some processing required to get it into a useful form and to isolate the key features of the data
- Extracting the key features of data
 - Important step that begins the transformation from data to information
 - re-organize data to make it possible to achieve useful outcomes
 - understand the scientific principles associated with data
- How to
 - extract features from a given dataset
 - normalize them to establish meaningful correlation
 - use available scientific or mathematical tools to minimize the number of features for consideration

Introduction (2)

- many tools and methods for extracting features from data
 - Geometrical analysis for image processing
 - Fourier transform analysis
 - short-time Fourier transform analysis
- Real-world examples of feature extraction using vector geometry and mathematics
 - Image analysis: medical imaging (X-rays, MRI's, Ultrasounds)
 - Signal processing: Fourier transform

Features (1)

- a **measurable** property of the object (sample) in dataset
- Examples
 - predict how much he/she is going to get in his/her final exam
 - data on students taking a course: past performance in CGPA, the grades in assignments, mid-terms, attendance in class
 - evidence of archaeological sites, species migrations, and changes in shorelines
 - satellite images
 - Facial recognition software and autonomous vehicle technology
 - extensive image collections
 - aid in medical diagnosis and treatment
 - medical images, such as X-rays, MRIs, and Ultrasounds

Features (2)

- a **measurable** property of the object (sample) in dataset
- Features in other scenarios



1. Where to go out to eat for dinner

- **Type of food**
(Mexican, Italian, etc.)
- **Price**
- **Size of portions**
(All you can eat, small, etc.)
- **Style of restaurant**
(Takeout, sit down, etc.)



2. Where to go on vacation

- **Distance from home**
(out of state, international, etc.)
- **Time of year**
(Summer, Winter, etc.)
- **Languages spoken**
- **Expense of trip**
(Flights, hotel, food, etc.)



3. Determining the weather

- **Wind speed**
- **Time of year**
(Fall, Winter, etc.)
- **Number of clouds**
- **Climate of location**
(Humid beach, desert, etc.)

Commercial Applications of Feature Engineering

- **Billboard Advertisement**
- Need to identify the important features of the advertisement
- Once the important features are identified, a relationship between the features can be established.
- Billboard can be designed to emphasize the important features, factoring in the relationships between them,



Normalization of Feature Data

[Standard Normalization]

$$z_i = \frac{X_i - \mu}{\sigma}$$

z_i : standard normalized data point i

X_i : original data point i

$\mu = \frac{1}{N} \sum_{i=1}^N X_i$: mean for a particular feature (N is the total number of data points)

$\sigma = \sqrt{\frac{\sum_{i=1}^N (X_i - \mu)^2}{N}}$: standard deviation of data for that same feature

[Feature Scaling]

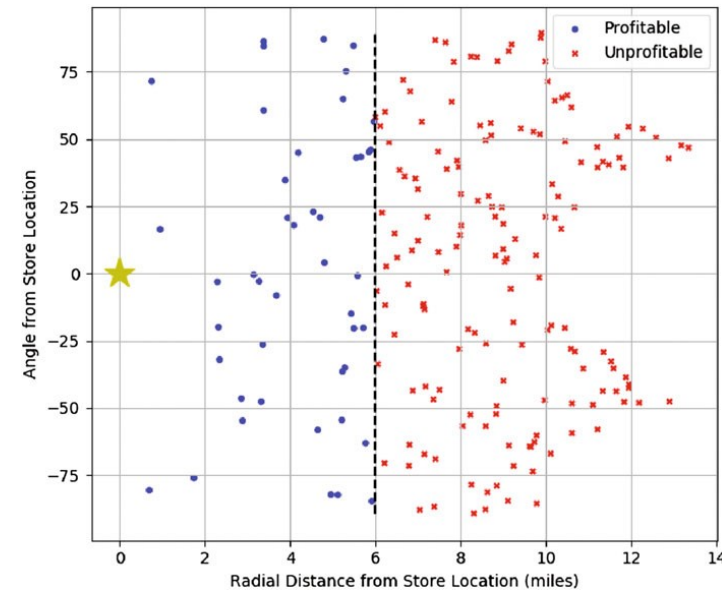
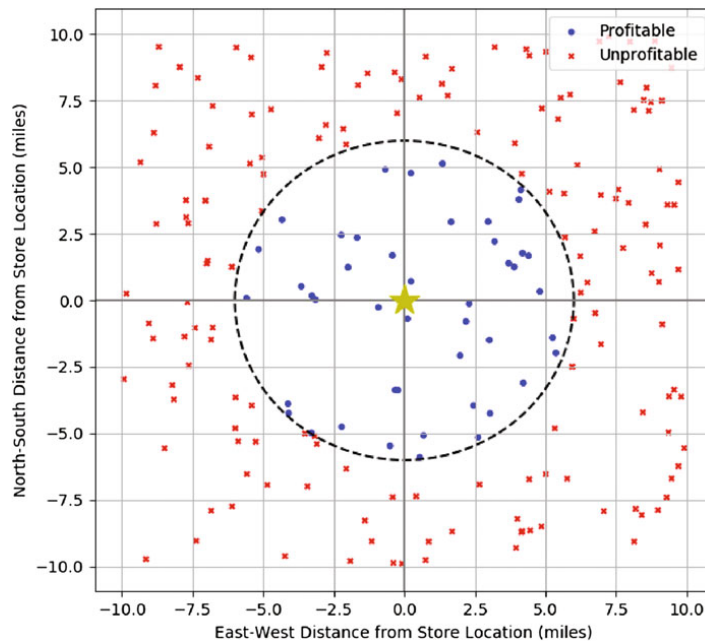
$$z_i = \frac{X_i - X_{\min}}{X_{\max} - X_{\min}}$$



Proximity to schools	Bedrooms	Age of house	Price, \$
0.3 miles	3	10 year	200,000
1.2 miles	2	12 years	100,000
2.0 miles	4	2 years	400,000
0.7 miles	3	8 years	150,000

Feature Engineering

- transform the features in a dataset into a form that is easier to use and interpret, while maintaining the critical information
 - Coordinate systems: Cartesian \rightarrow polar



Example: Determining a New Store Location Using Coordinate Transformation Techniques

Projection of Images (3D to 2D) and Image Processing

- Photographs and videos are a constant aspect of daily life
 - television, online images and videos, or photos and video recording of dynamic testing
- Forensic challenge with photos and videos
 - how to extract meaningful measurements and data these images
- instant replay reviews in major sporting events
 - On December 7, 1963, Director Tony Verna pioneered the use of instant replay during the annual Army Navy football game.
- Instant replay allows analysts and viewers to relive key moments.
- To implement instant replay, sporting events are filmed from multiple camera angles simultaneously.

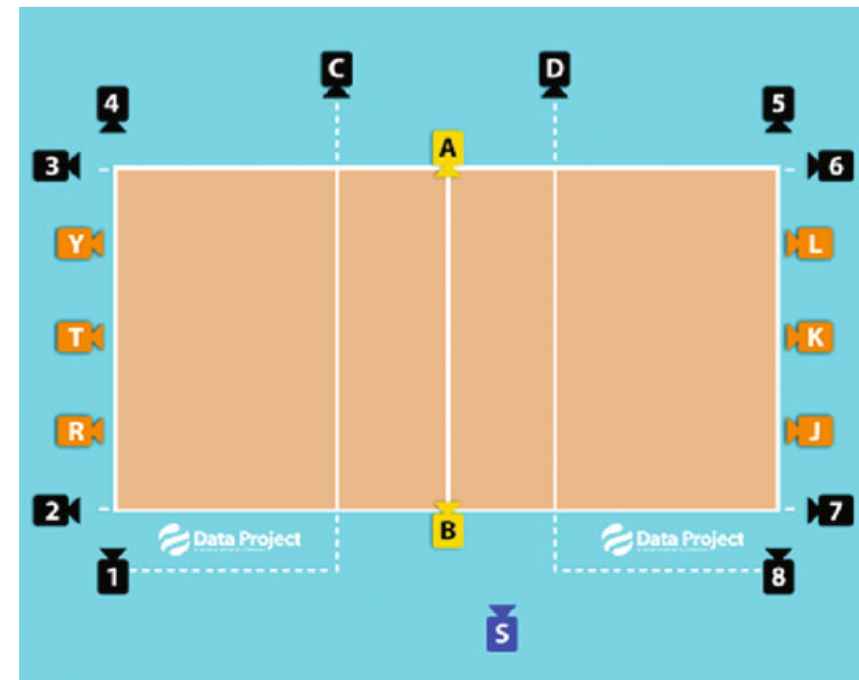
Motivation of 3D image reconstruction

- Houston Astros claim that Game 4 of the 2018 American League Championship Series (ALCS) was impacted by insufficient instant replay. Jose Altuve appeared to hit a baseball over the fence but was called “out” for fan interference. Replay review was inconclusive:
- despite Major League Baseball’s multitude of cameras, no angle clearly captured the ball’s position relative to the fielder’s glove, the fans’ hands, and the fence.



Tracking the ball in Volleyball game

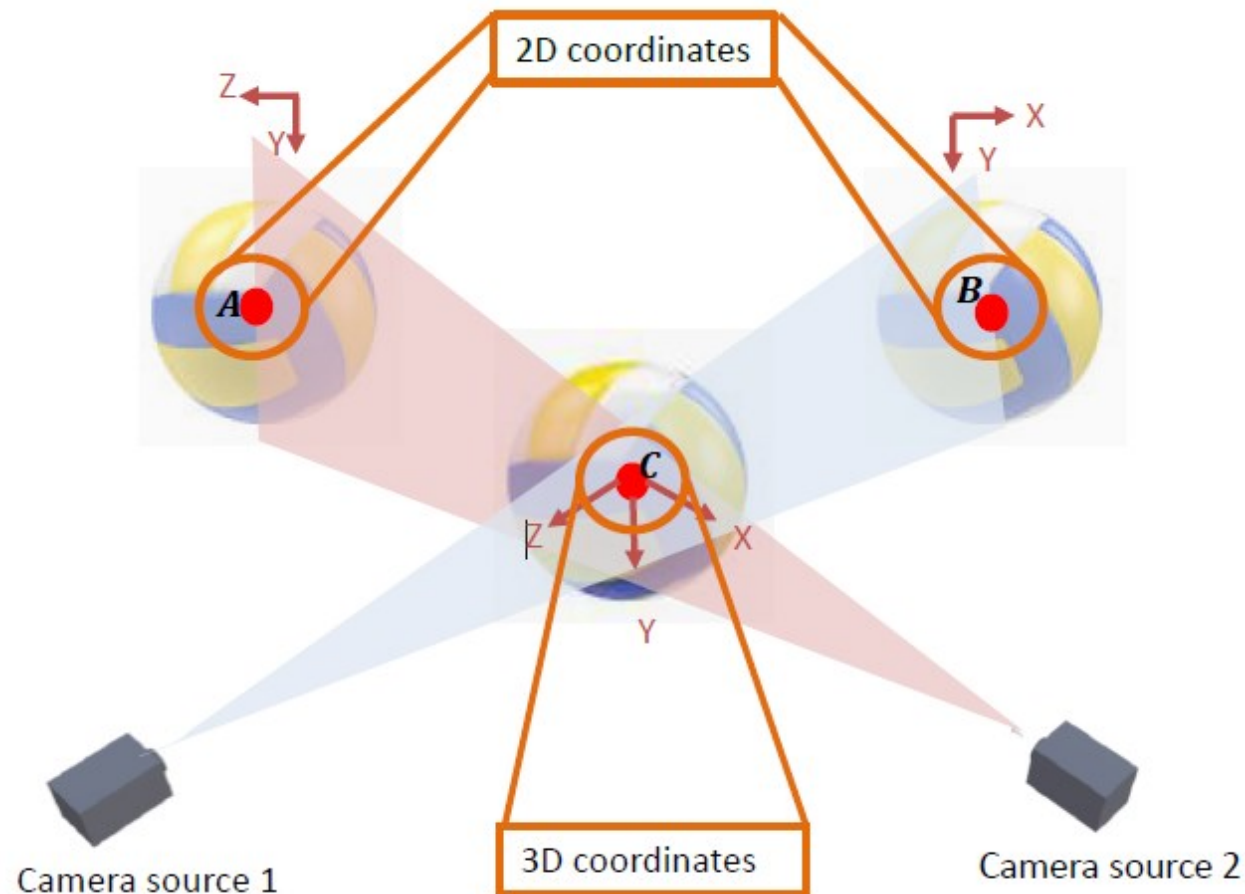
- Tracking the 3D location of the ball is crucial.
- It is very common that the referees request video check to determine the position of a ball.
- Some rules and fouls that are related to the position of the ball.



[VideoCheck \(dataproject.com\)](https://dataproject.com)

Problem Statement

- A point in 3D space is projected to the planes XY and YZ.
- How can we construct the 3D geometry from 2D images?



Review of 3D geometry: a vector and a point in 3D space

- A 3D vector (\mathbf{d}) is a line segment in three dimensional space running from point \mathbf{O} (tail) to point \mathbf{A} (head).
- A line in 3D space can be uniquely defined with a single point (\mathbf{P}) and a direction (\mathbf{d}).

$$\mathbf{O} = O_x \mathbf{i} + O_y \mathbf{j} + O_z \mathbf{k}$$

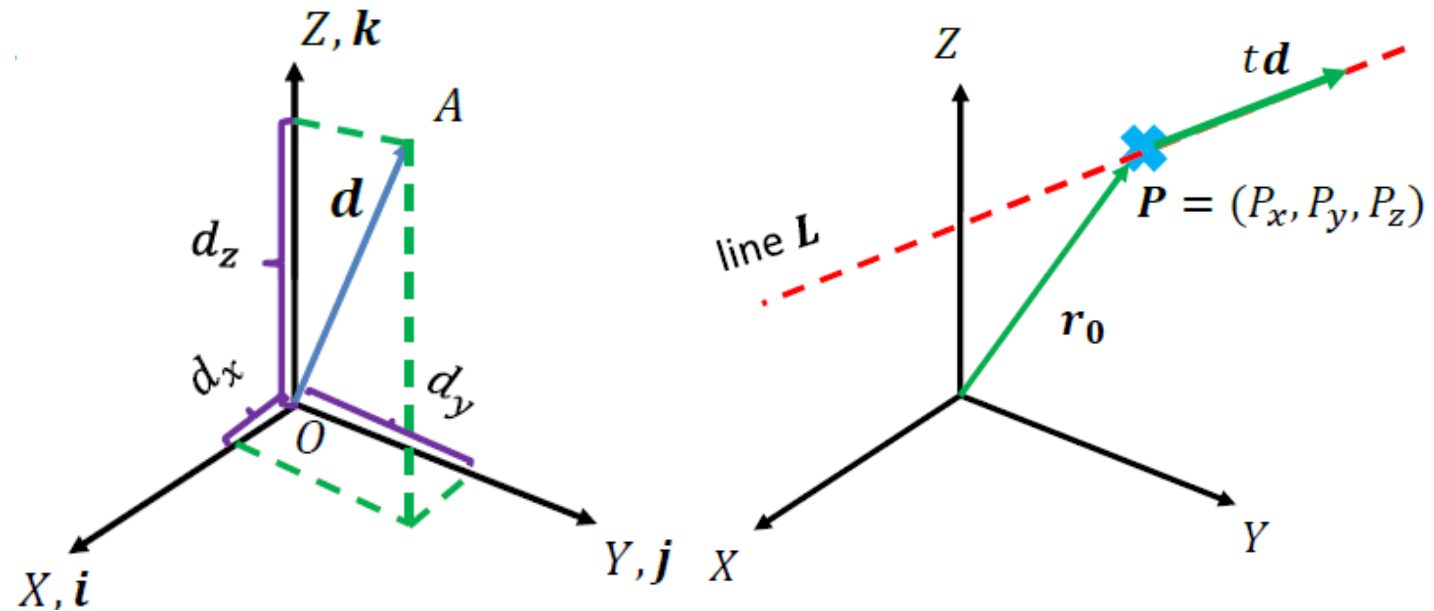
$$\mathbf{A} = A_x \mathbf{i} + A_y \mathbf{j} + A_z \mathbf{k}$$

$$\begin{aligned} \mathbf{d} &= \mathbf{A} - \mathbf{O} = (A_x - O_x) \mathbf{i} + (A_y - O_y) \mathbf{j} + (A_z - O_z) \mathbf{k} \\ &= d_x \mathbf{i} + d_y \mathbf{j} + d_z \mathbf{k} \end{aligned}$$

$$\mathbf{r} = \mathbf{r}_0 + t\mathbf{d}$$

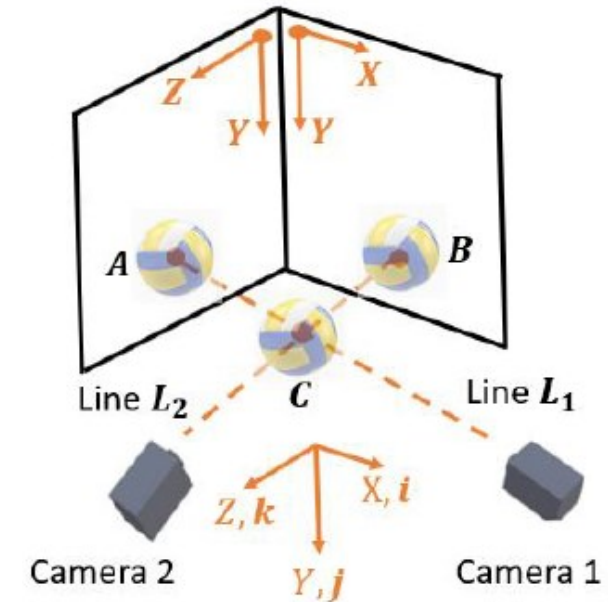
\mathbf{r}_0 : position vector of point P

$$\begin{pmatrix} r_x \\ r_y \\ r_z \end{pmatrix} = \begin{pmatrix} P_x \\ P_y \\ P_z \end{pmatrix} + t \begin{pmatrix} d_x \\ d_y \\ d_z \end{pmatrix} \rightarrow t = \frac{r_x - P_x}{d_x} = \frac{r_y - P_y}{d_y} = \frac{r_z - P_z}{d_z}$$



Problem Definition and Assumptions

- Objective: locate the 3D coordinates of the volleyball (point C)
 - Point C is the center point of a ball
 - Camera 1 captures the 2D projection of point C on plane ZY (point A)
 - Camera 2 captures the 2D projection of point C on plane XY (point B)
- Problem setup for locating the volleyball position
 - $L1, L2$: the line passing through point A and Camera 1, point B and Camera 2
 - $d1$: direction vector of line $L1$ connecting Camera 1 to point A on plane ZY
 - $d2$: direction vector of line $L2$ connecting Camera 2 to point B on plane XY
 - Point C is located at the intersection of $L1$ and $L2$



$$\text{point } \mathbf{A} = y_A \mathbf{j} + z_A \mathbf{k}$$

$$\text{point } \mathbf{B} = x_B \mathbf{i} + y_B \mathbf{j}$$

$$\mathbf{d}^1 = d_x^1 \mathbf{i} + d_y^1 \mathbf{j} + d_z^1 \mathbf{k}$$

$$\mathbf{d}^2 = d_x^2 \mathbf{i} + d_y^2 \mathbf{j} + d_z^2 \mathbf{k}$$

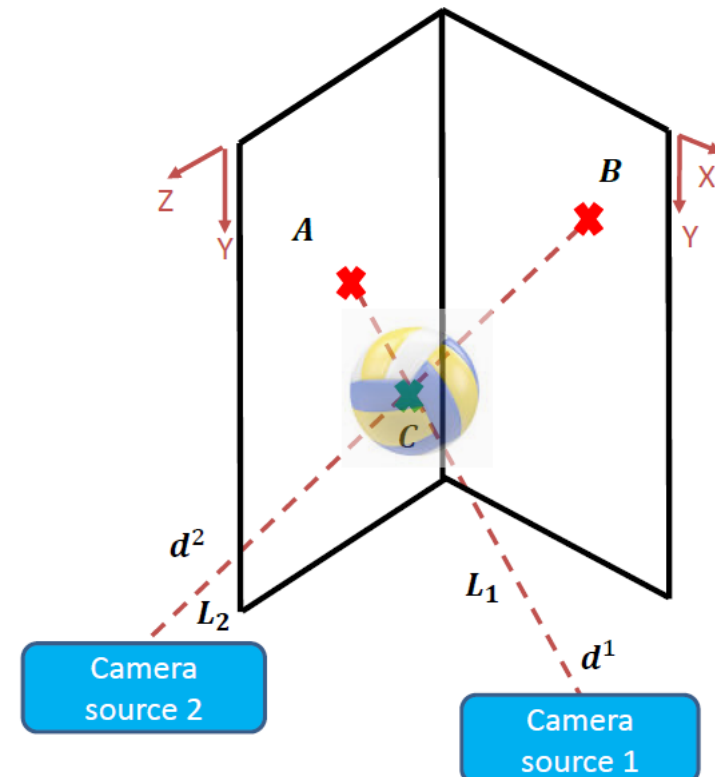
How to find the intersection of lines L1 and L2?

