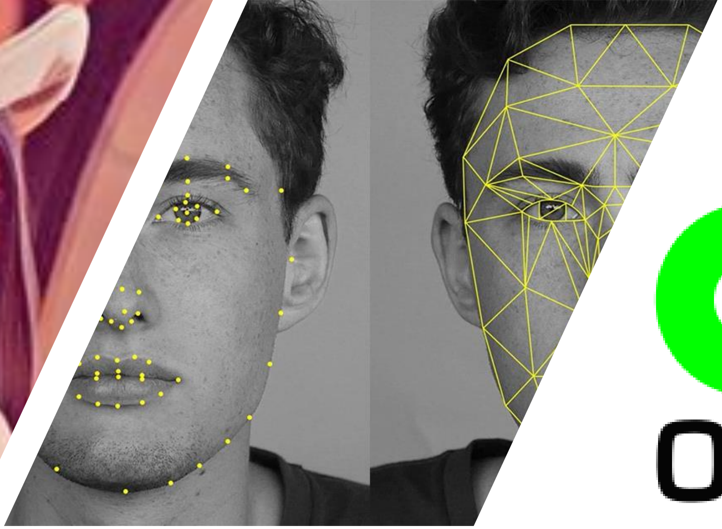
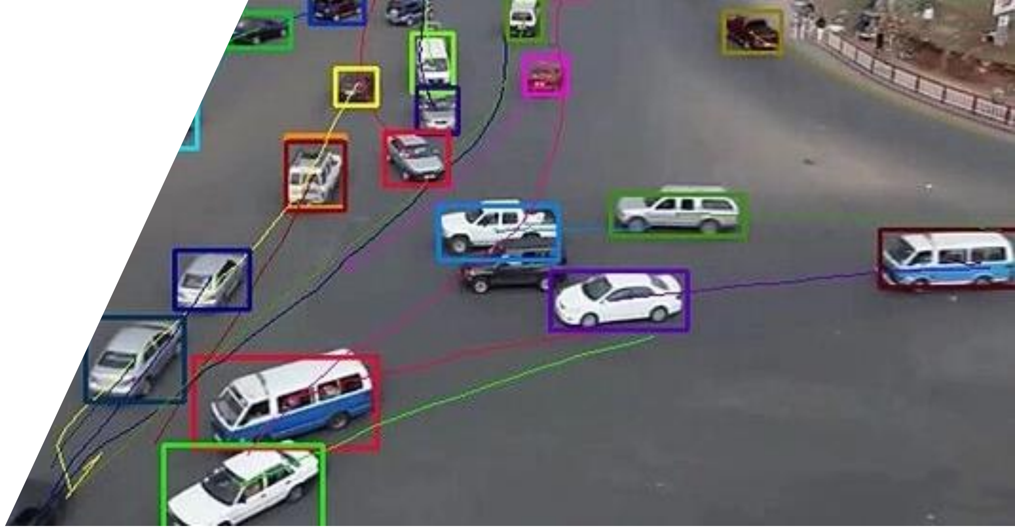
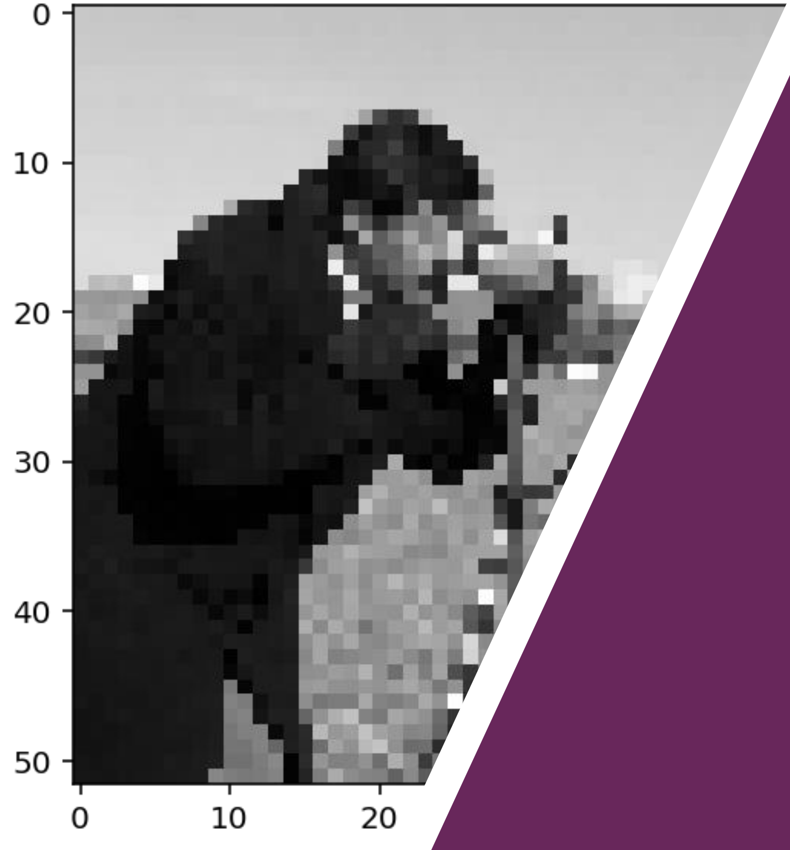