- Known:
 - Point A , B : The projection of point C on plane ZY and plane XY
 - d_1 , d_2 : the direction of line L_1 and line L_2
- Unknown point C coordinates(x,y,z) are given by the intersection of lines L_1 and L_2

$$\mathbf{L}_1 = \begin{pmatrix} 0 \\ y_A \\ z_A \end{pmatrix} + \alpha \begin{pmatrix} d_x^1 \\ d_y^1 \\ d_z^1 \end{pmatrix} \leftrightarrow \mathbf{L}_2 = \begin{pmatrix} x_B \\ y_B \\ 0 \end{pmatrix} + \beta \begin{pmatrix} d_x^2 \\ d_y^2 \\ d_z^2 \end{pmatrix}$$

intersection of lines \mathbf{L}_1 and $\mathbf{L}_2 \rightarrow$ find point C

$$\begin{pmatrix} \alpha d_x^1 \\ y_A + \alpha d_y^1 \\ z_A + \alpha d_z^1 \end{pmatrix} = \begin{pmatrix} x_B + \beta d_x^2 \\ y_B + \beta d_y^2 \\ \beta d_z^2 \end{pmatrix}$$



How to find the intersection of lines L1 and L2?

$$\begin{cases} \alpha d_x^1 - \beta d_x^1 = x_B \\ \alpha d_y^1 - \beta d_y^2 = -y_A + y_B \\ \alpha d_z^1 - \beta d_z^2 = -z_A \end{cases} \rightarrow \begin{bmatrix} d_x^1 & -d_x^1 \\ d_y^1 & -d_y^2 \\ d_z^1 & -d_z^2 \end{bmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} x_B \\ -y_A + y_B \\ -z_A \end{pmatrix} \Leftrightarrow \mathbf{A} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \mathbf{b}$$

three equations with two unknowns $(\alpha, \beta) \rightarrow$ no unique solution

try to find a solution that minimize the error of the three equations \rightarrow Least Squares Regression Method
to find the system of equation using the least square method, the following equation should be solved:

$$\mathbf{A}^T \mathbf{A} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \mathbf{A}^T \mathbf{b}$$

Find the corresponding values for α and β , and substitute in the equations of either \mathbf{L}_1 or \mathbf{L}_2

Find the intersection point (point C)

Numerical Example

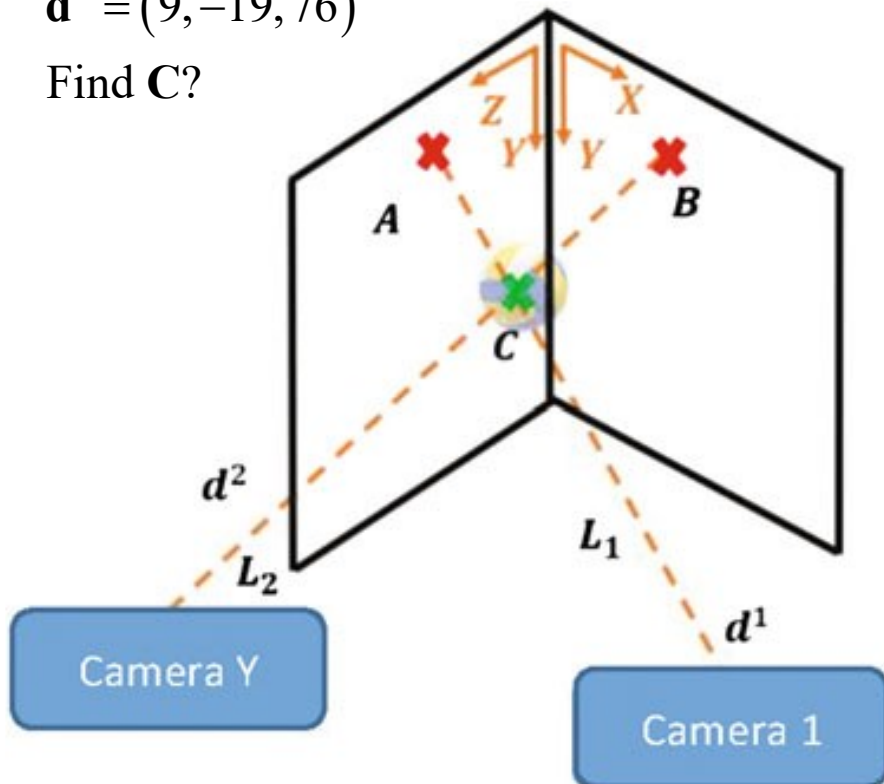
$$\mathbf{A} = (A_x, A_y, A_z) = (0, 50, 70)$$

$$\mathbf{B} = (B_x, B_y, B_z) = (30, 40, 0)$$

$$\mathbf{d}^1 = (7, 3, 2)$$

$$\mathbf{d}^2 = (9, -19, 76)$$

Find C?



1. Define directional lines

$$\mathbf{L}_1 = \begin{pmatrix} 0 \\ 50 \\ 70 \end{pmatrix} + \alpha \begin{pmatrix} 7 \\ 3 \\ 2 \end{pmatrix}$$

$$\mathbf{L}_2 = \begin{pmatrix} 30 \\ 40 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 9 \\ -19 \\ -76 \end{pmatrix}$$

2. Equate x, y, and z equations

$$\begin{aligned} x &= 7\alpha = 30 + 9\beta \\ y &= 50 + 3\alpha = 40 - 19\beta \\ z &= 70 + 2\alpha = -76\beta \end{aligned}$$

3. Rewrite equations in matrix form

$$\mathbf{A} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \mathbf{b}$$

$$\begin{bmatrix} 7 & -9 \\ 3 & 19 \\ 2 & 76 \end{bmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} 30 \\ -10 \\ -70 \end{pmatrix}$$

4. Calculate the least squares solution

$$\mathbf{A}^T \mathbf{A} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \mathbf{A}^T \mathbf{b}$$

yields $\alpha = 3$ and $\beta = 1$

5. Determine C

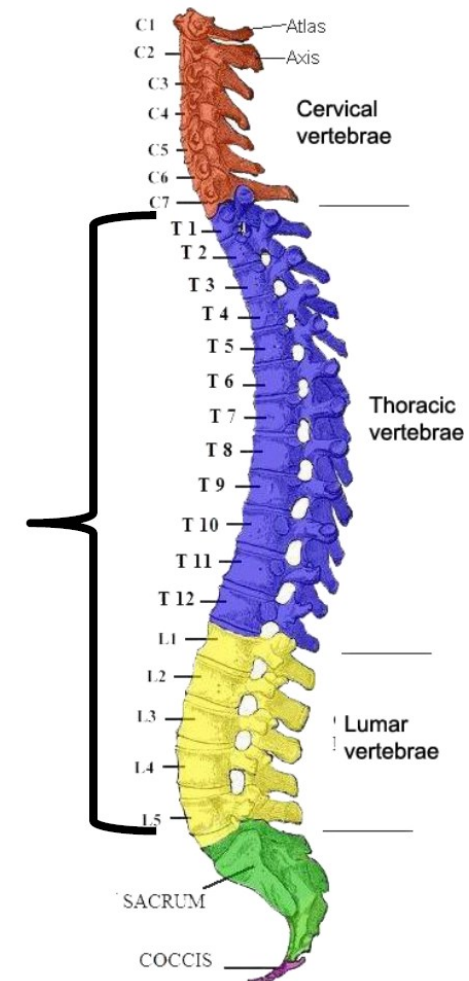
$$\mathbf{C} = (C_x, C_y, C_z) = (21, 59, 76)$$

Scoliosis & Overview of spine Anatomy

- Scoliosis is a three dimensional deformity of the spine characterized by a lateral deviation of the spine, accompanied by an axial rotation of the vertebrae.
- The human spine contains 24 vertebrae in three regions:
 - Cervical Spine (7 vertebrae)
 - Thoracic Spine (12 vertebrae)
 - Lumbar Spine (5 vertebrae)
- It is assumed that Thoracic spine and Lumbar spine are responsible for scoliosis (17 vertebrae).

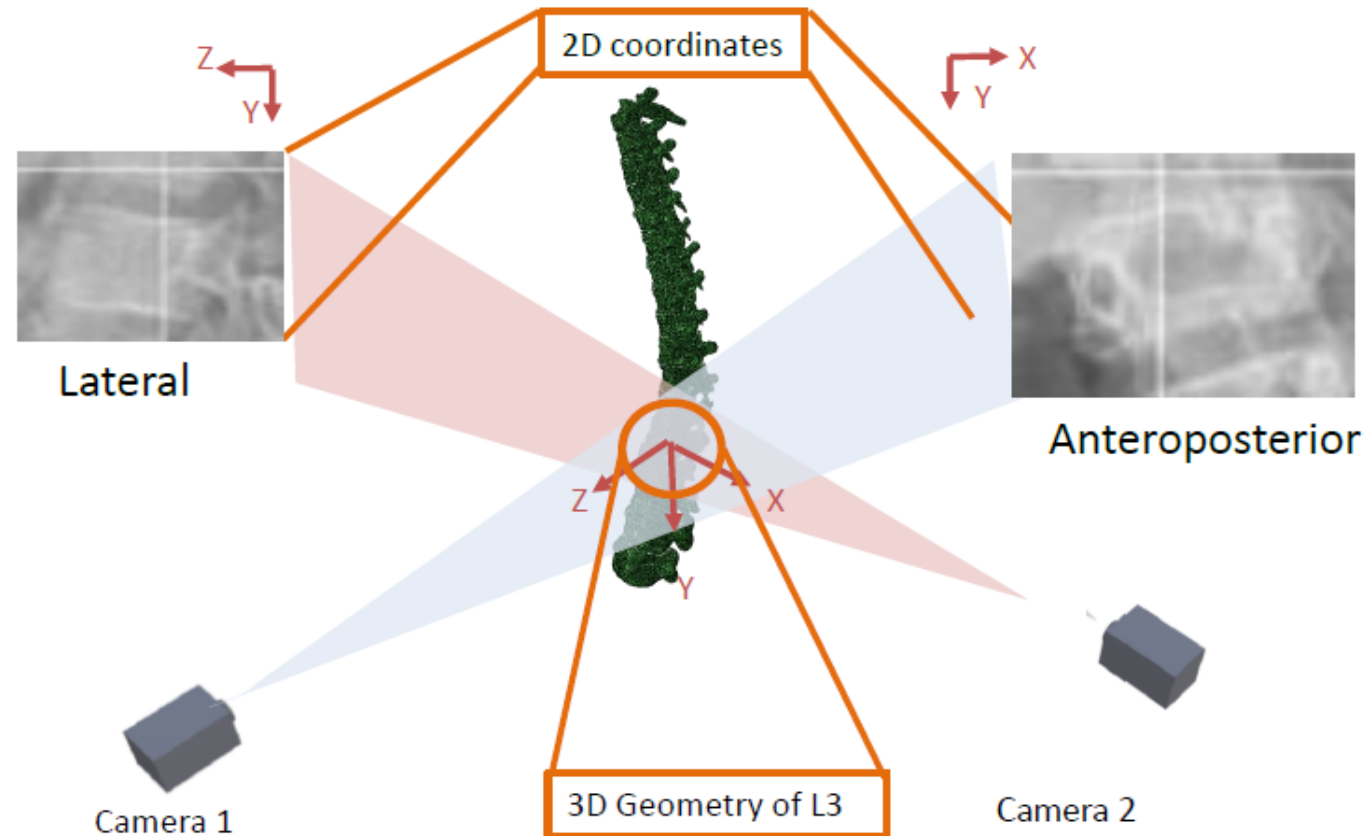


Scoliosis



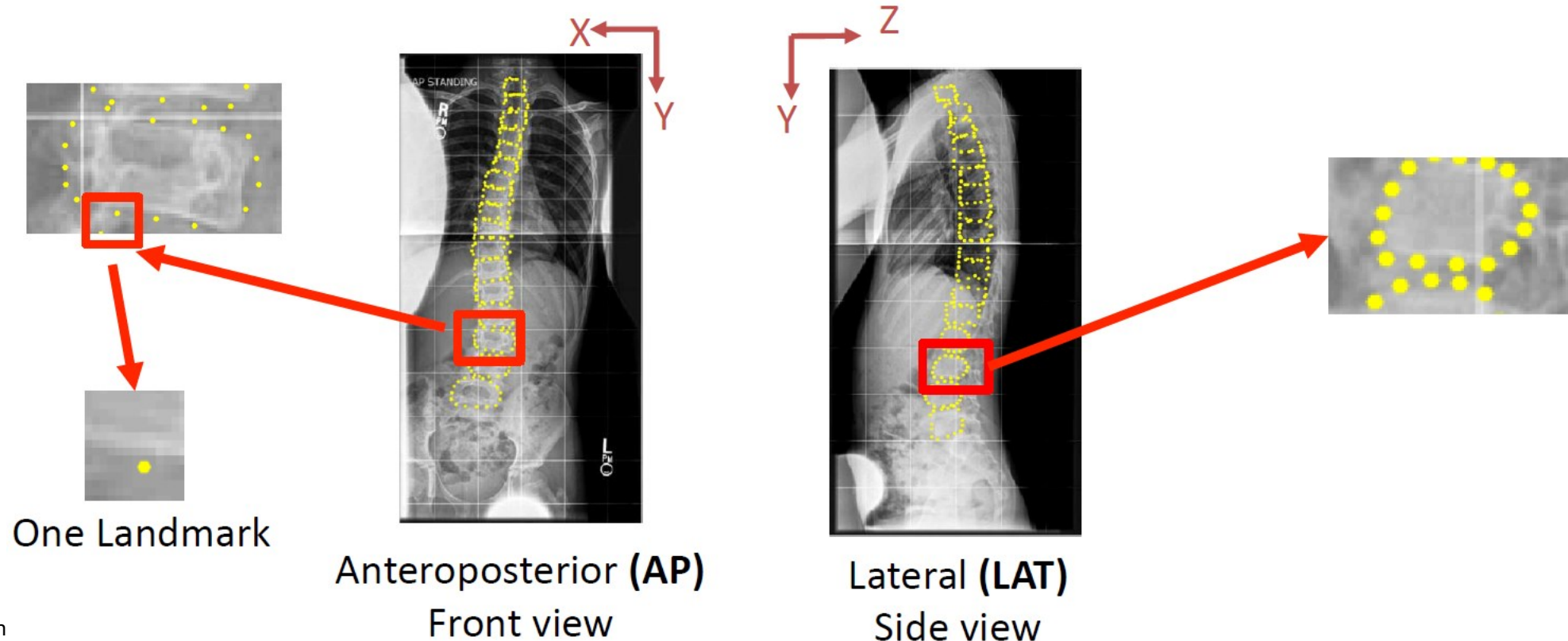
3D Reconstruction of the spinal column

- The X-ray images are the 2D projections of the vertebrae from 3D space to the planes XY and ZY.
- Goal: reconstruct the 3D vertebrae from 2D images?



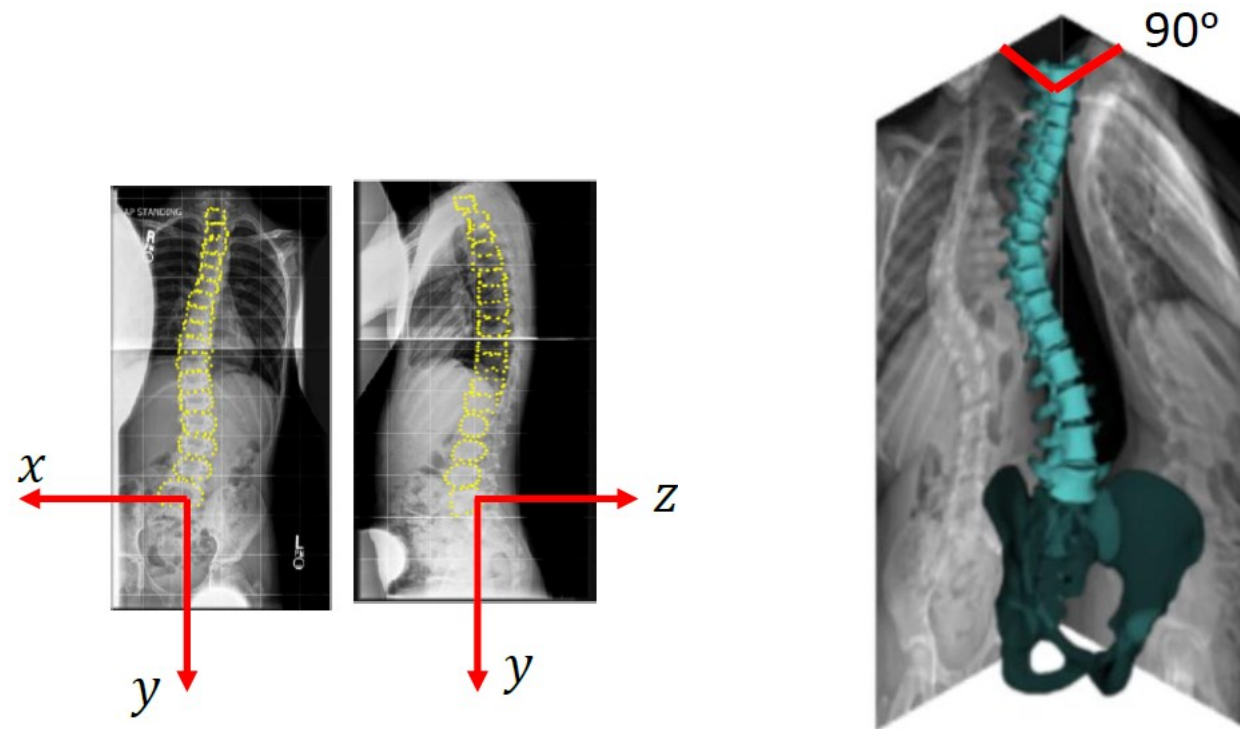
Outline each vertebra

- In this problem we have considered 16 landmarks (yellow dots) to segment each vertebra
 - 16 landmarks have been chosen in each view to define the shape of each vertebra.
 - X-ray Segmentation and Snake algorithm to locate the landmarks



Problem Setup: Reference of the coordinate system

- In AP and LAT view, the reference of the coordinate system is the center point of L5 (5th vertebra of Lumbar spine).
- AP and LAT views are perpendicular: the angle between AP and LAT views are 90° .



Pasha, Saba, and John Flynn. "Data-driven classification of the 3D spinal curve in adolescent idiopathic scoliosis with an applications in surgical outcome prediction." *Scientific reports* 8.1 (2018): 1-10.

Extracting Geometry Features Using 2D X-ray Images

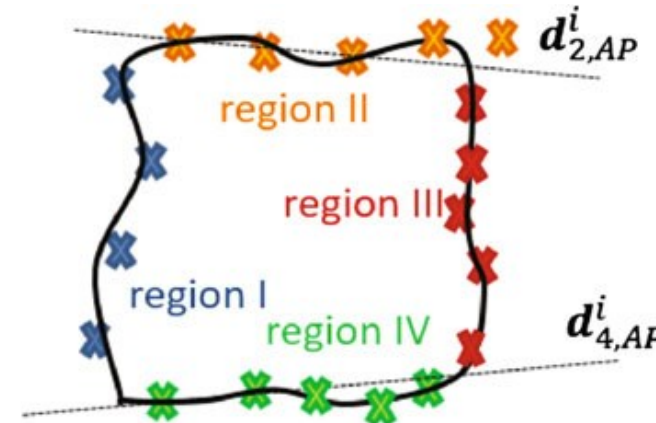
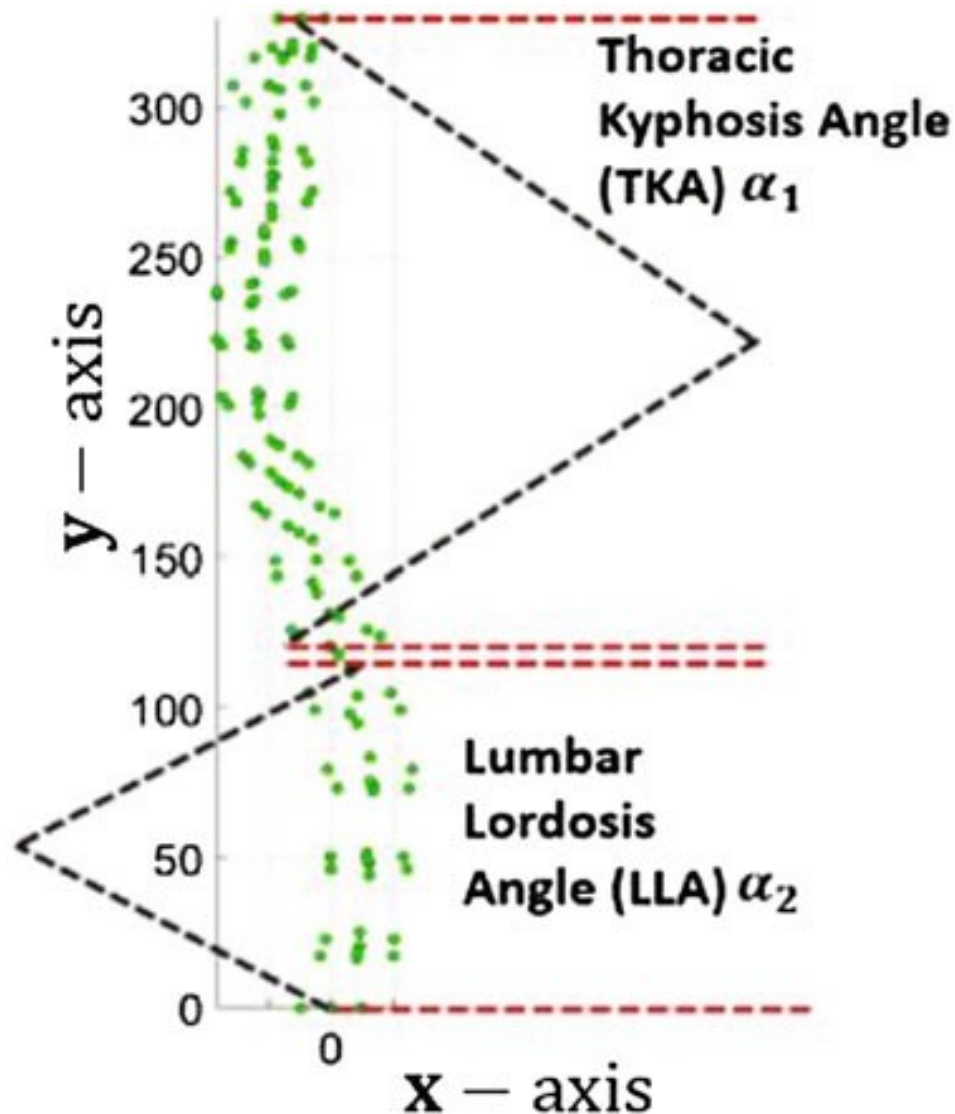
- spinal deformities can be described by five global angles
 - Trunk Inclination Angle (TIA)
 - Sacral Inclination Angle (SIA)
 - Thoracic Kyphosis Angle (TKA)
 - Lumbar Lordosis Angle (LLA)
 - Cobb Angle (CA)
- how to standardize feature calculations used in AIS diagnosis and classification
 - Doctors manually estimate vertebrae planes and locations from patient x-rays, measuring angles according to individual definitions
 - AIS treatment methodology varies by hospital

Global Angles

plane	angle	Line 1		Line 2		
		vertebra	region	vertebra	region	
LAT	TKA	T1	II	T12	IV	
	LLA	L1	II	L5	IV	
AP	TIA	L5	IV	L	IV	
	SIA	L5	IV	T	IV	
	CA	U	II	B	IV	C shape
	CA1	U1	II	B1	IV	S shape
	CA2	U2	II	B2	IV	

- L, T: the vertebra corresponding the point that has maximum curvature in the lumbar and thoracic spine
- U, B: vertebrae two above and two below C (the vertebra with maximum curvature)
- U1, B1: vertebrae two above and two below T (the thoracic vertebra with maximum curvature)
- U2, B2: vertebrae two above and two below L (the lumbar vertebra with maximum curvature)

Angles in the Lateral (LAT) Plane



$$v_1 \cdot v_2 = \|v_1\| \|v_2\| \cos \theta$$

$$\rightarrow \theta = \cos^{-1} \left(\frac{v_1 \cdot v_2}{\|v_1\| \|v_2\|} \right)$$

angle between Region II of T1 and Region 4 of T12

$$TKA = \cos^{-1} \left(\frac{d_{2,LAT}^1 \cdot d_{4,LAT}^{12}}{\|d_{2,LAT}^1\| \|d_{4,LAT}^{12}\|} \right)$$

angle between Region 11 of L1 and Region 4 of L5

$$LLA = \cos^{-1} \left(\frac{d_{2,LAT}^{13} \cdot d_{4,LAT}^{17}}{\|d_{2,LAT}^{13}\| \|d_{4,LAT}^{17}\|} \right)$$

Angles in the Anteroposterior (AP) Plane

angle between the region 4 of vertebra 17 (L5) and the vertebra L

$$TIA = \cos^{-1} \left(\frac{d_{4,AP}^{17} \cdot d_{4,AP}^L}{\|d_{4,AP}^{17}\| \|d_{4,AP}^L\|} \right)$$

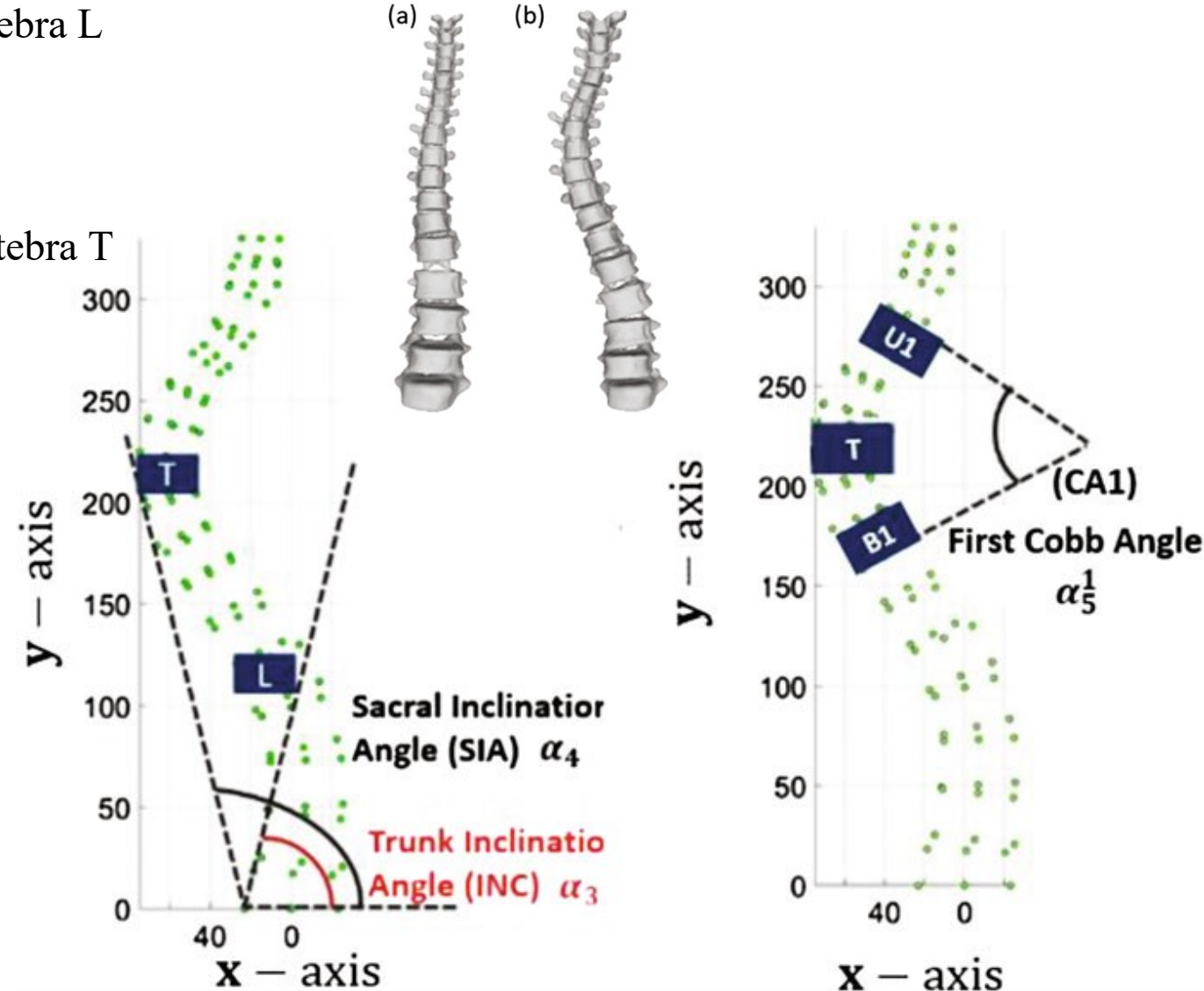
angle between the region 4 of vertebra 17 (L5), and the vertebra T

$$SIA = \cos^{-1} \left(\frac{d_{4,AP}^{17} \cdot d_{4,AP}^T}{\|d_{4,AP}^{17}\| \|d_{4,AP}^T\|} \right)$$

angle between Region 2 of U and Region 4 of B

single curvature (C shape): $CA = \cos^{-1} \left(\frac{d_{2,AP}^U \cdot d_{4,AP}^B}{\|d_{2,AP}^U\| \|d_{4,AP}^B\|} \right)$

double-curvature (S shape):
$$\begin{cases} CA1 = \cos^{-1} \left(\frac{d_{2,AP}^{U1} \cdot d_{4,AP}^{B1}}{\|d_{2,AP}^{U1}\| \|d_{4,AP}^{B1}\|} \right) \\ CA1 = \cos^{-1} \left(\frac{d_{2,AP}^{U2} \cdot d_{4,AP}^{B2}}{\|d_{2,AP}^{U2}\| \|d_{4,AP}^{B2}\|} \right) \end{cases}$$



Problem Setup: Scaling

- The images are taken with different scales. Images need to be scaled such that the heights of the spine in the two X ray views are the same.
- Assume the AP (Anteroposterior plane (xy view)) is fixed.
- We aim at calibrating LAT (lateral (yz view)).

(1) Calculate the scale factor: $s = \frac{y_{\max}^{AP} - y_{\min}^{AP}}{y_{\max}^{LAT} - y_{\min}^{LAT}}$

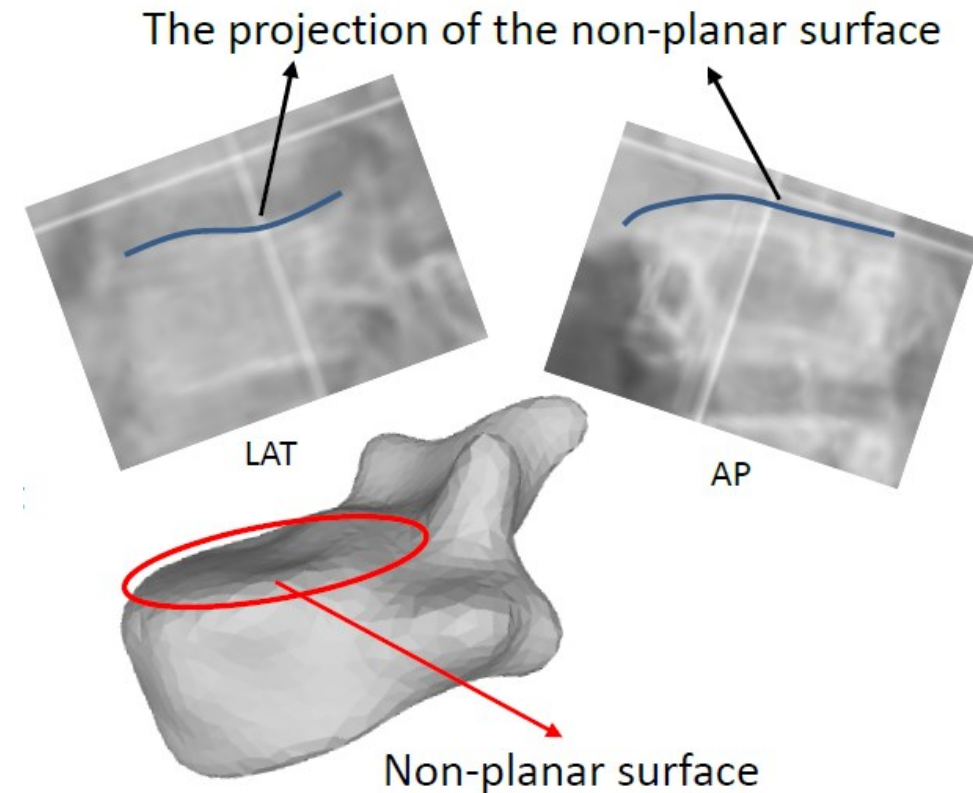
$y_{\max}^{AP}, y_{\min}^{AP}$: Maximum and Minimum of y coordinates of landmarks (yellow points) in AP

$y_{\max}^{LAT}, y_{\min}^{LAT}$: Maximum and Minimum of y coordinates of landmarks (yellow points) in LAT

(2) Update the coordinates of Lateral view:
$$\begin{cases} z \rightarrow s(z - z_{\min}^{LAT}) + z_{\min}^{LAT} \\ y \rightarrow s(y - y_{\min}^{LAT}) + y_{\min}^{AP} \end{cases}$$

3D Reconstruction of a vertebrae

- Challenges
 - The vertebra has a non planar surface.
 - The projection of the non-planer surface is not the same in two views.
 - We cannot find identical corresponding points in two different views.
 - The lines passing through the nodes will not intersect each other in the space.
- What is the solution?
 - Find a bounding box which gives us the estimation of the 3D reconstructed geometry in 3D space and use those points to generate the detailed geometry.
 - Bounding box: region in 3D space that contains the 3D reconstructed points.



3D Reconstruction of a vertebrae

- Object Detection and Bounding Boxes
 - Bounding boxes can be defined either in 2D or 3D.
 - 2D bounding boxes can be implemented to identify the vertebrae in medical images.
 - There are several techniques to define a bounding box.
 - In this problem, a bounding box will be determined by finding the intersections of lines in 3D space.



2D bounding boxes*



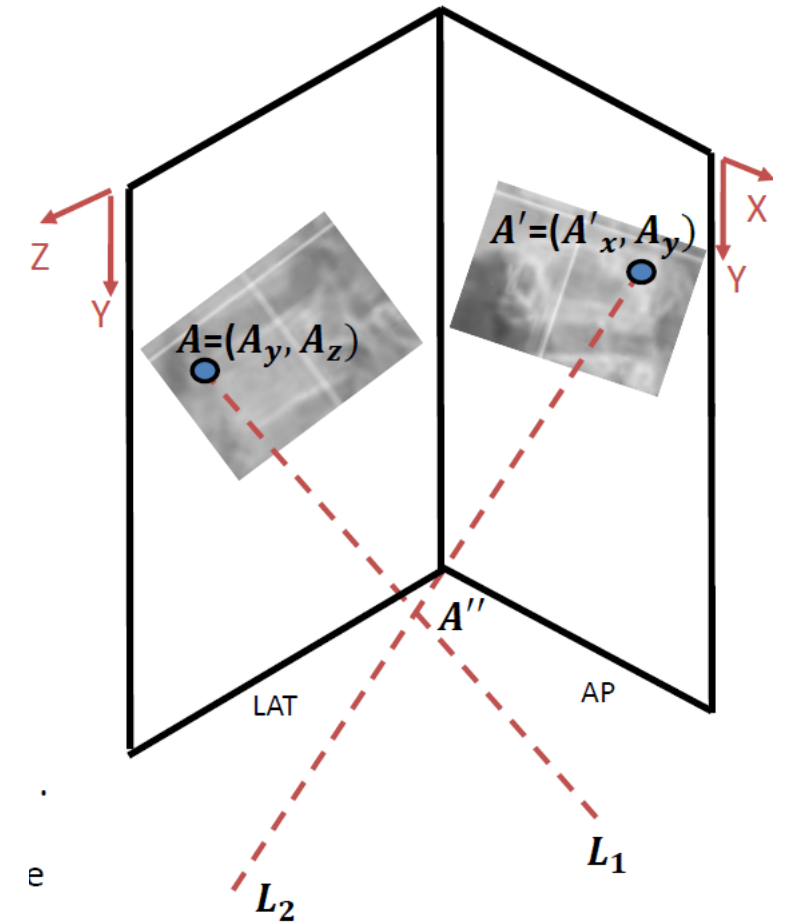
3D Bounding
boxes of the
lumbar
vertebrae in
lateral view



Detailed
geometry**

3D Reconstruction of a vertebrae

- A vertebra is projected to ZY and XY planes.
- Points A and A' are not corresponding the each other (are not the projection of a same point in vertebra).
- Line **L1** is perpendicular to the ZY plane and passes through point A.
- Line **L2** is perpendicular to the XY plane and passes through point A'.
- Goal: find a point (A'') that has the minimum distance from Lines **L1** and **L2**.
- point (A'') can be used to reconstruct the bounding box.



Problem Statement

- Find a point that has the minimum distance of lines $L1$ and $L2$.
- Note that $L1$ and $L2$ are not intersecting in 3D space.

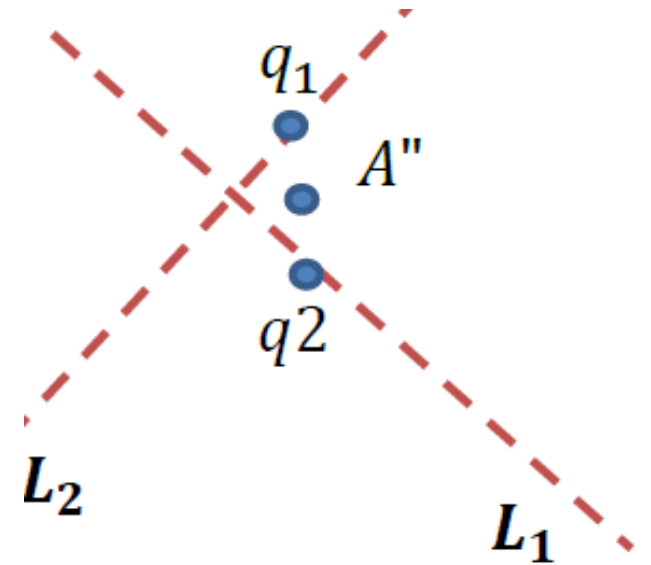
$$\mathbf{L}_1 = \begin{pmatrix} 0 \\ A_y \\ A_z \end{pmatrix} + \alpha \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \leftrightarrow \mathbf{L}_2 = \begin{pmatrix} A'_x \\ A'_y \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

$\mathbf{L}_1 = \mathbf{L}_2 \rightarrow$ solve for α and β

Substitute α in $\mathbf{L}_1 \rightarrow$ find the corresponding point \mathbf{q}_1

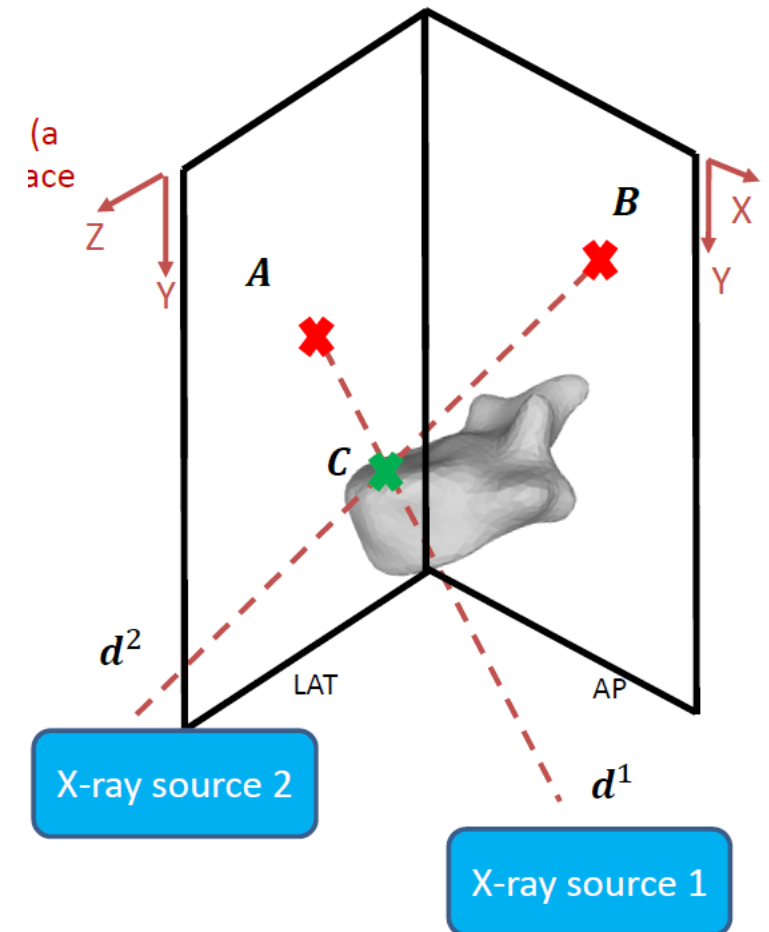
Substitute β in $\mathbf{L}_2 \rightarrow$ find the corresponding point \mathbf{q}_2

The closest point between lines $L1$ and $L2$ can be derived as: $A'' = \frac{\mathbf{q}_1 + \mathbf{q}_2}{2}$



Numerical Example

- The coordinate of the projection of a given point C (a point on the top surface of a vertebra) in the 3D space on plane YZ and XZ are described as follows:
 - $A = (0, 34, 52)$ [mm]
 - $B = (28, 36, 0)$ [mm]
- Let's assume that the direction of the line connecting X ray source 1 to the point A located on the YZ plane and that the direction of the line passing through X ray source 2 and point B located on the XZ plane are given by:
 - $d^1 = (1, 0, 0)$
 - $d^2 = (0, 0, 1)$
- Find the coordinates of point C in the 3D space?



2D to 3D reconstruction algorithm

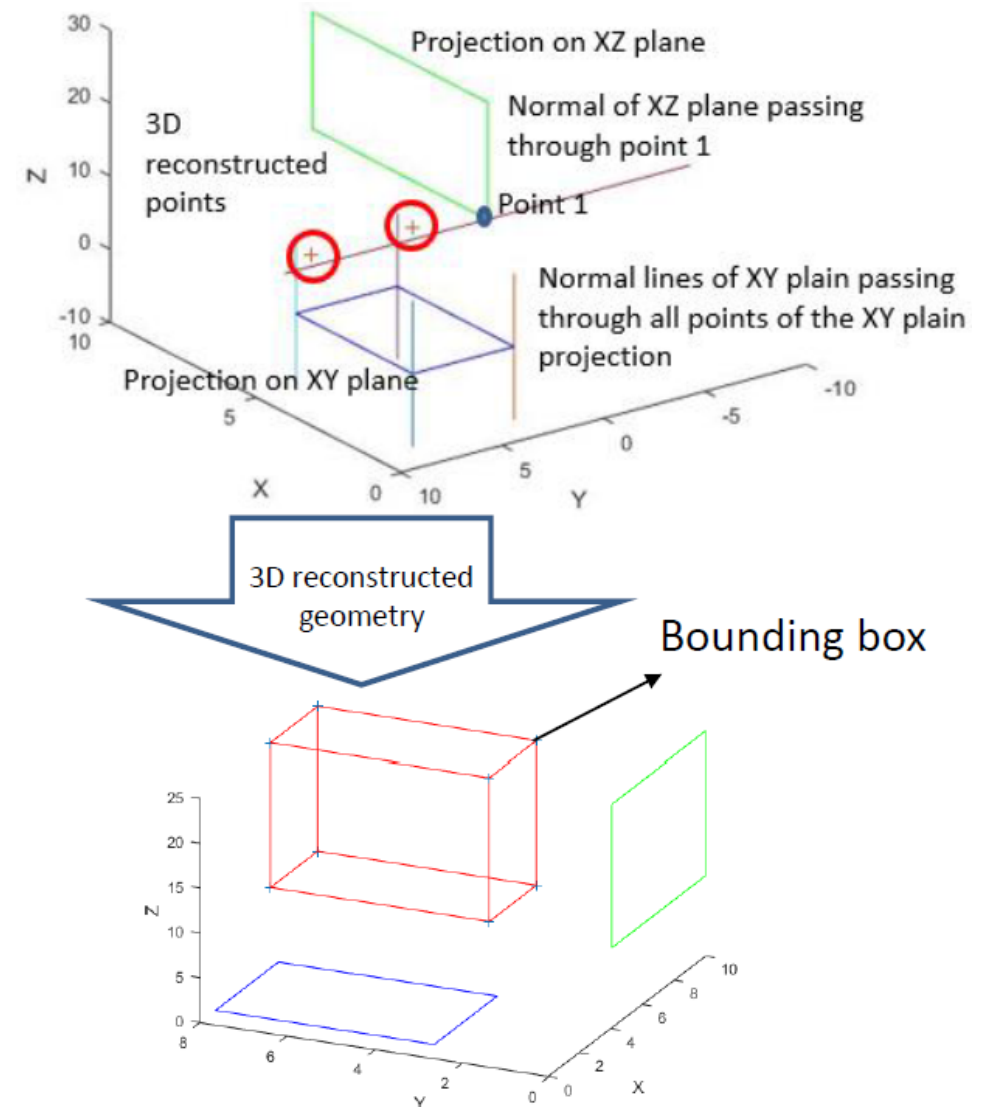
Load segmented data (coordinates of the points)