OpenCV





ECE 1390/2390

Image Processing and Computer Vision

About this course

Course number 1390 (40 students)

Course number 2390 [Graduate level] (20 students)

- graduate level will have harder problem sets and more involved project goals

About this course

Learning Objectives:

- Introduction to image processing methods including data structures, filtering, enhancement, transforms, and homography.
- Introduction to computer vision through OpenCV and Python including object detection and tracking methods, segmentation, digital photography, and in-painting.
- Hands on learning through group project. How to use GitHub for collaborations, push requests, peer-review, and code checks

Who am I?

Instructor: Ted Huppert, PhD;

Associate Professor (ECE)

Dept of Radiology (2007-2019)

Dept of ECE (2019-current)

Email: huppert1@pitt.edu

Office: BEND 1238L (in 12th floor ECE admin area)

Office hours: M/W 3:30-4:30

https://calendly.com/huppert1

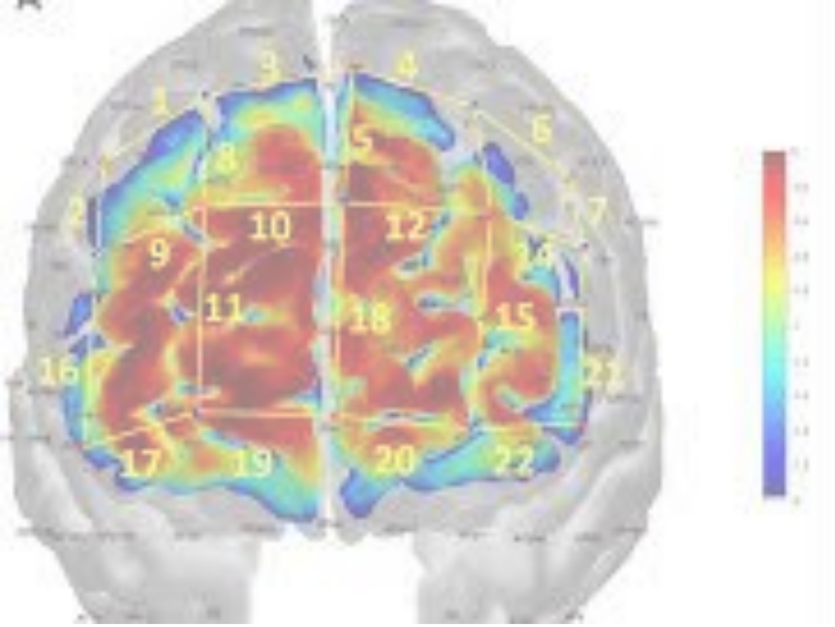
Who am I?