For each projected plane, set the normal line direction

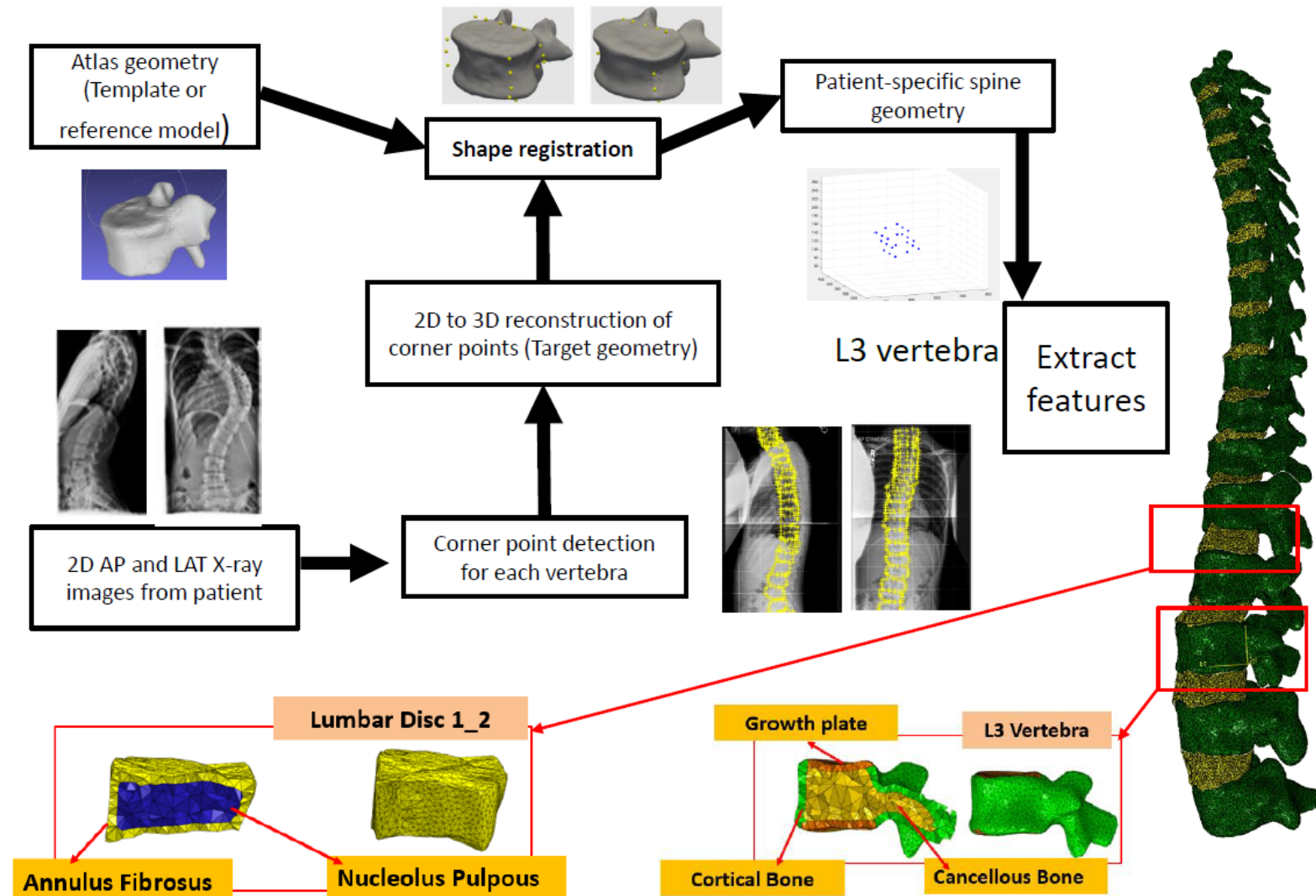
The normal lines are drawn passing through all points (landmarks) of the projected planes

The program will find the minimum distances between the normal lines and sorting them based on the distances

The Intersection of normal lines will be calculated using the mean square error



Feature extraction and geometry generation for spine

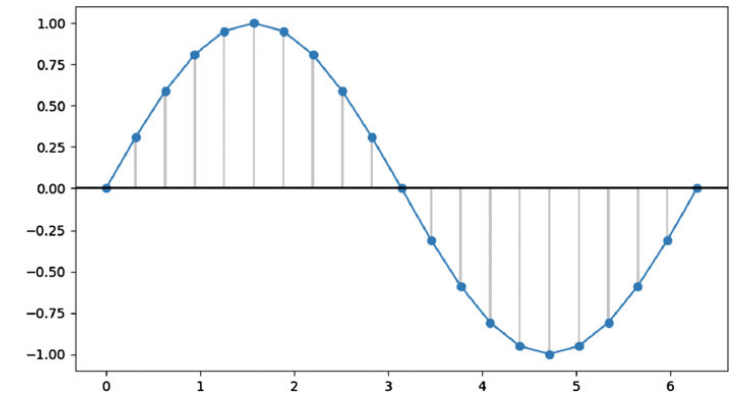
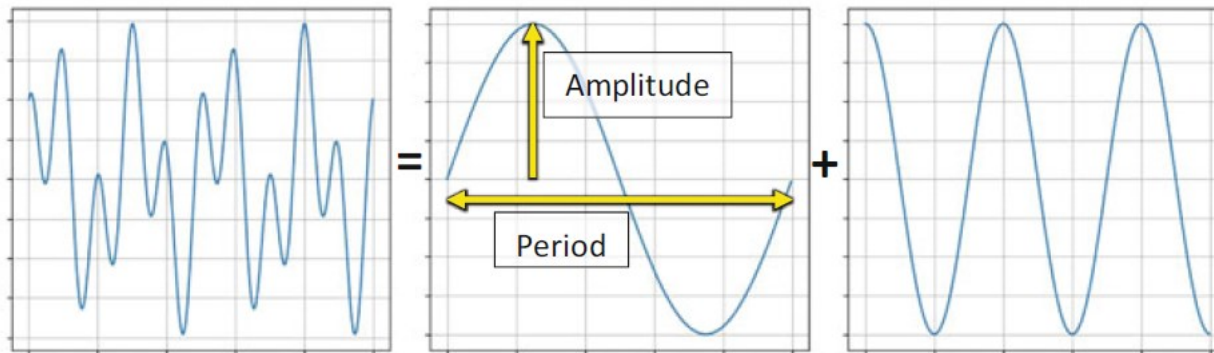


Signals and Signal Processing

- Signal: time-varying or space-varying impulse that has a meaning or stores some information
 - high-pitched siren from an ambulance is a **time**-varying signal intended to get peoples' attention.
 - A photograph from a family vacation is a **space**-varying signal composed of color and intensity
- Calculus: invented independently by both Isaac Newton and Gottfried Leibniz
- Fourier transform: developed by Joseph Fourier (1768-1830)
 - decomposed a dynamic signal into its component **frequencies** and **amplitudes**
 - periodic dynamic functions can be represented as a sine and cosine series called a Fourier series

Fourier Transform (FT)

- everything from sounds to photographs can be described in terms of waves
 - it is assumed that there is a periodic (continuously repeating) pattern
 - Amplitude, period, frequency

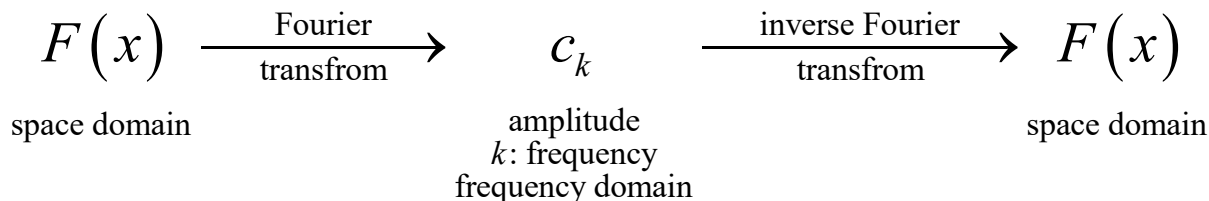


$$\left\{ \begin{array}{l} g(t) = a_0 + \sum_{i=1}^n a_i \sin(i\omega t) + \sum_{i=1}^n b_i \cos(i\omega t) \\ a_n = \frac{\omega}{\pi} \int_0^{\frac{2\pi}{\omega}} g(t) \sin(n\omega t) dt \\ b_n = \frac{\omega}{\pi} \int_0^{\frac{2\pi}{\omega}} g(t) \cos(n\omega t) dt \end{array} \right. \rightarrow \left\{ \begin{array}{l} g(t) = A_0 + \sum_{i=1}^n A_i \sin(i\omega t + \theta_i) \\ A_n^2 = a_n^2 + b_n^2 \\ \theta_n = \tan^{-1} \left(\frac{b_n}{a_n} \right) \end{array} \right.$$

Fourier Series and Transform

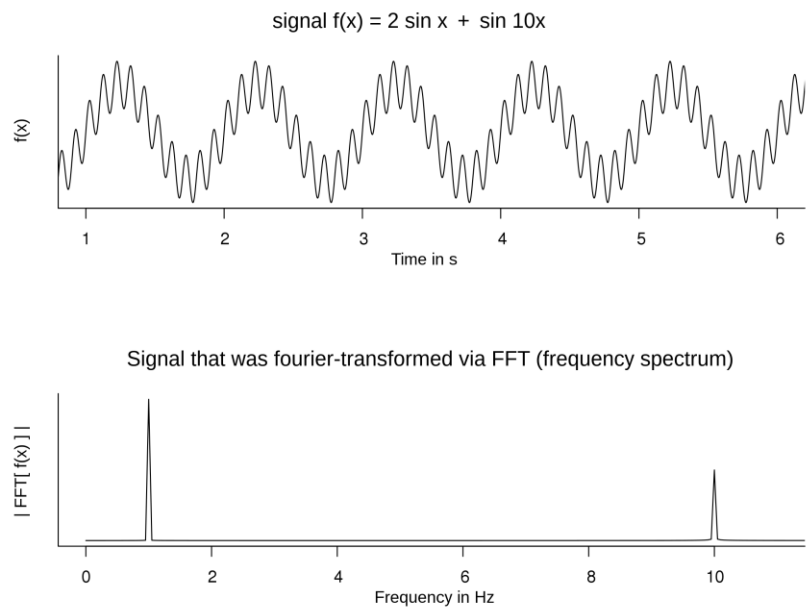
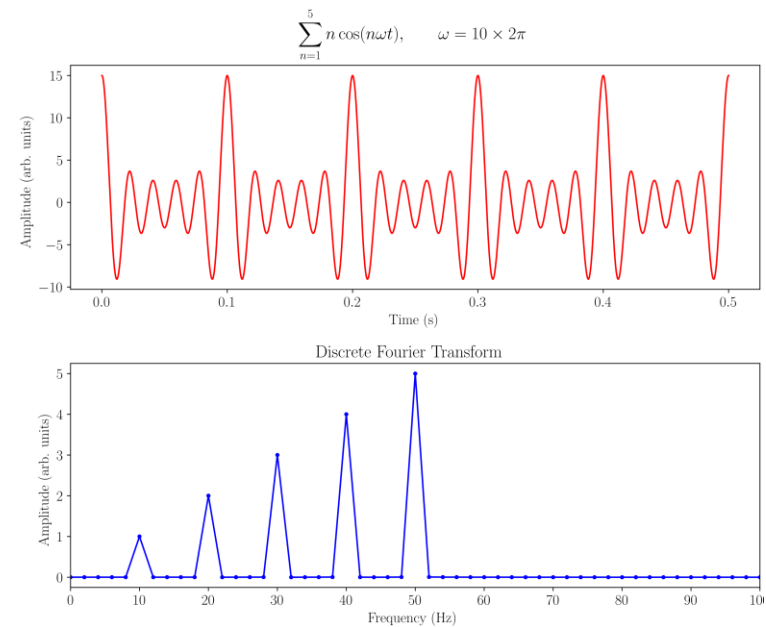


$$\left\{ \begin{array}{l} F(x) = \sum_{n=-\infty}^{\infty} c_n e^{inx} = c_0 + c_1 e^{ix} + c_{-1} e^{-ix} + \dots \\ \int_{-\pi}^{\pi} F(x) e^{-ikx} dx = 2\pi c_k \rightarrow c_k = \frac{1}{2\pi} \int_{-\pi}^{\pi} F(x) e^{-ikx} dx \end{array} \right.$$



$$\left\{ \begin{array}{l} S(x) = \sum_{n=1}^{\infty} b_n \sin nx \quad \text{where } S(x+2\pi) = S(x), S(-x) = -S(x) \\ \int_0^{\pi} S(x) \sin kx dx = b_k \frac{\pi}{2} \rightarrow b_k = \frac{1}{\pi} \int_{-\pi}^{\pi} S(x) \sin kx dx \\ C(x) = a_0 + \sum_{n=1}^{\infty} a_n \cos nx \quad \text{where } C(-x) = C(x) \\ \int_0^{\pi} C(x) dx = a_0 \pi \rightarrow a_0 = \frac{1}{2\pi} \int_{-\pi}^{\pi} C(x) dx \\ \int_0^{\pi} C(x) \cos kx dx = a_k \frac{\pi}{2} \rightarrow a_k = \frac{1}{\pi} \int_{-\pi}^{\pi} C(x) \cos kx dx \\ F(x) = a_0 + \sum_{n=1}^{\infty} a_n \cos nx + \sum_{n=1}^{\infty} b_n \sin nx = C(x) + S(x) \\ a_0 = \frac{1}{2\pi} \int_{-\pi}^{\pi} F(x) dx \\ a_k = \frac{1}{\pi} \int_{-\pi}^{\pi} F(x) \cos kx dx \\ b_k = \frac{1}{\pi} \int_{-\pi}^{\pi} F(x) \sin kx dx \end{array} \right.$$

Fast Fourier Transform

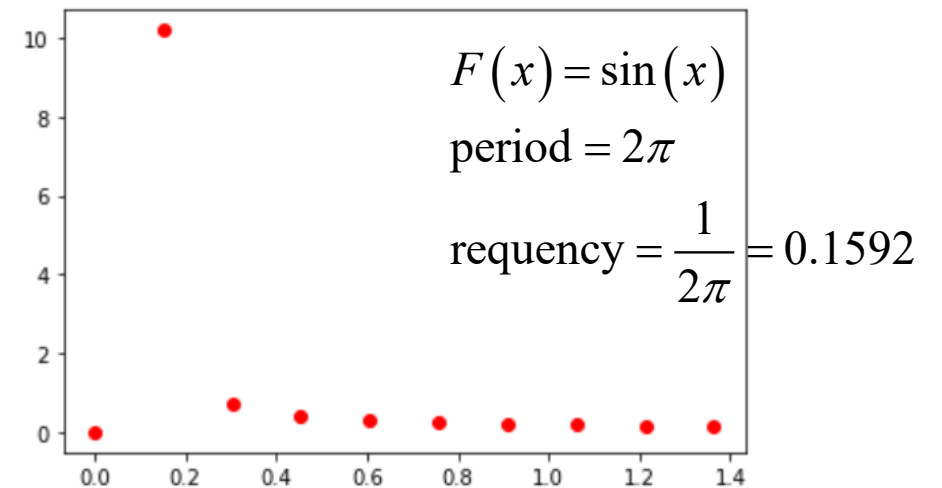
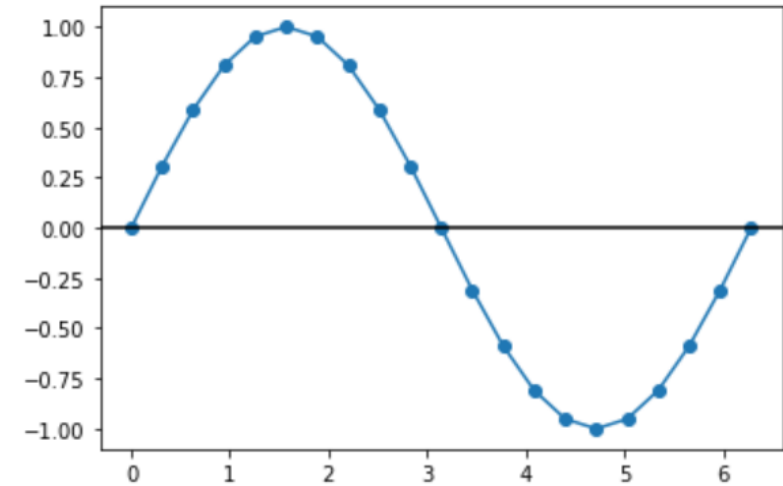


Language	Command/Method	Pre-requisites
R	<code>stats::fft(x)</code>	None
Scilab	<code>fft(x)</code>	None
Octave/MATLAB	<code>fft(x)</code>	None
Python	<code>fft.fft(x)</code>	numpy or scipy
Mathematica	<code>Fourier[x]</code>	None
Fortran	<code>fftw_one(plan,in,out)</code>	FFTW
Julia	<code>fft(A [,dims])</code>	FFTW
Rust	<code>fft.process(&mut x);</code>	rustfft ↗
Haskell	<code>dft x</code>	fft ↗

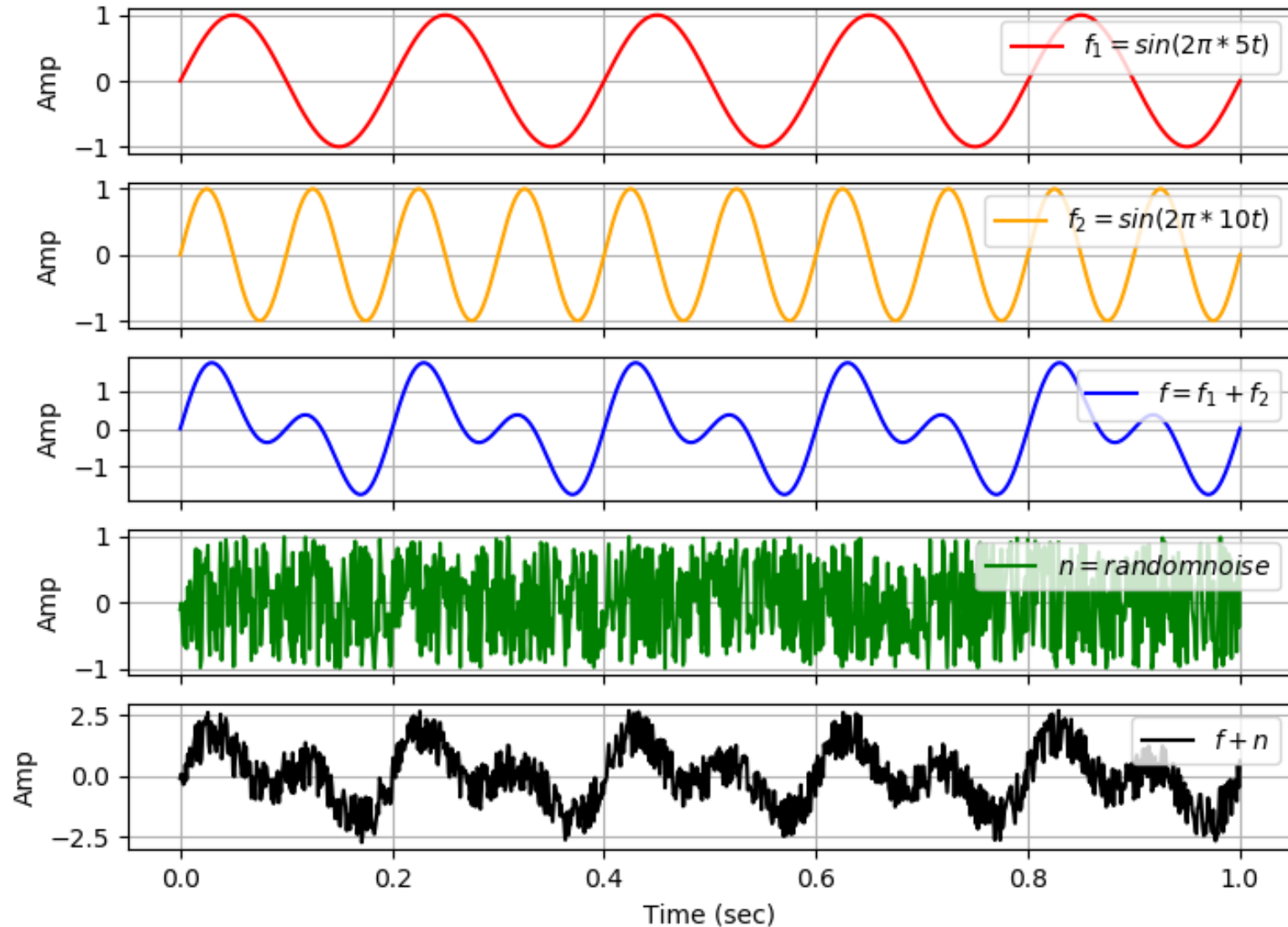
FFT of Sine Wave Signal (Fig.4.25)

```
import scipy.fft
import numpy as np
from matplotlib import pyplot as plt

# Set up sine wave signal
tt = np.linspace(0, 2*np.pi, 21)
yy = np.sin(tt)
plt.plot(tt, yy, '-o')
plt.axhline(y=0, color='k')
N = yy.shape[0]
FFT = abs(scipy.fft.fft(yy))
FFT_side = FFT[range(N//2)] # one side FFT range
freqs = scipy.fft.fftfreq(yy.size, tt[1]-tt[0])
fft_freqs = np.array(freqs)
freqs_side = freqs[range(N//2)] # one side frequency range
plt.figure()
p3 = plt.plot(freqs_side, abs(FFT_side), "ro")
#p3 = plt.semilogy(freqs_side, abs(FFT_side), "b")
```