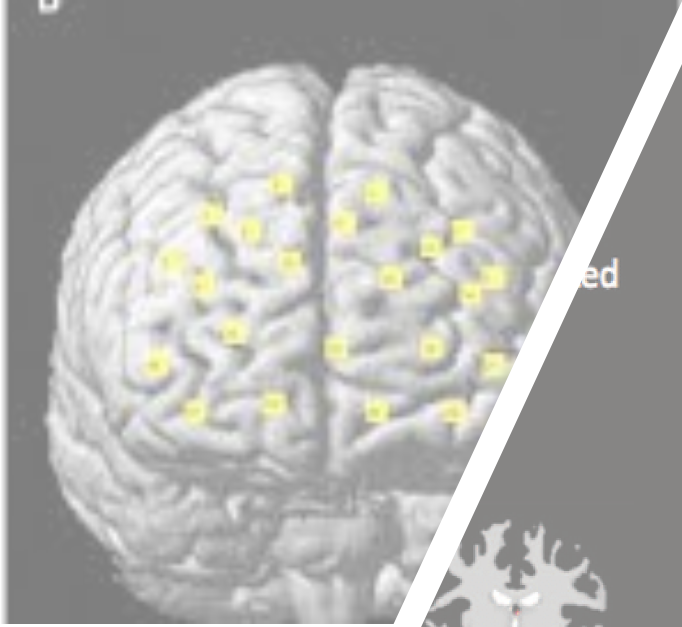
Teaching Instructor:

Yang Du

yang.du@pitt.edu

Office hours by request.





M^{3}(NI)^{2} Lab

Multimodal Methods in Noninvasive Neuroimaging Lab

www.huppertlab.net

Organization of Course

- Monday & Wednesday 4:30 – 5:45pm
- 1211 BEND

- **Lectures** (PowerPoint & Python Notebooks)
  - Materials will be provided on Canvas
  - No required textbook for course
- **Classwork** (60% of grade)
  - Practical examples done in Python/OpenCV
- **Semester project** (30% final; 10% peer-ratings).
- **No midterm or Final exams**

Grading Scale:

Homework	60% [12 x 10 points each]
Semester Project Peer scores	10% [20 pts]
Final Semester Project	30% [60 pts]

	Total 200 points

[98 - 100) A+	[88 - 90) B+	[78 - 80) C+	[68 - 70) D+
[92 - 98) A	[82 - 88) B	[72 - 78) C	[62 - 68) D
[90 - 92) A-	[80 - 82) B-	[70 - 72) C-	[60 - 62) D-

<60: F

Syllabus

Week	Date	Topic	In class work	Due date
1	8/25/2025	Introduction to course	Intro to OpenCV (HW0)	Due 9/1
	8/27/2025	Image operations	Planning projects	
2	9/1/2025	Labor Day [No Class]		
	9/3/2025	Analysis and Color spaces	Initial project proposal	
3	9/8/2025	Image Transforms	Python problems (HW1)	Due 9/15
	9/10/2025	Spatial & Frequency Filtering	Group Project work	White paper on project
4	9/15/2025	Image Restoration/Inverse filters	Python problems (HW2)	Due 9/22
	9/17/2025	Edge Detection	Group Project work	
5	9/22/2025	GMM Segmentation	Python problems (HW3)	Due 9/29
	9/24/2025	Morphometric Segmentation	Python problems (HW4)	Due 10/1
6	9/29/2025	Image Feature Detection	Group Project work	
	10/1/2025	Registration & Homography	Python problems (HW5)	Due 10/8
7	10/6/2025	Computational Photography	Python problems (HW6)	Due 10/15
	10/8/2025	Inpainting	Mid project assessment	Group ratings

Syllabus

Week	Date	Topic	In class work	Due date
8	10/13/2024	Camera Calibration	Python problems (HW7)	Due 10/20
	10/15/2024	Image Compression	Group Project work	
9	10/20/2024	Depth & 3D Reconstruction	Python problems (HW8)	Due 10/27
	10/22/2024	Haar Cascade Classifiers	Python problems (HW9)	Due 11/3
10	10/27/2024			
	10/29/2024	Face Detection	Project updates	
11	11/3/2024	Harris Corners and Blobs	Group Project work	
	11/5/2024	HOG and Custom Detectors	Python problems (HW10)	Due 11/10
12	11/10/2024	Object Tracking	Group Project work	
	11/12/2024	OCR Text Detection	Python problems (HW11)	Due 11/19
13	11/17/2024	Final Project Presentations		
	11/19/2024	Final Project Presentations		
14	11/24/2024	Thanksgiving recess		
	11/26/2024	Thanksgiving recess		
15	12/1/2024	OpenCV DNN	Python problems (HW12)	Due 12/3
	12/3/2024	Super Resolution	Python problems	

Semester Project

Design a program to implement image processing.

- Python function library of methods
- Can be GUI or command line
- Groups of 4 people (1390/2390 separated)

Examples:

- Virtual web camera
- Face-swap
- Object detection
- Text reader
- “Instagram” filter
- 3D reconstruction
- Automatic face blurring software

Class Attendance Taker:

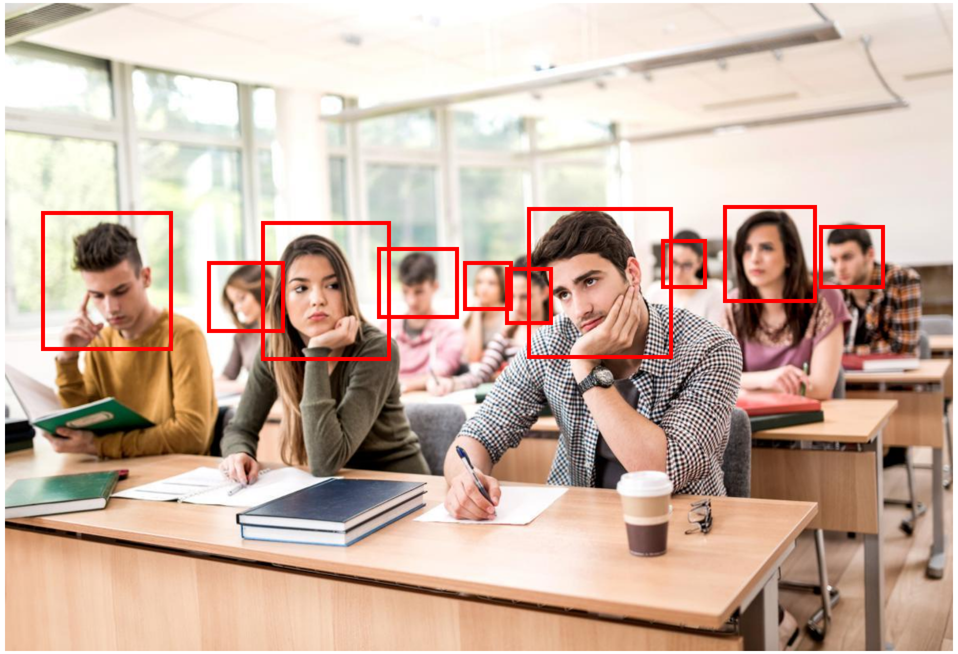
Objective: From a photo of a class, find and identify the faces in the image.

Project elements:

- Using custom Haar classifier from images of “students”, train model to identify students. Demo case, just using the four members of the group.
- Using Media Pipe, find all possible faces in the class image.
- Cut out ROI around each face
- Identify the person from feeding in the ROI to the Haar classifier

Testing case:

- Train model on team members
- Take set of photos with some/all of the team members
- Show that program can ID the team members



Soduku Puzzle Solver:

Objective: Take a picture of a Soduku puzzle. Read the number values. Program solves the puzzle (used 3rd party code). Display the solutions super-imposed on the original image.

Project elements:

- Find edges and lines of the puzzle grid to define ROI boxes.
- Use HOG classifier to ID numbers
- Solve puzzle
- Create image with solution in the right boxes

Testing case:

- Printed out puzzles and took pictures to feed into program

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

Lifting Form Feedback (2390):

Objective: From a video of a deadlift, calculate the position of the body parts and provide feedback on the correct form.