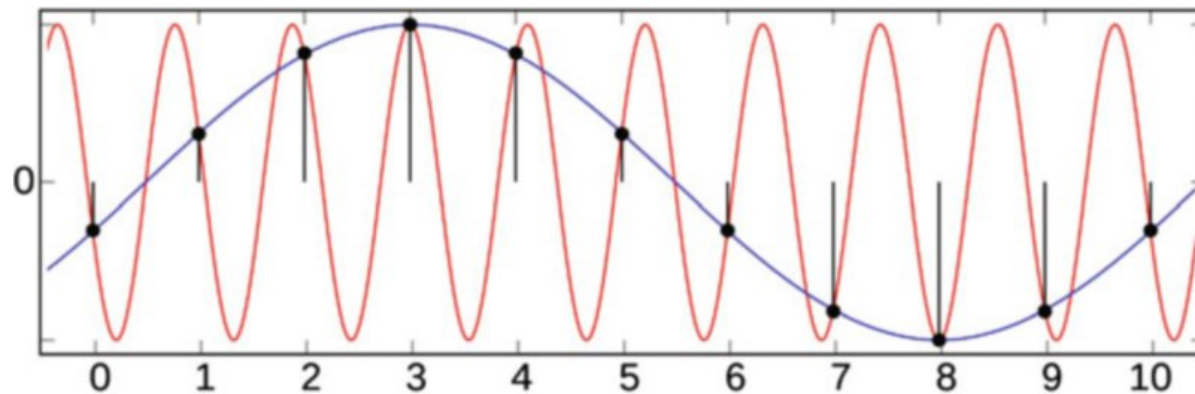


Example: Analysis of Separate and Combined Signals

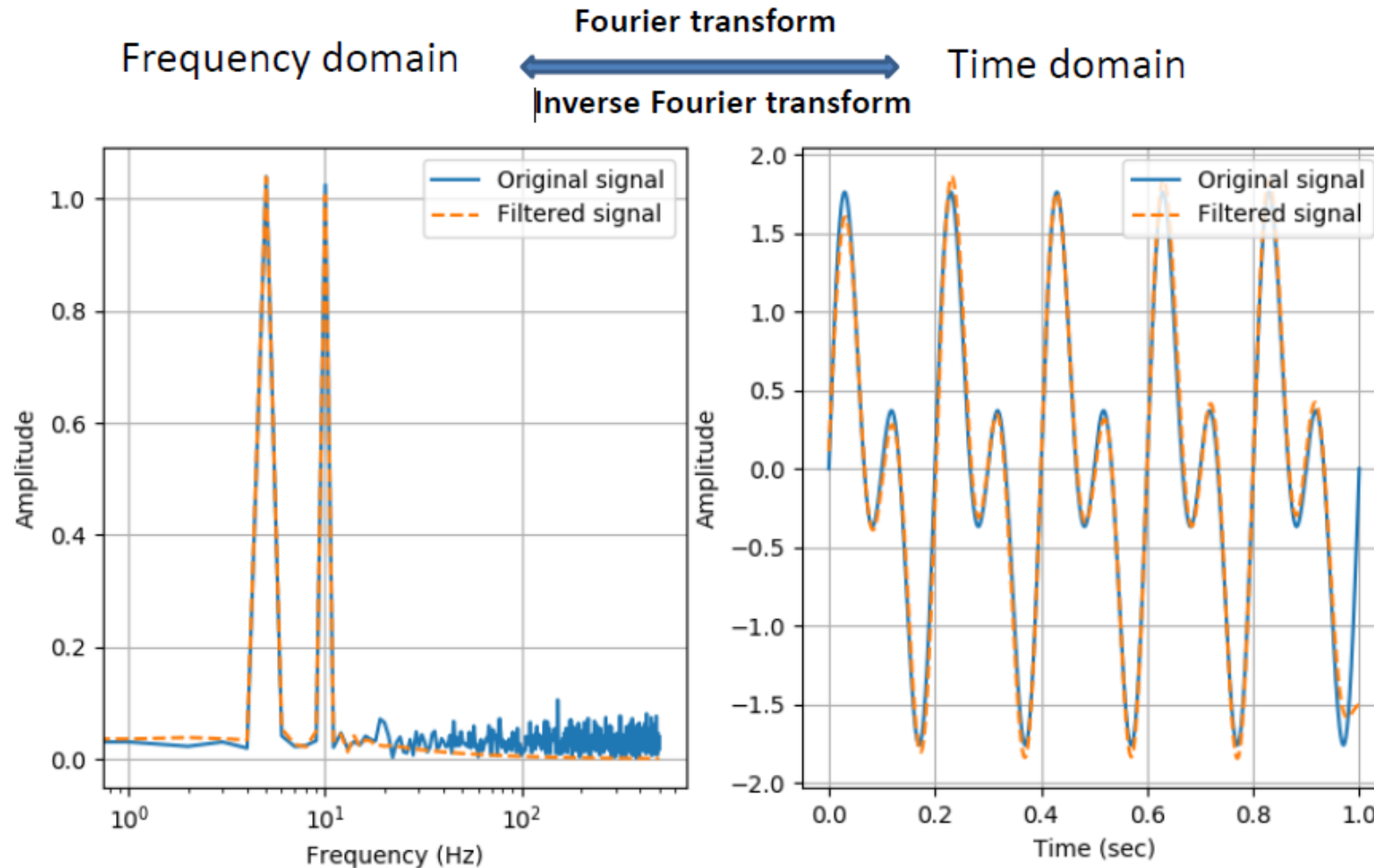


Example: Analysis of Separate and Combined Signals

- Aliasing: inability to distinguish between signals due to inadequate sampling frequency
 - sampling data rate must be at least half of the frequency of the curve being sampled
 - critical sampling rate is called the Nyquist frequency
- Filter: device or process that removes some selected frequencies from a signal
 - Low-pass, high-pass, band-pass



Example: Analysis of Separate and Combined Signals



A lowpass Butterworth filter: remove all frequencies above 15 Hz, leaving only the two dominant peaks corresponding to $f_1(t)$ and $f_2(t)$

Analysis of Sound Waves from a Piano

- A piano makes music based on vibration of strings of different lengths and thicknesses inside the body of the piano
- The piano keyboard consists of 88 white and black keys and around 230 strings (the total number of strings can vary from one piano-maker to another)
- When a key is struck, a felt-tipped hammer → vibration of a wire (or wires) → bridge → soundboard → wave
- different frequencies based on the diameter and length of the wire(s) and the tension in the wire(s): 27.5~4186 Hz

Terminologies of Music

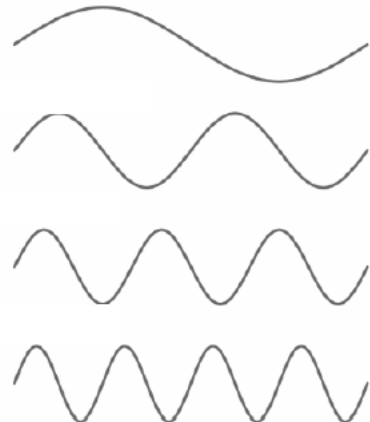
- In music, a note is a symbol denoting a musical sound.
- A volume is the subjective perception of sound pressure.
- Pitch can be identified as the frequency of a tune.
- Sustain is the speed or pace of a given musical piece.
- The lowest frequency of any vibrating object is called the fundamental frequency.
- Harmonics are integer multiples of the fundamental frequency with lower amplitudes. For musical recording there can be inharmonic frequencies which are non integer multiples of fundamental frequency.

$$\omega = \omega_0$$

$$\omega = 2\omega_0$$

$$\omega = 3\omega_0$$

$$\omega = 4\omega_0$$



Fundamental frequency

Harmonics (waves having frequencies that are multiples of the fundamental frequency)

The note A4 or La



https://en.wikipedia.org/wiki/Musical_note



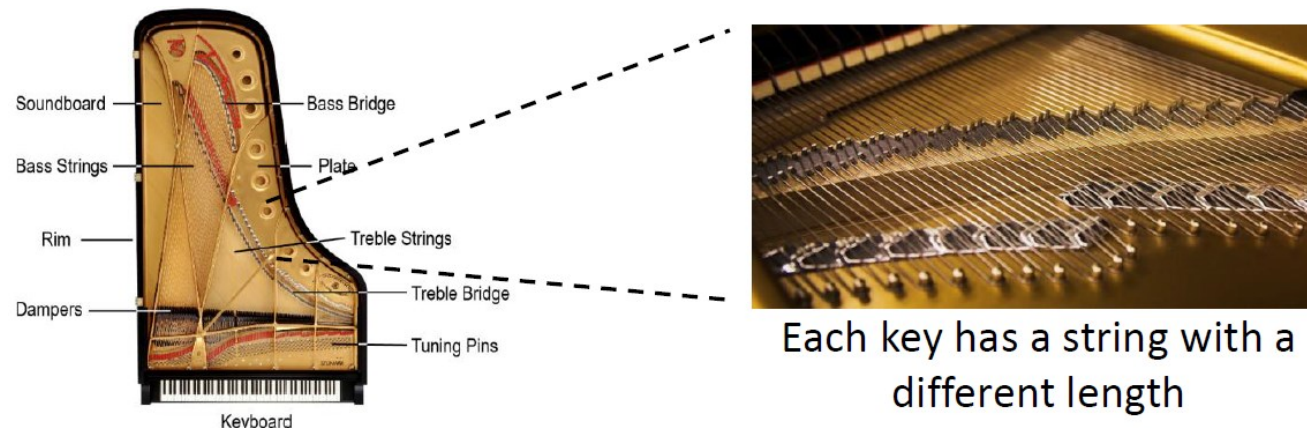
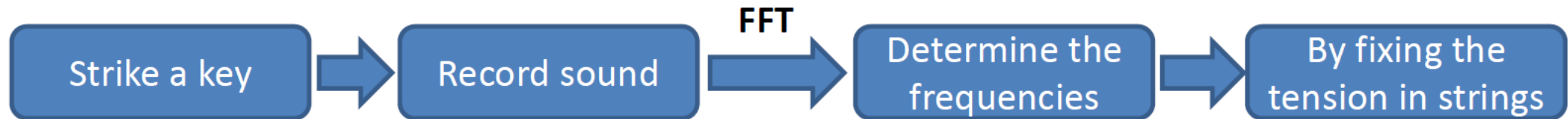
[https://en.wikipedia.org/wiki/Scale_\(music\)](https://en.wikipedia.org/wiki/Scale_(music))



The note A4 has a pitch of 440 Hz

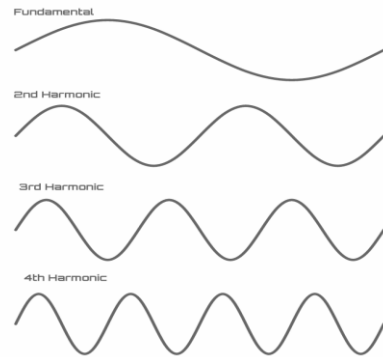
Tale of Tuning a Piano (1)

- When a key is struck, the strings vibrate and generate sound
 - There are about 220 to 240 strings inside a piano.
- To tune a piano, one needs to know the frequencies of the sound generated by adjusting the tension of associated string(s) to get the desired frequency
 - The frequencies of a sound can be identified using Fast Fourier Transform (FFT)

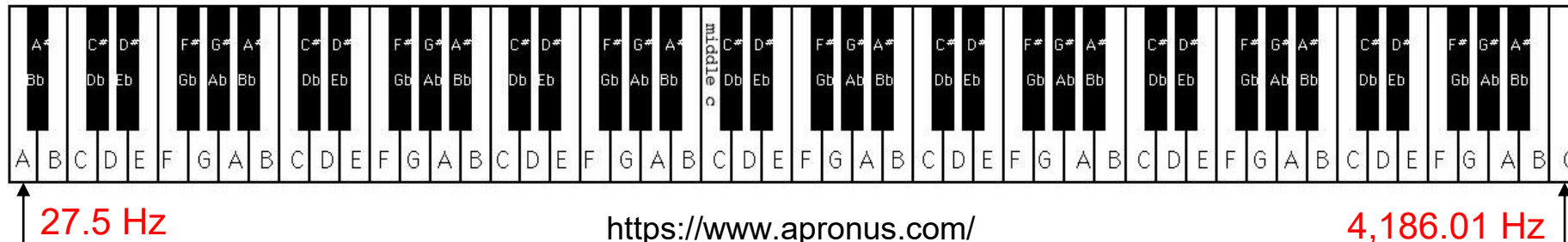


Tale of Tuning a Piano (2)

- Tuning is a big part of playing a piano, usually costs \$65 to \$225
- Tuning means adjusting each key of the piano to a specific pitch (frequency)

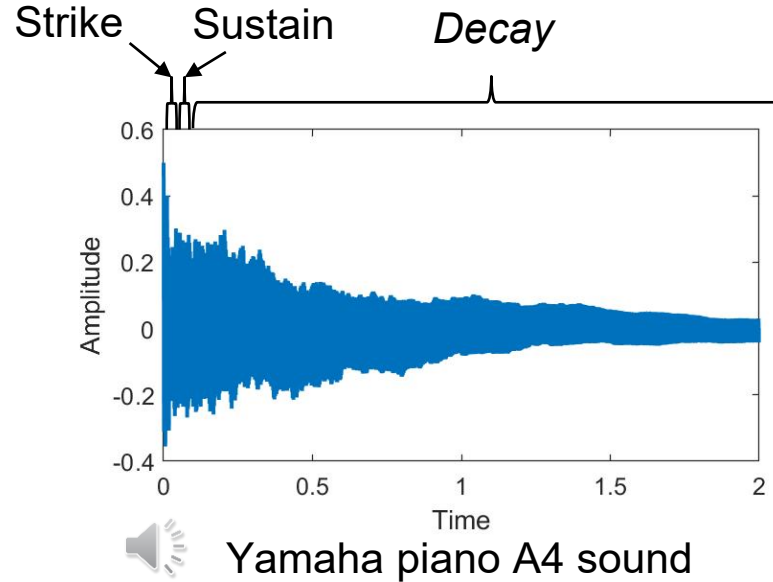
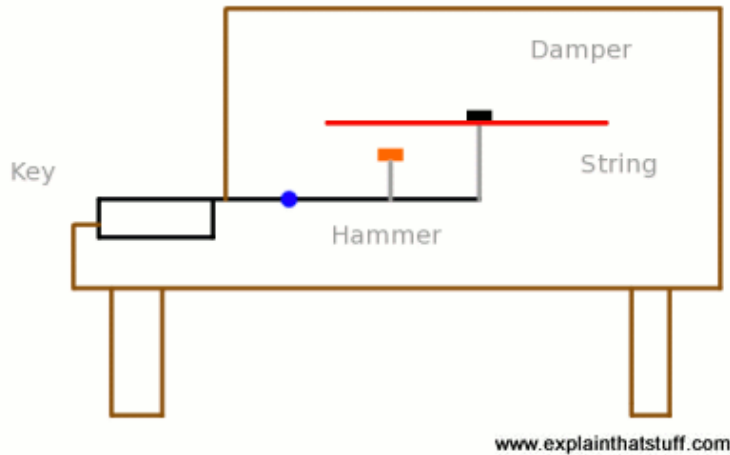


- Fundamental frequency: the lowest frequency of any vibrating object
- Harmonics: waves having frequencies that are integer multiples of the fundamental frequency with lower amplitudes



Different keys are designed to have different frequencies Range: 27.5 Hz to 4,186.01 Hz.

Structural Components of a Piano



Yamaha piano A4 sound

- Sound generation in three stages
 - Strike: hammer hits the string and transmits the energy to the string
 - Sustain: hammer stays on the string
 - Decay: hammer is withdrawn, damper dampens the string vibration
- Assumption
 - Time of strike and sustain stage in piano is very short
 - Amplitude: relative scale of air pressure in transmitting the sound

Generate a Piano Sound (Fig.4.36, Matlab→Python)

```
function generate_piano(fs,tn,a,omega)
%This function is to generate a sound file with a series of
%sine waves
%to approximate the piano sound.
%ts: time integral in a sound
%tn: duration of the sound
%a: amplitude of each sine wave, should be an 1 by l vector.
%omega: frequency of each sine wave, should be an 1 by l
%vector.
t=0:1/fs:tn; %Generate the time array
n=size(a); %Count the number of sine waves
y=zeros(size(t));%Define signal
for i=1:1:n
y=y+a(i).*sin(2.*pi.*omega(i).*t);
end
audiowrite('piano.wav',y,fs); %Write the sound file
```

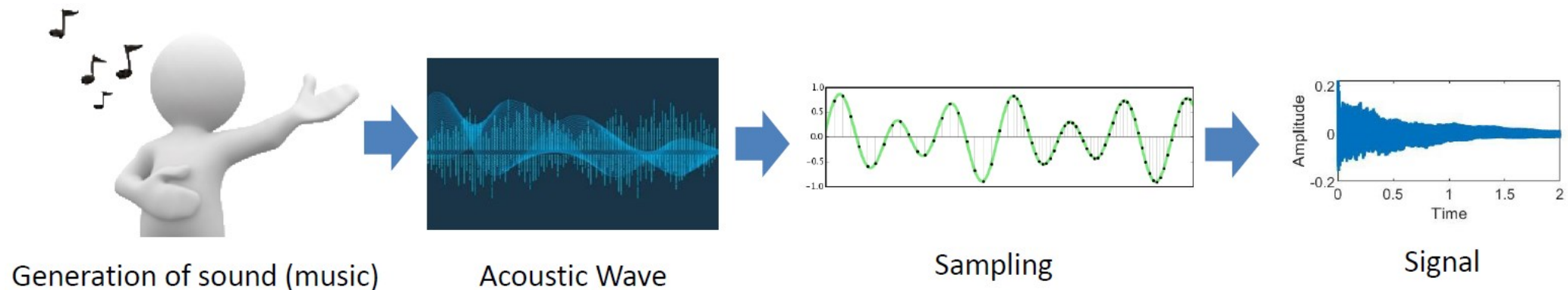
```
import numpy as np
from scipy.io import wavfile

def generate_piano_sound(fs, tn, a, omega):
    t = np.arange(0, tn, 1/fs) # Generate the time array
    n = len(a) # Count the number of sine waves
    y = np.zeros(len(t)) # Define signal
    for i in range(n):
        y += a[i] * np.sin(2 * np.pi * omega[i] * t)
    wavfile.write('piano.wav', fs, y.astype(np.float32)) # Write the sound file

fs = 44100 # Sampling frequency
tn = 2 # Duration of sound in seconds
a = [0.5, 0.3, 0.2] # Amplitude of each sine wave
omega = [440, 880, 1320] # Frequency of each sine wave
generate_piano_sound(fs, tn, a, omega) # Generate and write the sound file
```

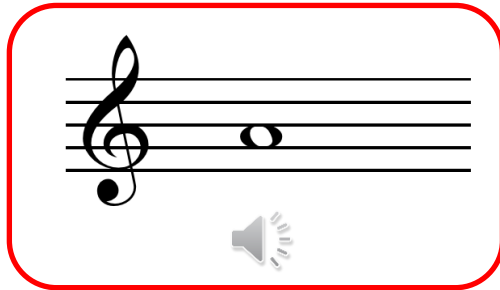
How to Convert a Sound to a Signal?

- A software package, such as MATLAB, provides tools to convert the sound file to a signal for mathematical analysis
- How does waveread() work?
 - The waveread () function takes a sound file in wave format
 - The file is converted to binary format (waveread () reads in 32 bit)
 - A sampling frequency is fixed at which the data acquisition occurs.
 - A vector of amplitudes and corresponding sampling times is expressed as signal