Project elements:

- From a side view camera, find the location of the dumbbell (used Hough circles and segmentation)
- Using OpenPose program (3rd part open source python code from CMU), find the location of the body parts and angles.

Testing case:

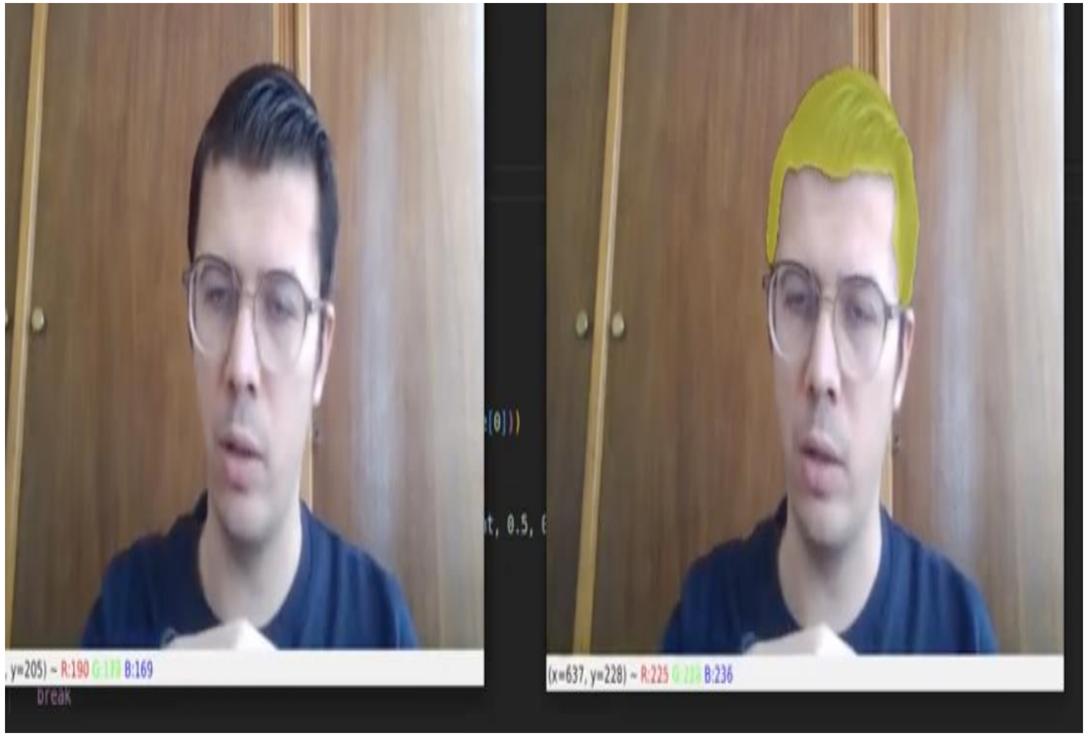
- Took sample videos of a team member lifting weights with various technique.

Virtual Hair Cut :

Objective: From a live web camera feed, find an image of a head, segment the hair and crop. Then replace the hair from one of several pre-defined styles

Project elements:
- Video feed of single person “selfie” style image
- Using media pipe for head and hair segmentation
- Overlay predefined hair templates

Testing case:
- Samples of various still and videos of faces.



Requirements:

Everyone

- Written in Python using OpenCV
- Code compliant with MIT use license
- Documented and maintained on GitHub
- Include demonstration/example code

ECE 1390 Students

- Process still images
- At least two methods described in class
- Final presentation can be via PowerPoint showing features of program

ECE 2390 Students

- Process video feeds
- At least four methods described in class
- Final presentation should include a live demo
- Best practices for collaborative code via GitHub

Project “Investor” Pitch (Due 9/3/2025)

- Each group will have 2 min to pitch their idea. (“elevator pitch”)
- 1 PowerPoint slide may be used (sent to Dr Huppert prior to class)
- Class will have 5min to ask questions, make suggestions, get clarifications, etc. (think focus group feedback)

Project White page (Due 9/10/2024)

One-page white paper describing purpose and specs of your project.