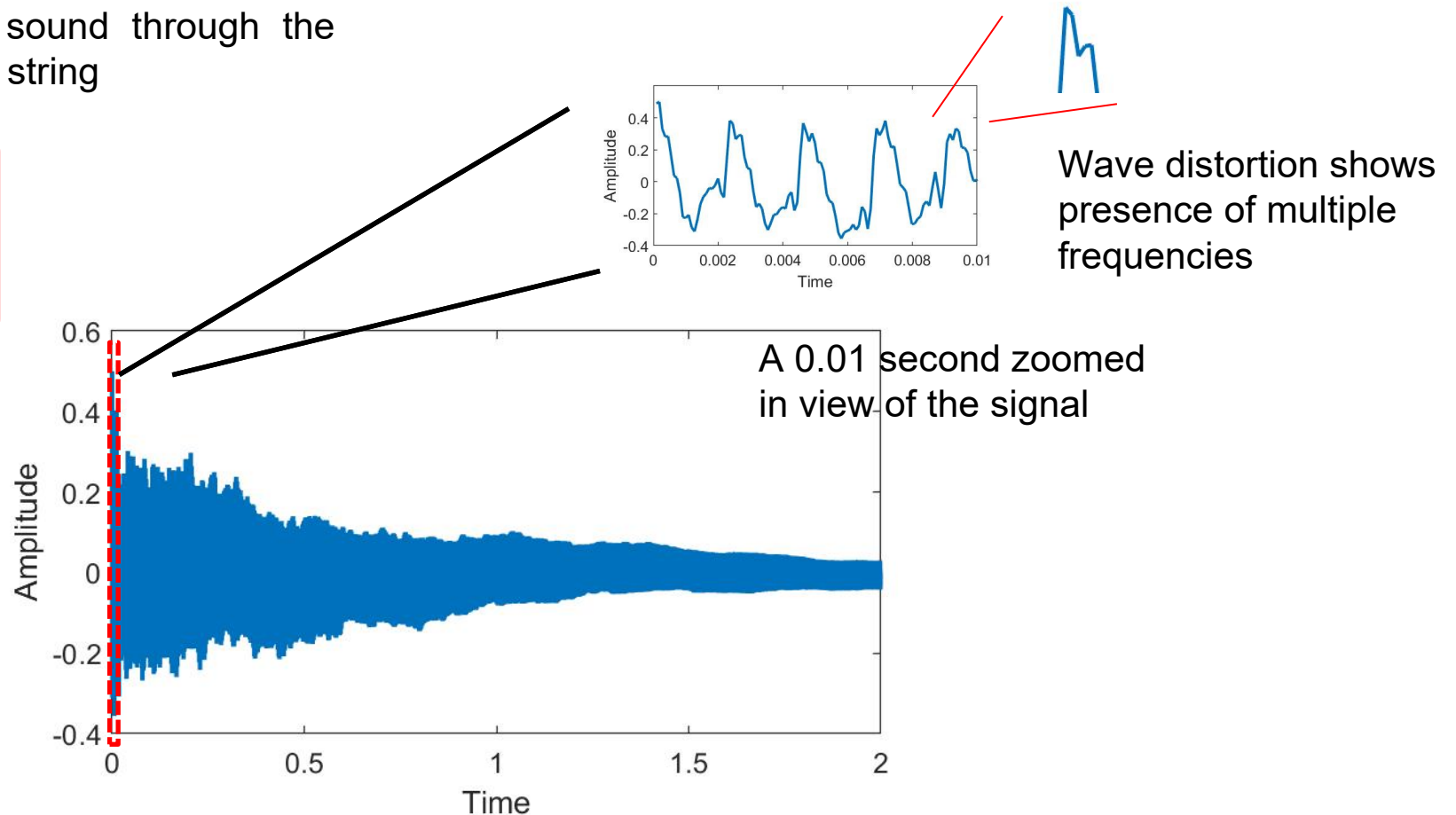
Multiresolution in a Signal

A strike piano key emits sound through the vibration of a pre-tensioned string



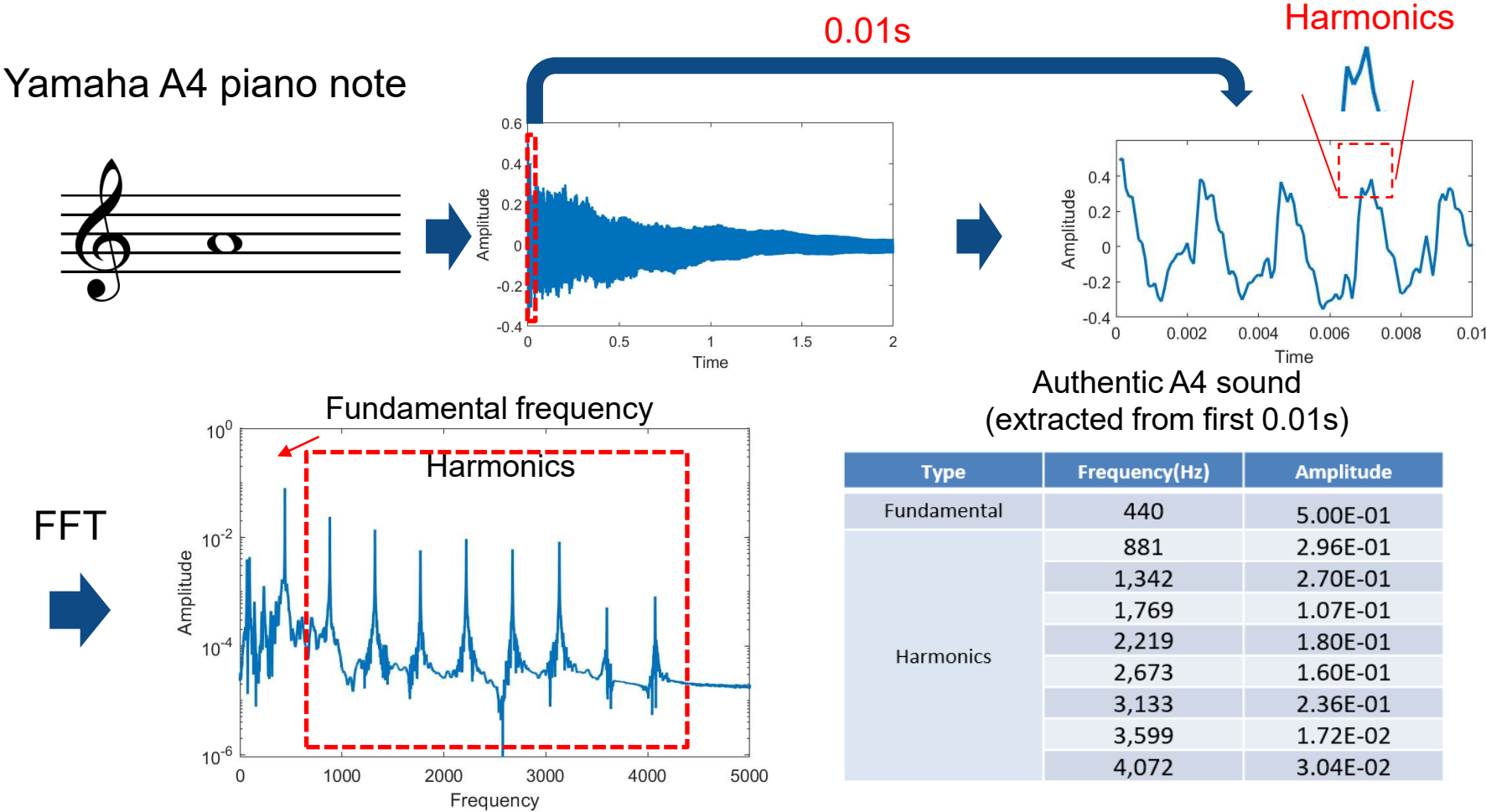
(observations)

- The sound wave consists of multiple frequencies
- The amplitude of the signal changes with time.
- There is a decaying pattern in the signal, which means sound becomes muffled with time.



- investigate the acoustic features using Fast Fourier Transform (FFT)
- Acoustic features: frequency, amplitude

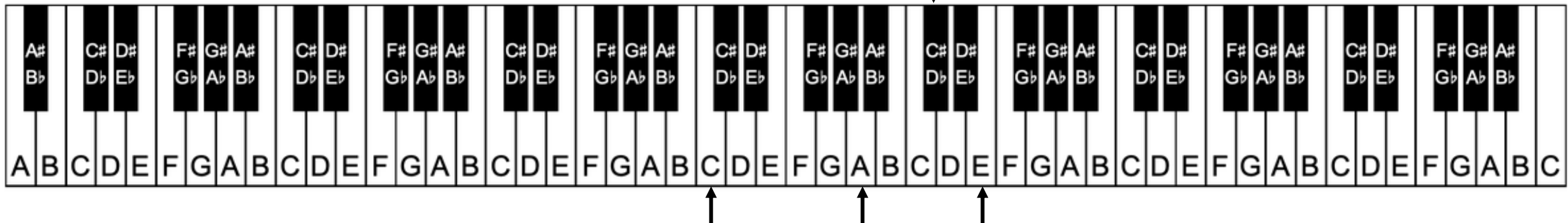
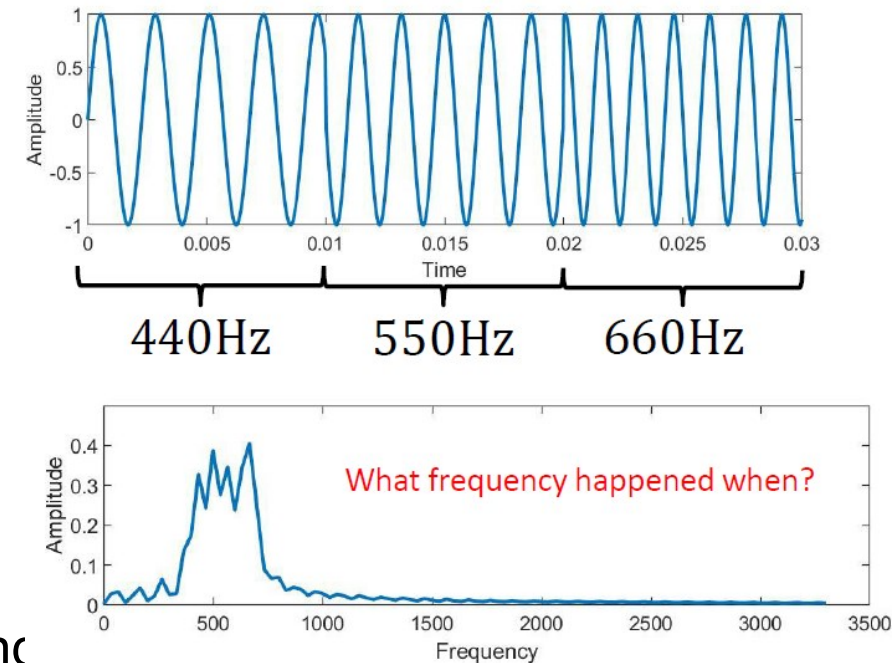
Feature Extraction of the Sound Signal Using Fast Fourier Transform (FFT)



Fourier transform can **extract** the frequency and amplitude **features** of fundamental and (eight) harmonic frequencies of the piano sound.

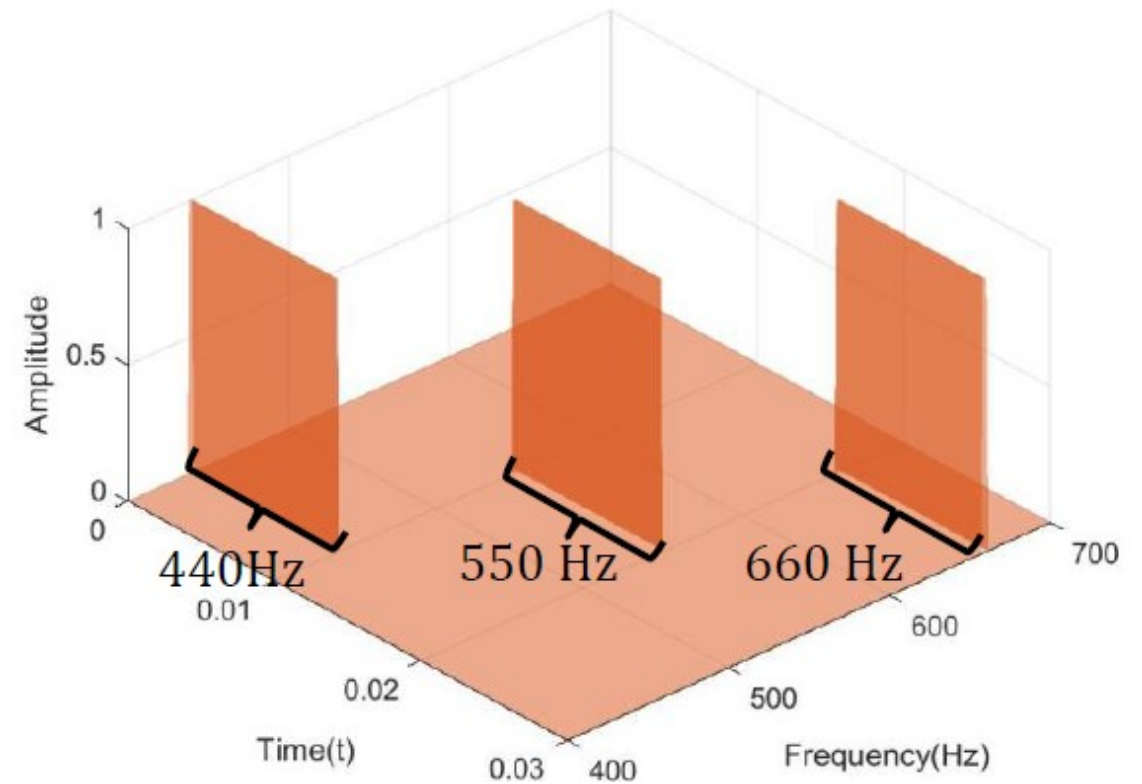
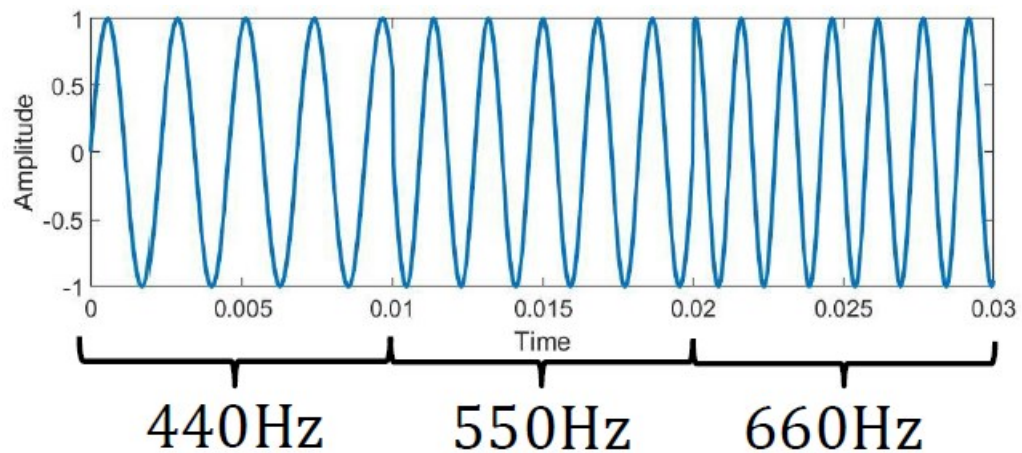
Why we need short time Fourier transform?

- Limitations of Fourier Transform: Not very useful for analyzing time variant, non-stationary signals
 - If we press *A5, C#6 and E6* each for 1s
 - The sound wave (sine function) is:
 - (To better show the plot, we define each frequency happened 0.01s)
 - If we use Fourier transform to analysis the wave, we cannot know how frequencies change with time.
 - We can use Short time Fourier transform to analysis the sound



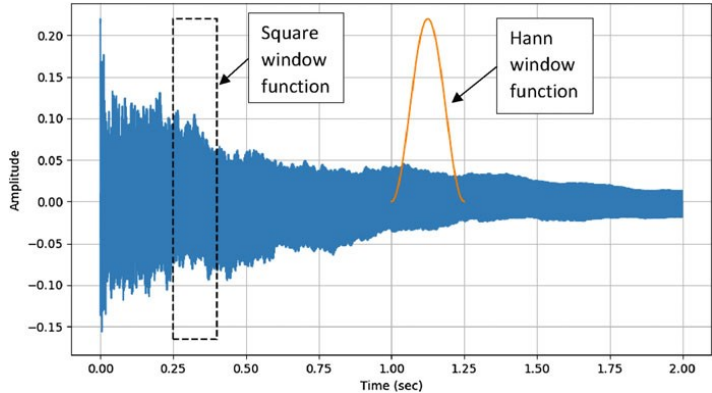
Short time Fourier transform steps

- STFT presents the frequencies with different time, so it can tell us **how frequencies change with time**.

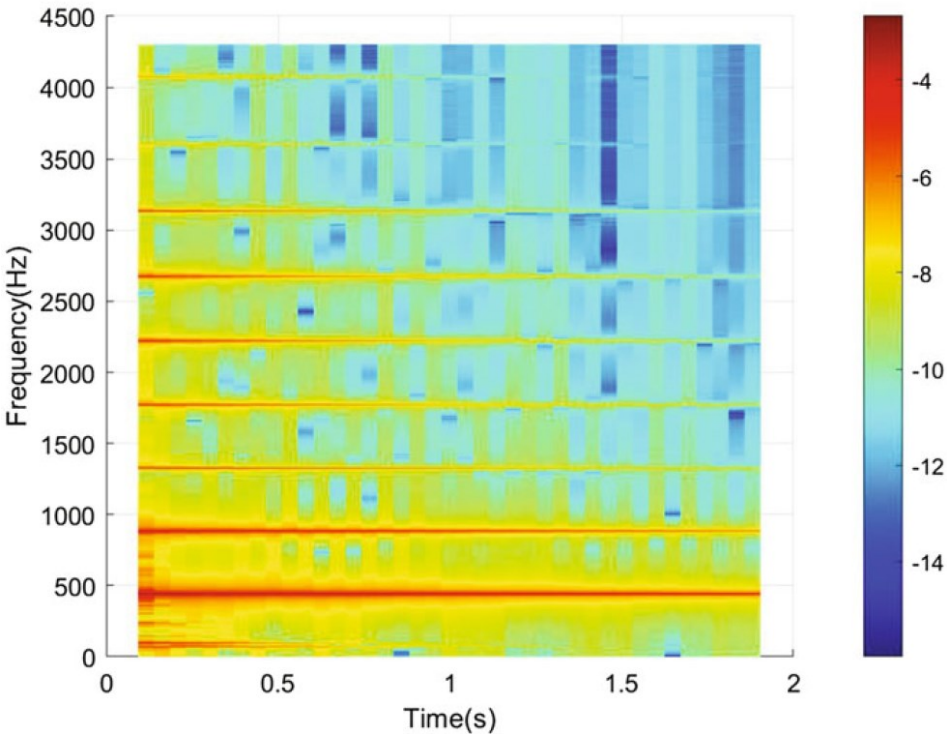


Use Short Time Fourier transform to analyze the piano sound

- Short Time Fourier transform shows the how the fundamental frequency and each harmonic rise and damping with time.

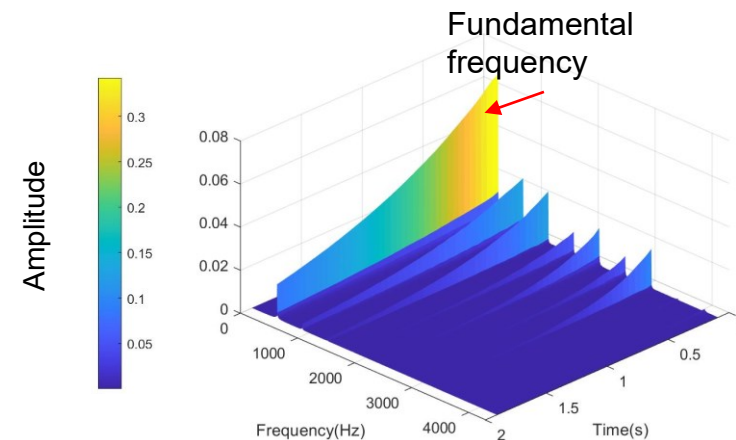
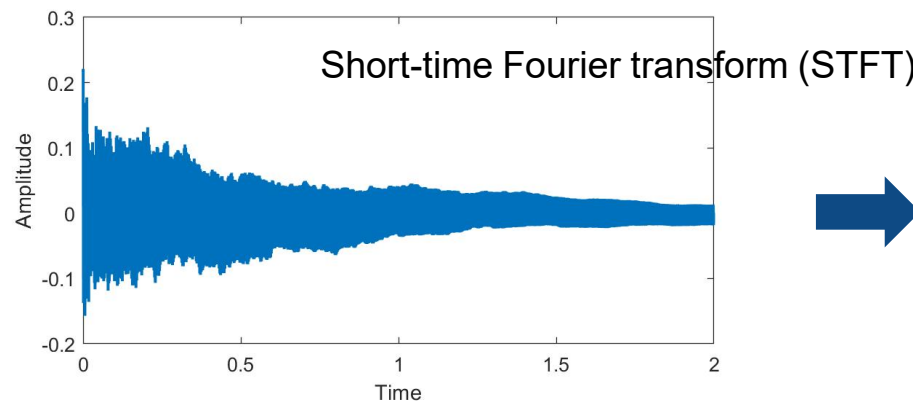


Type	Frequency(Hz)	Averaged Amplitude
Fundamental	440	0.1
Harmonics	881	0.006352
	1322	0.001489
	1762	0.0009335



Improvement of Feature Extraction

- To enhance the reconstruction of the authentic A4 key, Short Time Fourier Transform (STFT) is used to reveal the strike, sustain and decay.
- Short-time Fourier transform (STFT)
 - Fourier-related transform used to determine the sinusoidal frequency and phase content of local sections of a signal as it changes over time



Signal

strike, sustain and decay

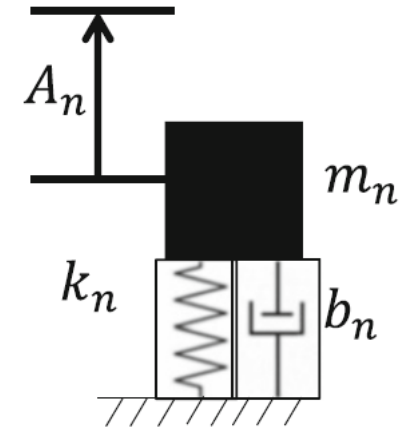
Mechanistic Equation in terms of ODE

- A mass-spring-damper model without external forces can be defined as:

$$m \frac{d^2 f}{dt^2} + b \frac{df}{dt} + kf = 0$$

$$f(t) = Ae^{-bt} \sin\left(\left(\omega\sqrt{1-\zeta^2}\right)t + \phi\right)$$

$$\omega = \sqrt{\frac{k}{m}}, \zeta = \frac{b}{2m\omega}$$



Simple Harmonic Vibration (Oscillator)

undamped case: (often damping is extremely small)

(b) $mg = ku_0$

(c) $mg - k(u_0 + x) = m \frac{d^2 x}{dt^2}$

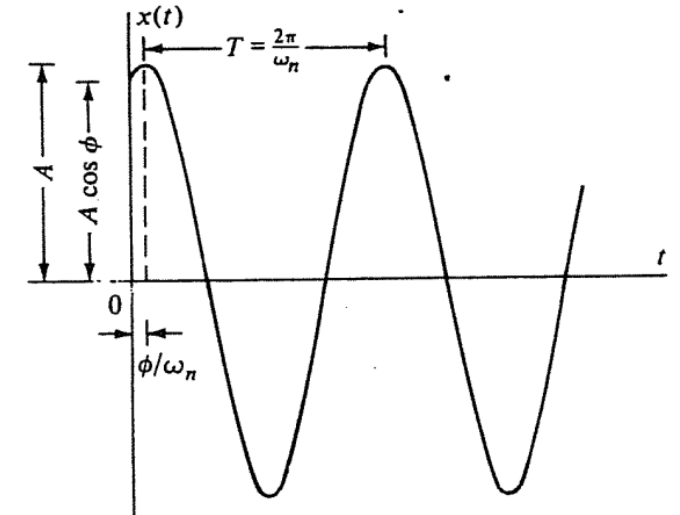
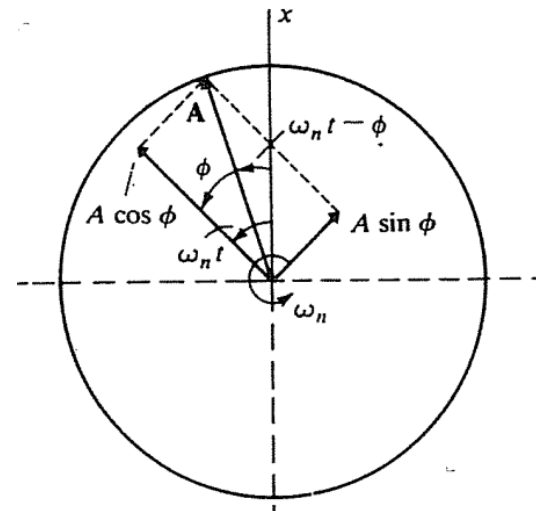
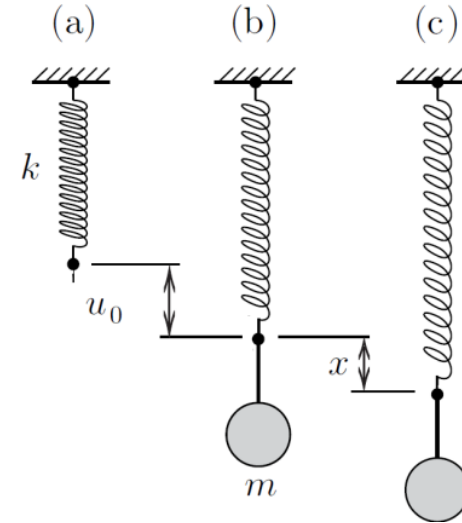
$$\begin{cases} m\ddot{x} + kx = 0 \\ \ddot{x} + \frac{k}{m}x = 0 \end{cases} \rightarrow \begin{cases} x(t) = Ae^{st} \\ s^2 + \omega_n^2 = 0 \rightarrow s = \pm i\omega_n \\ x(t) = A_1 e^{i\omega_n t} + A_2 e^{-i\omega_n t} \end{cases}$$

$$\begin{aligned} x(t) &= A_1 (\cos \omega_n t + i \sin \omega_n t) + A_2 (\cos \omega_n t - i \sin \omega_n t) \\ &= \underbrace{(A_1 + A_2)}_{A \cos \phi} \cos \omega_n t + i \underbrace{(A_1 - A_2)}_{A \sin \phi} \sin \omega_n t = A \cos(\omega_n t - \phi) \end{aligned}$$

$x(0) = x_0$ and $\dot{x}(0) = v_0$

$$\rightarrow A = \sqrt{x_0^2 + \left(\frac{v_0}{\omega_n}\right)^2}, \phi = \tan^{-1} \left(\frac{v_0}{x_0 \omega_n} \right)$$

$$\omega_n = \sqrt{\frac{k}{m}}, T = \frac{2\pi}{\omega_n} = 2\pi \sqrt{\frac{m}{k}}, f = \frac{1}{T} = \frac{1}{2\pi} \sqrt{\frac{k}{m}}$$



Simple Harmonic Vibration (Oscillator)

free vibration of damping

$$m\ddot{x} + c\dot{x} + kx = 0$$

c : coefficient of viscous damping (점성 감쇠 계수)

$$\begin{cases} \ddot{x} + \frac{c}{m}\dot{x} + \frac{k}{m}x = 0 \\ \ddot{x} + 2\zeta\omega_n\dot{x} + \omega_n^2x = 0 \\ \zeta = \frac{c}{2m\omega_n} = \frac{c}{2\sqrt{km}} \end{cases} \rightarrow \begin{cases} x(t) = Ae^{st} \\ s^2 + 2\zeta\omega_ns + \omega_n^2 = 0 \rightarrow s_{1,2} = (-\zeta \pm \sqrt{\zeta^2 - 1})\omega_n \\ x(t) = A_1e^{s_1t} + A_2e^{s_2t} \\ = \left[A_1 \exp(\omega_nt\sqrt{\zeta^2 - 1}) + A_2 \exp(-\omega_nt\sqrt{\zeta^2 - 1}) \right] e^{-\zeta\omega_nt} \end{cases}$$

viscous damping factor (점성 감쇠 비)

(1) $\zeta > 1$ (overdamped) aperiodic and decaying exponentially

(2) $\zeta = 1$ (critical damping) $s_{1,2} = -\omega_n \rightarrow x(t) = (A_1 + tA_2)e^{-\zeta\omega_nt}$

(1) $\zeta < 1$ (underdamped)

$$x(t) = \left[A_1 \exp(i\omega_nt\sqrt{1 - \zeta^2}) + A_2 \exp(-i\omega_nt\sqrt{1 - \zeta^2}) \right] e^{-\zeta\omega_nt}$$

$$= (A_1 e^{i\omega_d t} + A_2 e^{-i\omega_d t}) e^{-\zeta\omega_nt} = Ae^{-\zeta\omega_nt} \cos(\omega_d t - \phi)$$

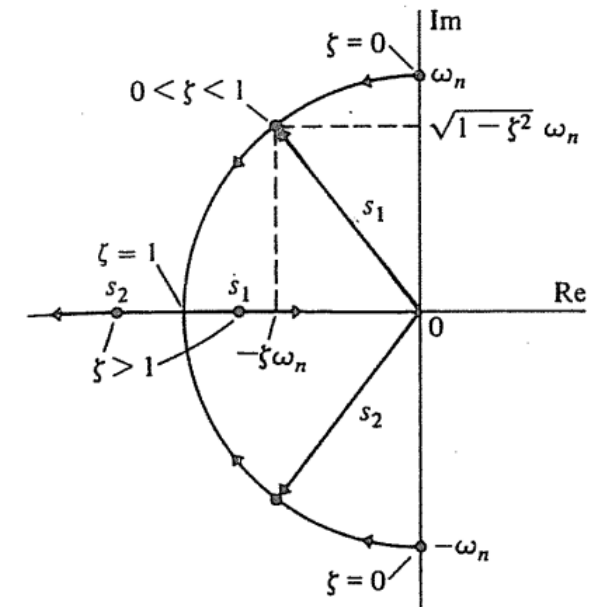
$$m\ddot{x} + c\dot{x} + kx = f$$

$f = 0$: free vibration

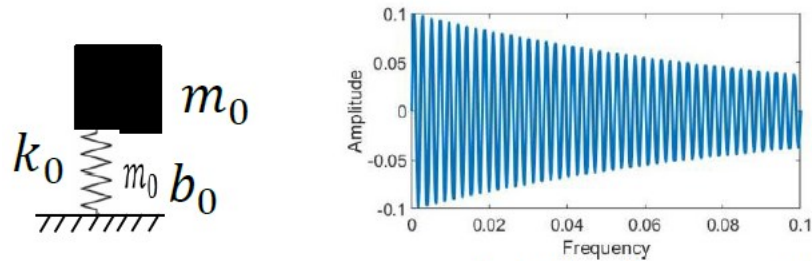
$f \neq 0$: forced vibration

$c = 0$: without damping

$c \neq 0$: with damping

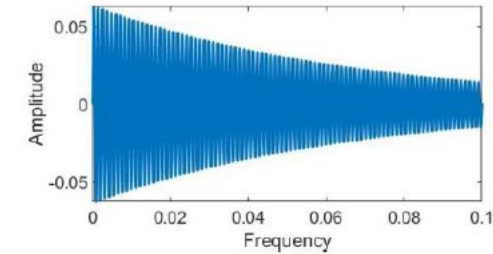
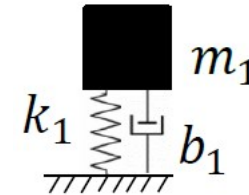


Mechanistic Equation for a Sound from Piano Key



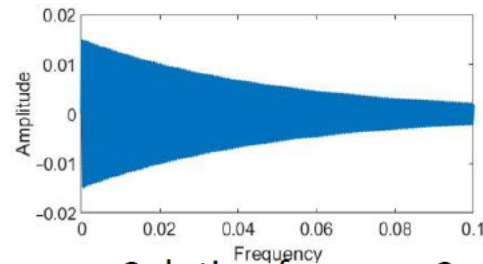
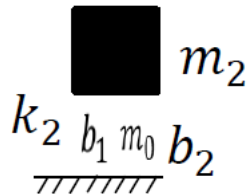
Solution for wave 1

$$f_0(t) = A_0 e^{-b_0 t} \sin\left(\left(\omega_0 \sqrt{1 - \zeta_0^2}\right)t + \phi_0\right)$$

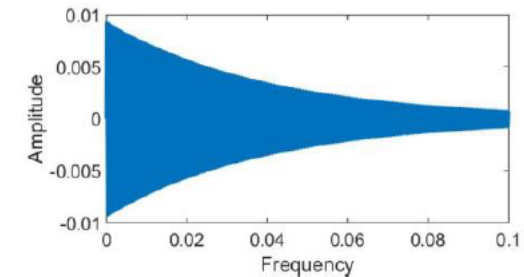
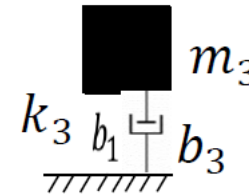


Solution for wave 2

$$f_1(t) = A_1 e^{-b_1 t} \sin\left(\left(\omega_1 \sqrt{1 - \zeta_1^2}\right)t + \phi_1\right)$$



Solution for wave 3

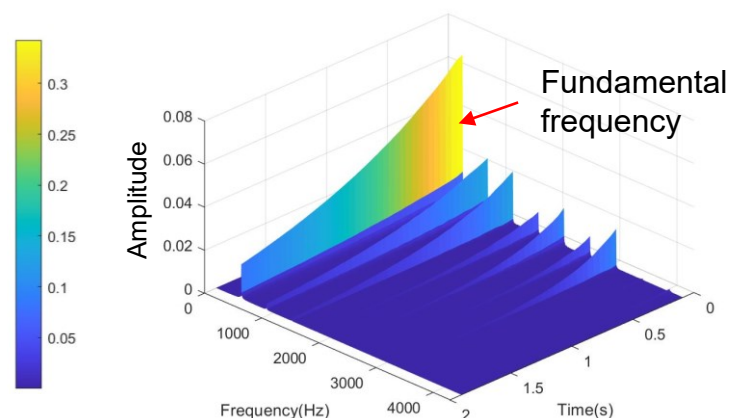


Solution for wave 4

$$f(t) = \sum_{i=0}^I A_i e^{-b_i t} \sin\left(\left(\omega_i \sqrt{1 - \zeta_i^2}\right)t + \phi_i\right)$$

$$\underset{b}{\text{minimize}} \quad \frac{1}{N} \sum_{n=1}^N \left(f_{PIANO}(t_n) - f_{MECHANISTIC}(t_n)\right)^2$$

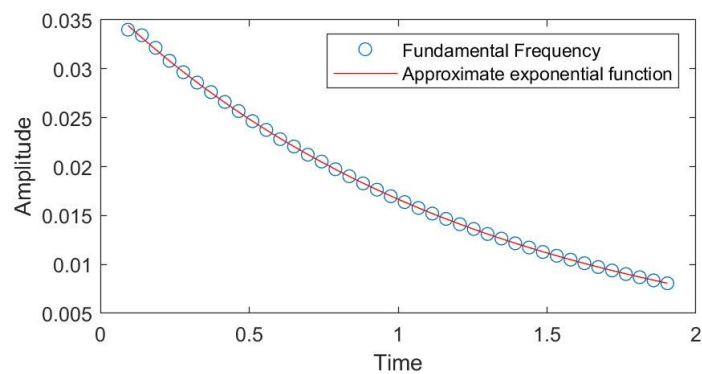
Determination of the Envelop Mechanistic Features



“Yamaha A4” coefficient for fundamental frequency and each harmonic

Type	Frequency(Hz)	Initial amplitudes	Damping coefficients
Fundamental	440	3.710E-02	8.003E-01
Harmonics	881	1.429E-02	1.189E+00
	1,342	4.265E-04	1.594E+00
	1,769	6.342E-03	2.017E+00
	2,219	1.083E-02	2.464E+00
	2,673	4.259E-03	2.542E+00
	3,133	1.178E-02	2.694E+00
	3599	2.200E-04	1.478E+00
	4072	3.700E-04	1.224E+00

Fundamental frequency



$$g_i(t) = A_i e^{-b_i t} \sin(\omega_i t + \phi_i)$$

A_i : initial amplitude

b_i : damping ratio

ω_i : frequency








ϕ_i : phase angle

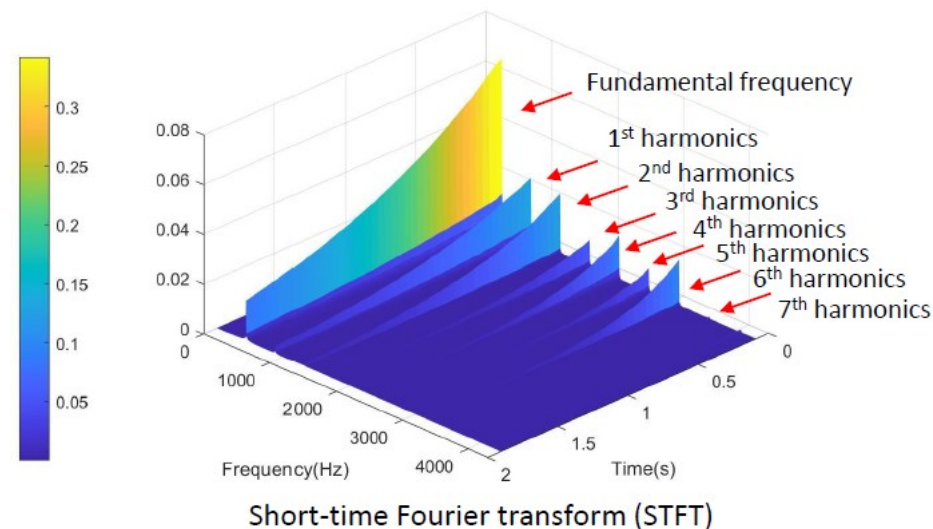
Reconstruction of the Piano Sound

- As more features (harmonics) are added, the reconstructed sound quality improves
- The sound generated with 7 (total of 8) harmonics is closer to the original sound



Yamaha piano A4 sound

No. of harmonics	1	2	3	4	5	6	7	8
Sound								



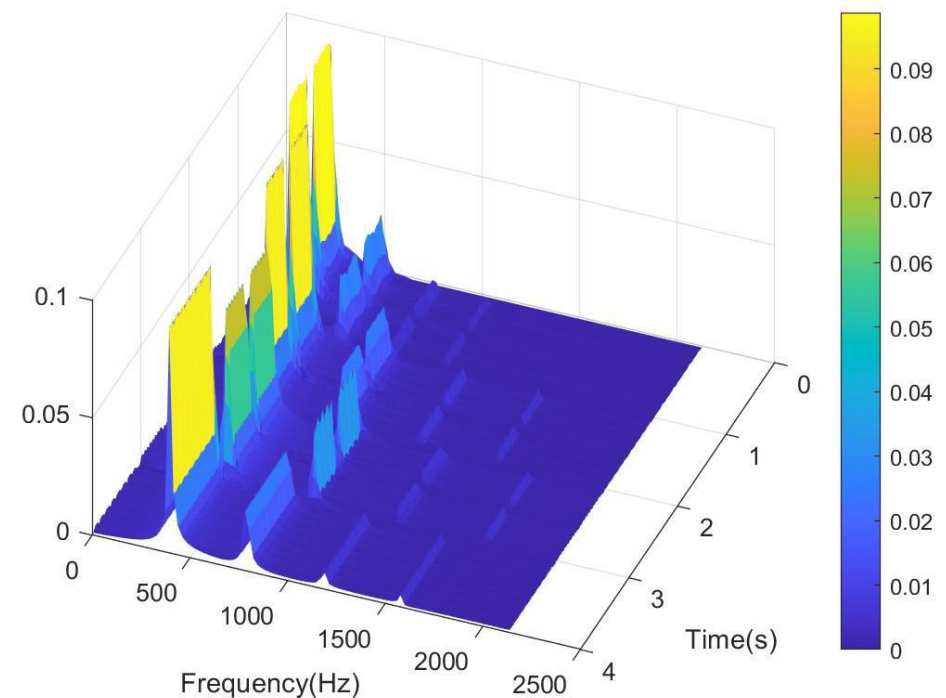
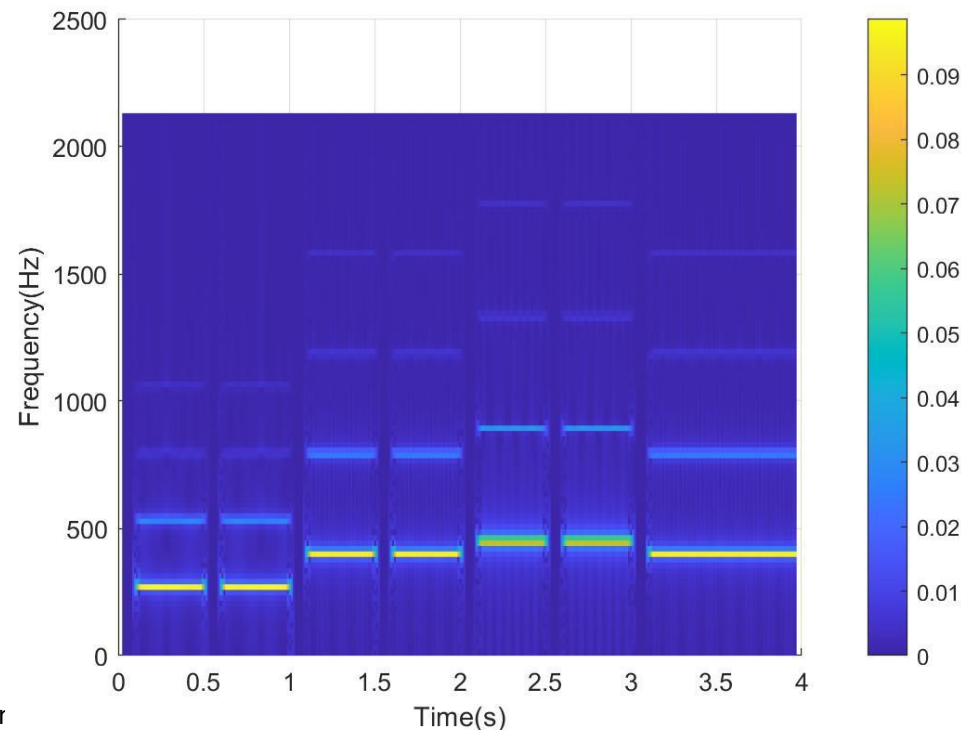
Use Short Time Fourier transform to analyze the piano melody

- A melody is a series of musical notes or tones arranged in a definite pattern of pitch.
- How to write down a piano note by analyzing the music?
- At least, one should know what frequency and amplitude happened at what point of time.
- Short Time Fourier transform can be used to extract frequency, amplitude, and time from the music sound.



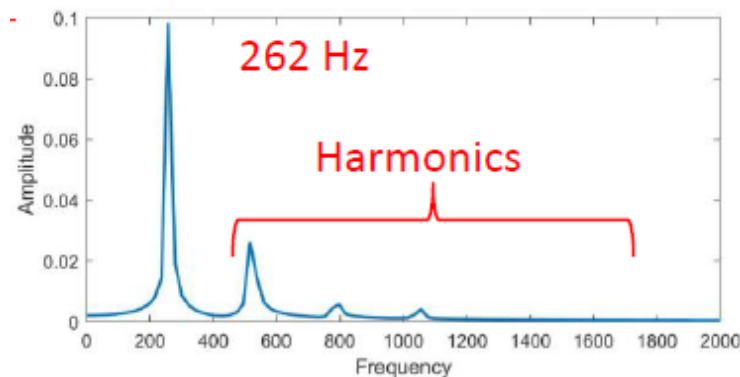
Short Time Fourier transform of “Twinkle twinkle little star”

- For the first second, two keys are pressed with fundamental frequency 262 Hz. The key associate to this frequency is C4.
- For each key, the fundamental frequency and harmonics are considered.
- The height is the amplitude and from the plot, one can tell which key is pressed at what time.
- Short Time Fourier transform shows the features of time, amplitude, and frequency.

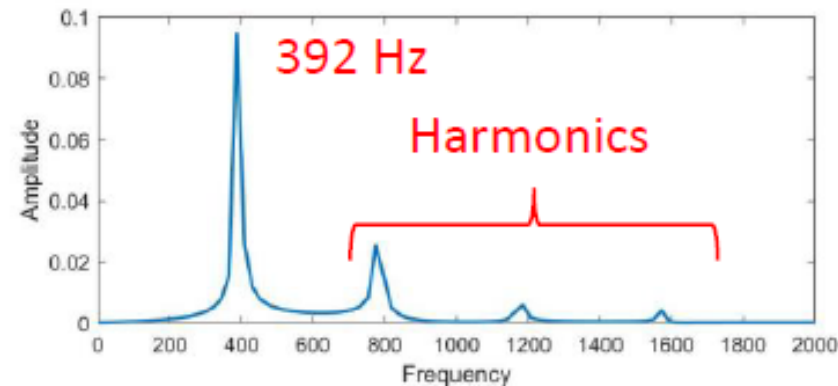


Short Time Fourier transform of “Twinkle twinkle little star”

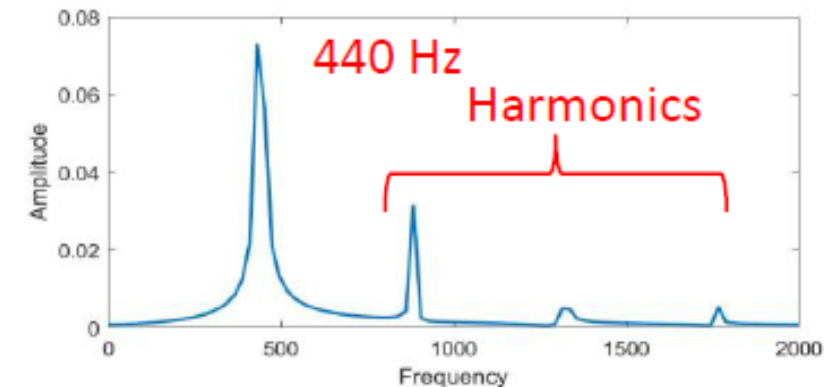
- At time window: $t=0.2554\text{s}$, STFT shows the main frequency is 262 Hz with harmonics. The key C4 is pressed at this time.
- For each time window, STFT shows the frequency and amplitude of fundamental part and harmonics.



$t_1=0.2554\text{s}$



$t_2=1.2539\text{s}$



$t_3=2.2223\text{s}$

Other mechanistic models

- How to regress the amplitude, damping, frequencies, and phase angles of the A4 key?
- Two-time scale model for the mass spring damper system (based on perturbation theory):

How machines extract/identify features from images?



Chicken wings

Hamburger

Convolution
plays a key role!