- Description.
- Code Specifications.
- Planned approach
- Time-line
- Metrics of success.
- Pitfalls and alternative solutions

Yes, you can borrow code and methods.

- With OpenCV and Python, you can find examples of virtually any code/project on the web.
- However, I grading on whether you demonstrate that you understood it.

- Requirements:
  - If you borrow code or use some tutorial to learn how to do the problem, then CITE IT!
  - Make the code your own. If you borrow code, then change variable names to be consistent with the rest of your code/library
  - Markup your notebooks and code with proper web-links
  - Implement best practices in coding
    - Names for variables/methods should be descriptive
    - Code should succinctly and clearly explain what it does.
    - Methods should document expected inputs/outputs/dependencies
    - Yes, it's fine to comment (e.g.) "I don't know why this flag is TRUE, but it doesn't work otherwise"
    - Make sure someone else could read and understand your code
  - Use GitHub. Document changes, do pull requests, issue reports when needed.

Frequently asked questions

- Can I do a video project even though I am enrolled in 1360?
  - Yes. You have to do the project requirements for 1360, but if you want to add the additional ability to do video or object tracking, you absolutely can.
- I have a great idea of methods to add, but it wasn't mentioned. Can I implement additional methods to my project?
  - Yes..

Frequently asked questions

- What is your late work policy?
  - You can turn in homework late for half credit. To stay on schedule with the course and project, it is important to keep up with the schedule, but I know things come up. You need to turn in the work by the end of the semester.

- What if I don’t finish the semester project?
  - If you keep up with the milestones, you shouldn’t have any problem finishing the minimum requirements, but you might not finish everything you planned to and wrote about in your white paper proposal of the project. In that case, I expect to see this documented on the white paper (which is part of the GitHub project).
  - I would rather you try too much and have to scale back, then to do the bare minimum.

Frequently asked questions

- Can I work on a language other than Python?
  - No. I really expect this to be done in Python. Matlab's version of OpenCV is very limited. If you have trouble with Python, there are many online resources, and we can help you in office hours.

- What if I don't know how to use GitHub?

  \- Part of developing software for a company is making sure that the code is documented and tracked for the next person to use. These are skills that I want you to learn (or improve) in this course.

Side note on licenses*

* It is a requirement of your projects that you pay attention to these

Permissive License

- Apache license (OpenCV)
- MIT license
- BSD license

- Can distribute, modify, and distribute modified versions without royalty
- Allows proprietization (you can use it as part of a more restrictive license)
- Requires NOTICE file to be included in all derivatives
- MIT requires statement of “As Is” and original copyright notice to be included in the documentation.

Copyleft

- GPL license
- AGPL license

- Can distribute, modify, and distribute modified versions without royalty
- No proprietization (derivatives must keep the same license as the original)
- Requires NOTICE file to be included in all derivatives

Non-commercial

- JRL license
- AFPL license

- Can only be used for non-commercial use

Public Domain

- PD license
- CC0 license

- No restrictions at all

Proprietary

- Can’t use!

Questions?

Types of Images

- Binary
  - 0 & 1
  - Used for masks
  - 1-bit per pixel

- Black & White image
  - 0 & 1
  - At least 8bit per pixel
  - Display provides smoothness

Although Binary and B&W both encode the same information (only 1 & 0's), the bit-depth changes how they are handled by display functions.