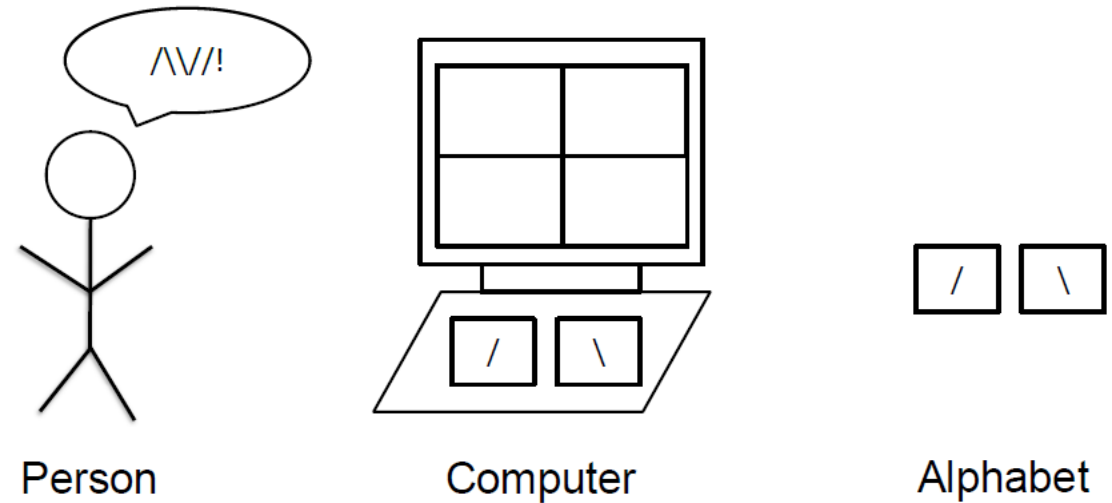
Next is to show a
simplified example.

Rainbow

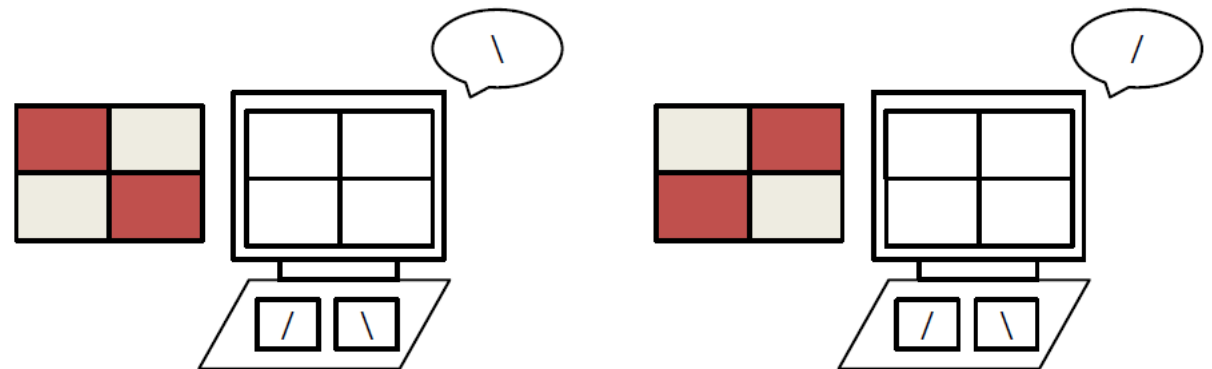


What is convolution?

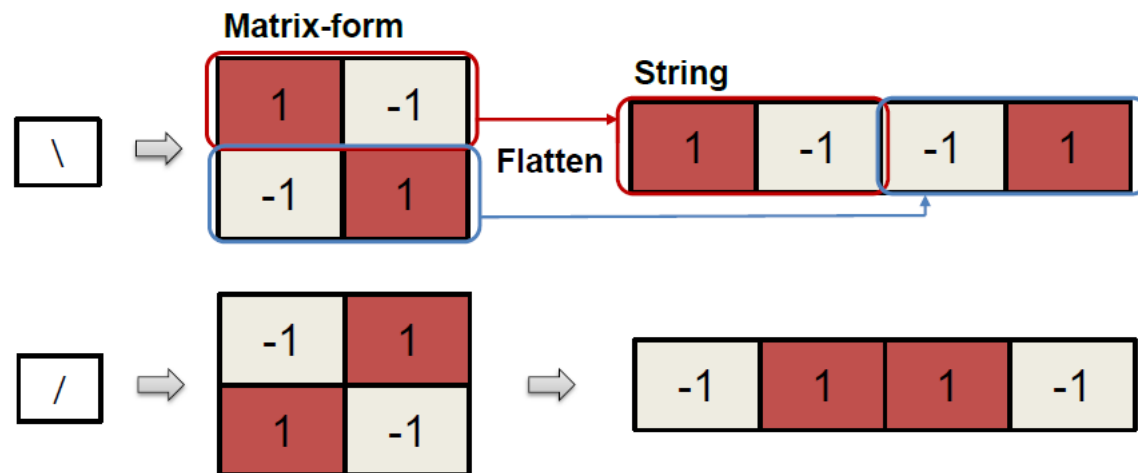
- Image a simple word:



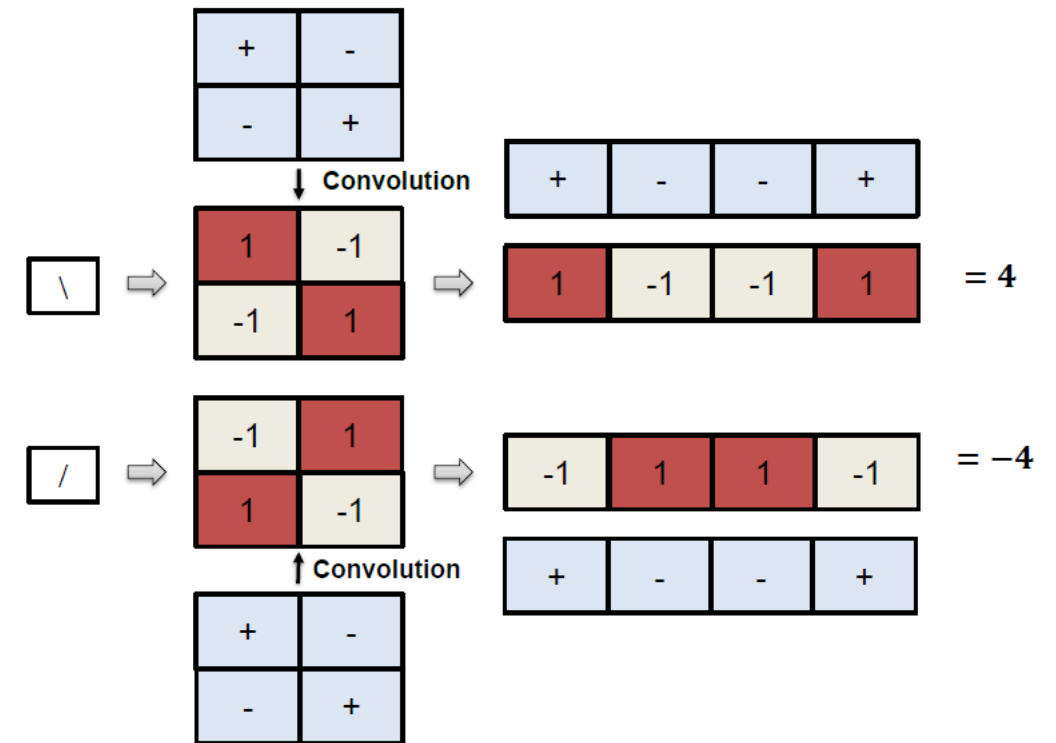
- Train computer the alphabet:



- How the data store in the computer

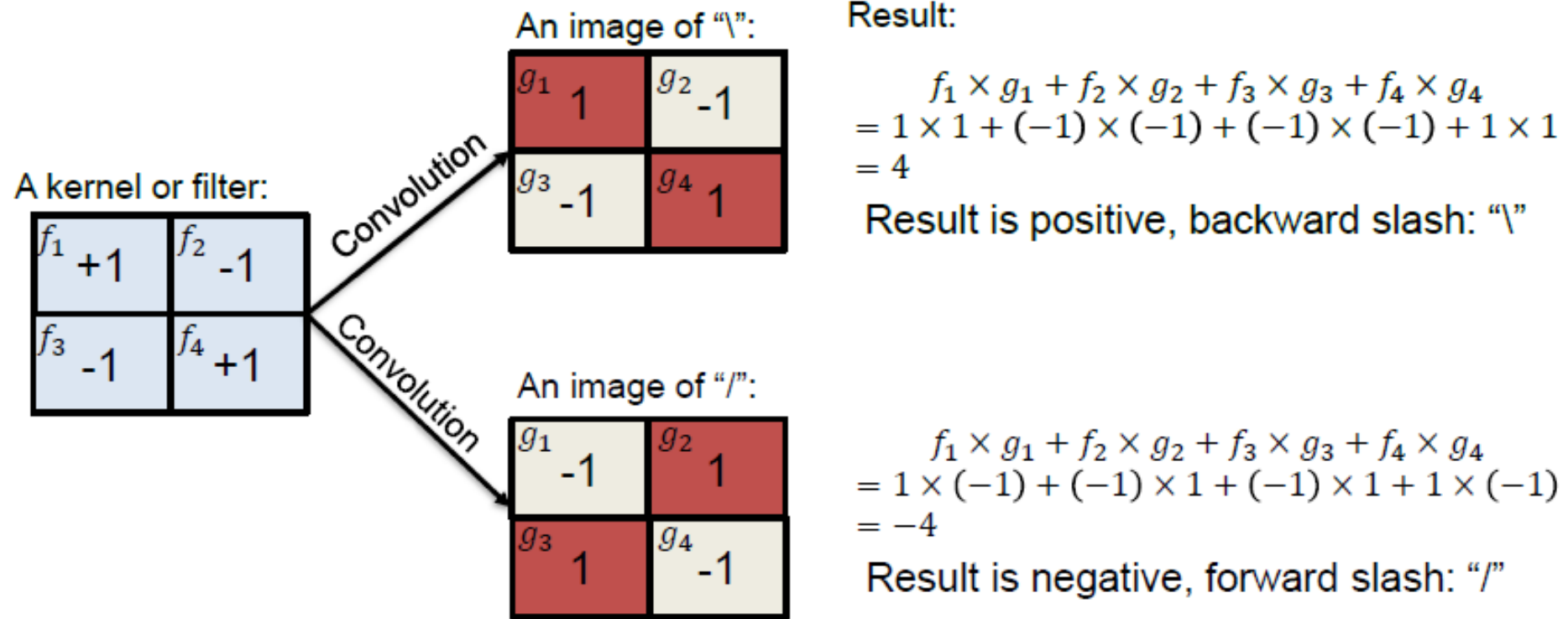


- How can we define a right operation?



An image recognition classifier

- Convolution is a formal mathematical operation, just as multiplication and addition/integration.
- Convolution is a way to pass a filter over the image to find key features.



A more rigorous definition of convolution

$$(f * \phi)(t) = \int_{-\infty}^{\infty} f(t - \xi) \phi(\xi) d\xi$$

limiting the integration interval from $-l$ to l

$$(f * \phi)(t) = \int_{-l}^l f(t - \xi) \phi(\xi) d\xi$$

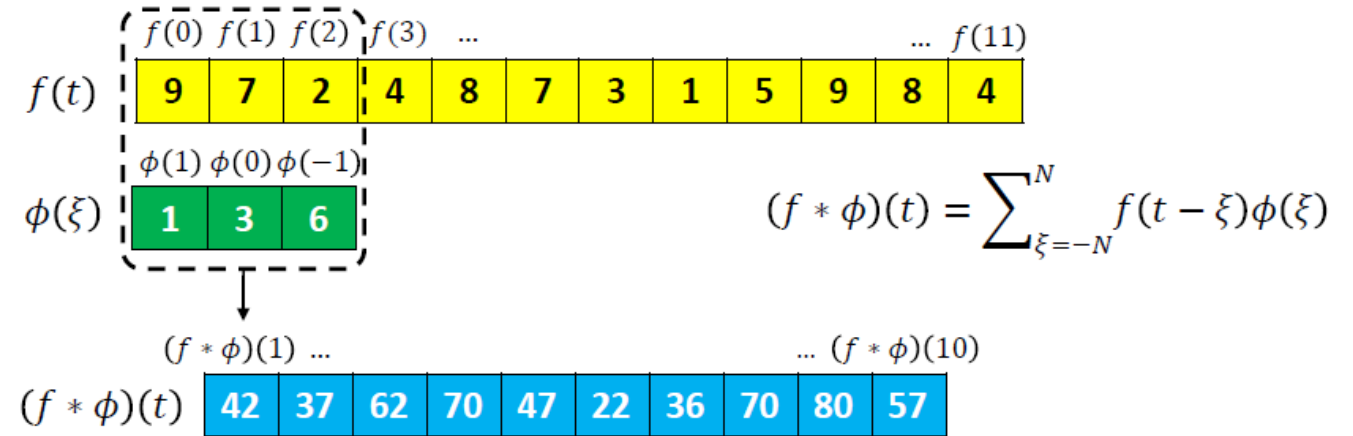
discrete convolution

$$(f * \phi)(t) = \sum_{\xi=-N}^N f(t - \xi) \phi(\xi) \Delta\xi$$

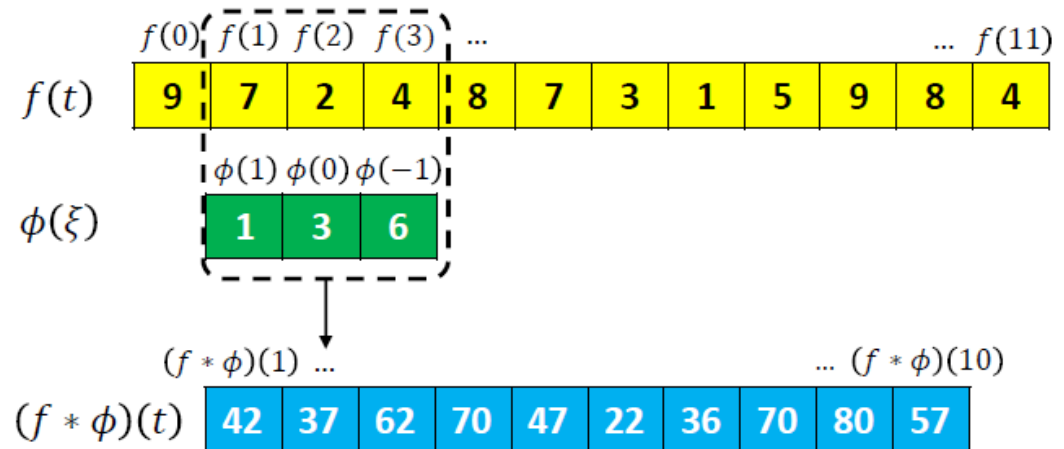
$\Delta\xi$ is assumed to be 1

$$(f * \phi)(t) = \sum_{\xi=-N}^N f(t - \xi) \phi(\xi)$$

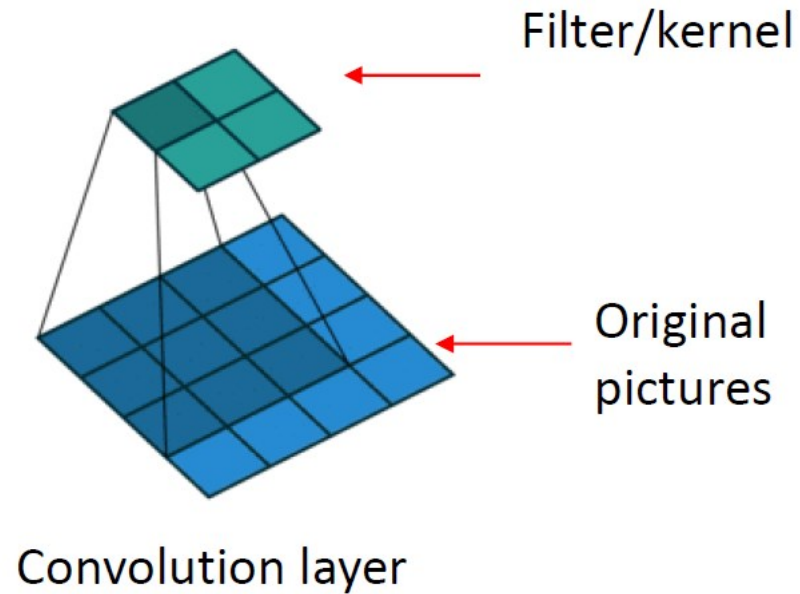
- Convolution operation step 1:



- Convolution operation step 2:



2D definition of convolution



1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Input

1	0	1
0	1	0
1	0	1

Filter / Kernel

1x1	1x0	1x1	0	0
0x0	1x1	1x0	1	0
0x1	0x0	1x1	1	1
0	0	1	1	0
0	1	1	0	0

Input x Filter

4		

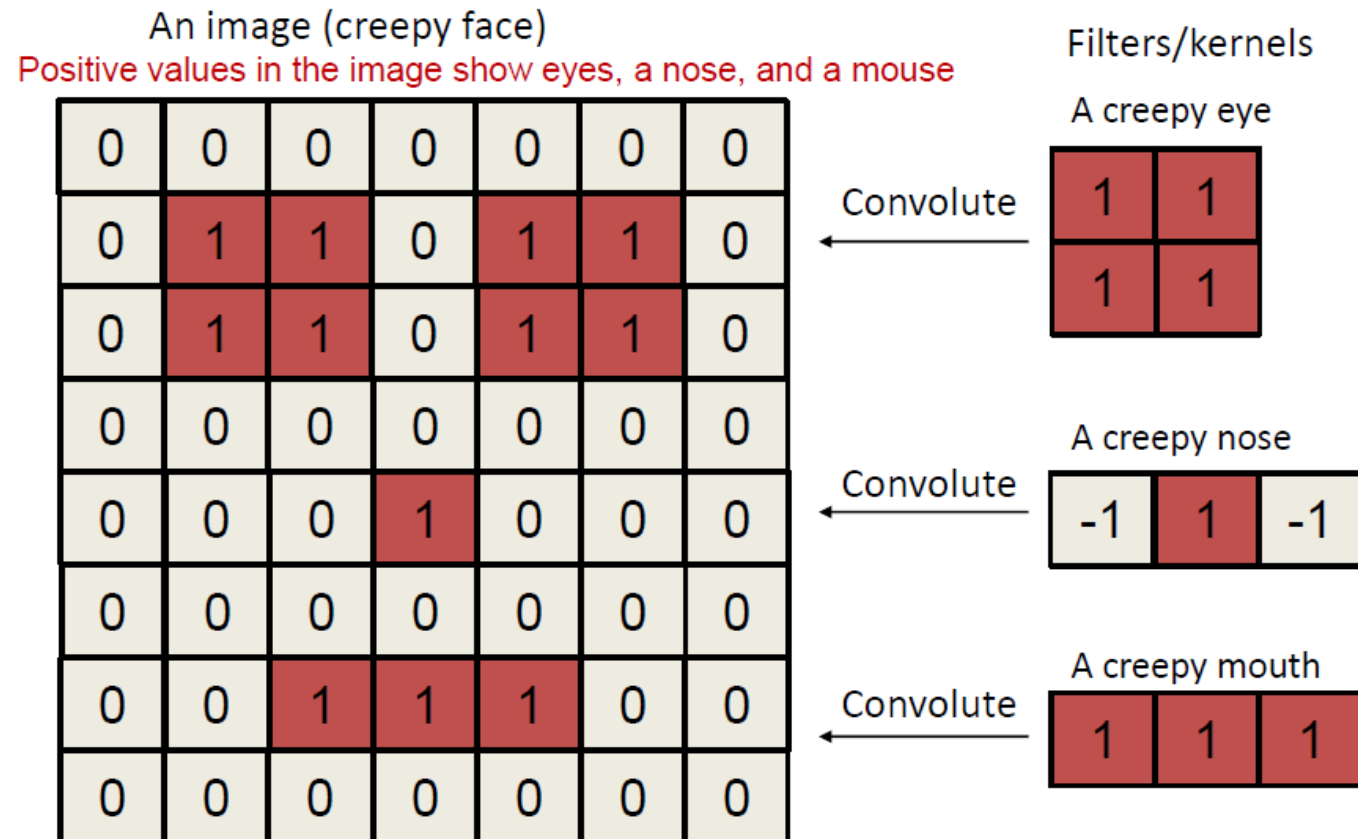
Feature Map

1x1	1x0	1x1	0	0
0x0	1x1	1x0	1	0
0x1	0x0	1x1	1	1
0	0	1	1	0
0	1	1	0	0

4		

Why can convolution extract features?

- We design three filters (called creepy eye, creepy nose, and creepy mouth) to identify corresponding features in the creepy face.



Convolution using creepy nose filter

- Only the “nose” is shown in the feature map after convolution with “a creepy nose” filter.

An image with padding
(creepy face)

0	0	0	0	0	0	0
0	1	1	0	1	1	0
0	1	1	0	1	1	0
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	0	0	0	0	0	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

A creepy nose

-1	1	-1
----	---	----

Convolution



We are looking for the **maximum values** (1 in this case) in the feature map, where shows the location of the creepy nose.

Feature map

0	0	0	0	0	0	0
0	0	0	-2	0	0	0
0	0	0	-2	0	0	0
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	0	0	0	0	0	0
0	0	0	-1	0	0	0
0	0	0	0	0	0	0

Convolution using creepy eye filter

- Only the “eyes” are shown in the feature map after convolution with “a creepy eye” filter.

An image with padding
(creepy face)

0	0	0	0	0	0	0
0	1	1	0	1	1	0
0	1	1	0	1	1	0
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	0	0	0	0	0	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

A creepy eye

1	1
1	1

We are looking for the **maximum values** (2 in this case) in the feature map, where shows the location of the creepy eyes.

Convolution
→

Feature map

1	2	1	1	2	1
2	4	2	2	4	2
1	2	1	1	2	1
0	0	1	1	0	0
0	0	1	1	0	0
0	1	2	2	1	0
0	1	2	2	1	0

Convolution using creepy mouth filter

- Only the “mouth” is shown in the feature map after convolution with “a creepy mouth” filter.

An image with padding
(creepy face)

0	0	0	0	0	0	0
0	1	1	0	1	1	0
0	1	1	0	1	1	0
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	0	0	0	0	0	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

A creepy mouth

1	1	1
---	---	---

Convolution



We are looking for the **maximum values** (3 in this case) in the feature map, where shows the location of the creepy mouth.

Feature map

0	0	0	0	0	0	0
1	2	2	2	2	2	1
1	2	2	2	2	2	1
0	0	0	0	0	0	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0
0	1	2	3	2	1	0
0	0	0	0	0	0	0