Types of Images

- Greyscale
  - Pixel values across range
  - At least 8bit per pixel
  - One color channel
- Color
  - Pixel values across range
  - At least 8bit per pixel
  - multiple color channels (typically 3 or 4)

Typically, the
“alpha” channel

Glossary:

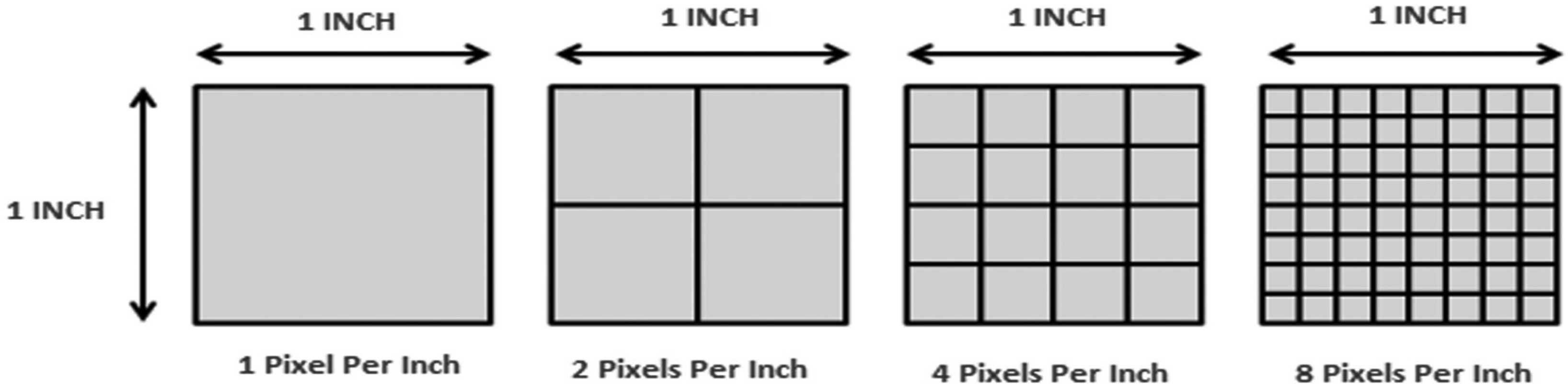
BPP: Bits per Pixel (bit depth)

Bits per Pixel (bpp)	Number of Colors
1 bpp	2 colors
2 bpp	4 colors
3 bpp	8 colors
4 bpp	16 colors
5 bpp	32 colors
6 bpp	64 colors
7 bpp	128 colors
8 bpp	256 colors
10 bpp	1024 colors
16 bpp	65,536 colors
24 bpp	16,777,216 colors (16.7 million colors)
32 bpp	4,294,967,296 colors (4294 million colors)

Bit value is an index! You always also need to define a color map (e.g. RGB, HSV, etc)

Glossary:

- Pixel Density



Humans can't differentiate details beyond around 300 PPI

- Resolution

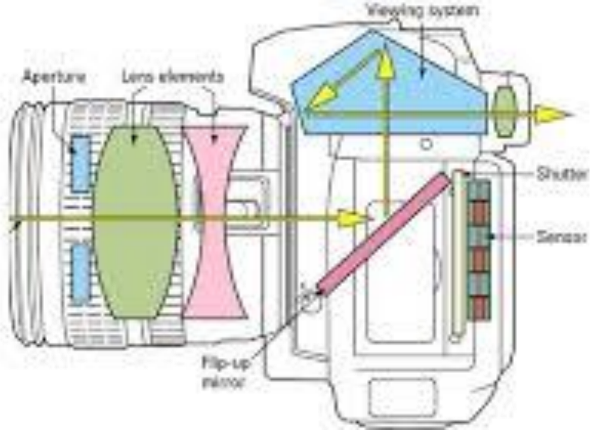
  Bit more ambiguous of a term.

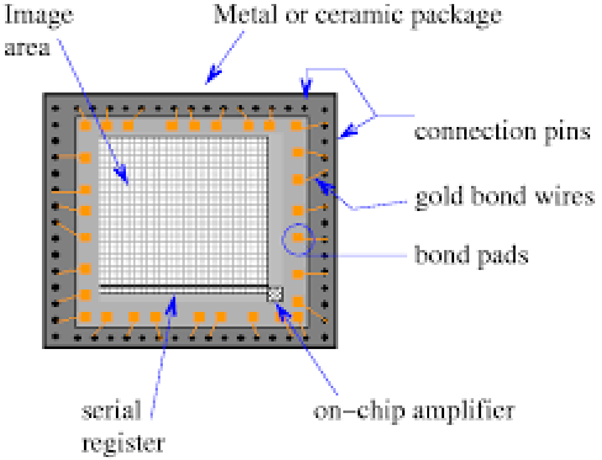
  Sometimes:

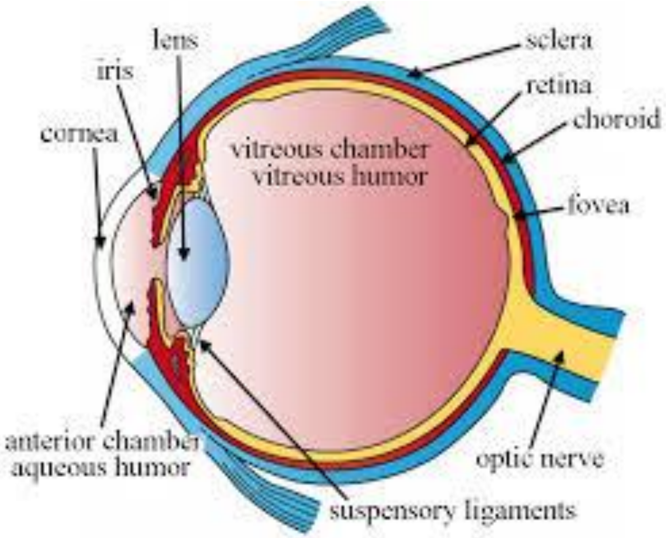
  - total number of pixels (e.g. 3Mpixels)
  - dimensions (e.g. 2556 x 1179)
  - PPI (e.g. 460ppi)
  - Pixels/degree (e.g. Apple Retina™ display)

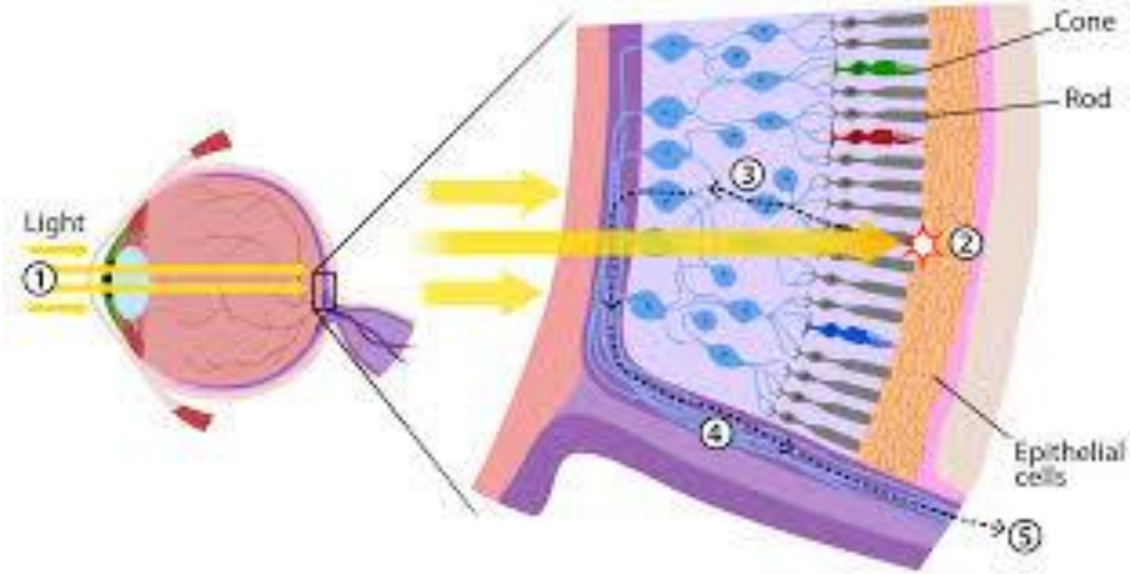
Glossary:

<u>Resolution</u>



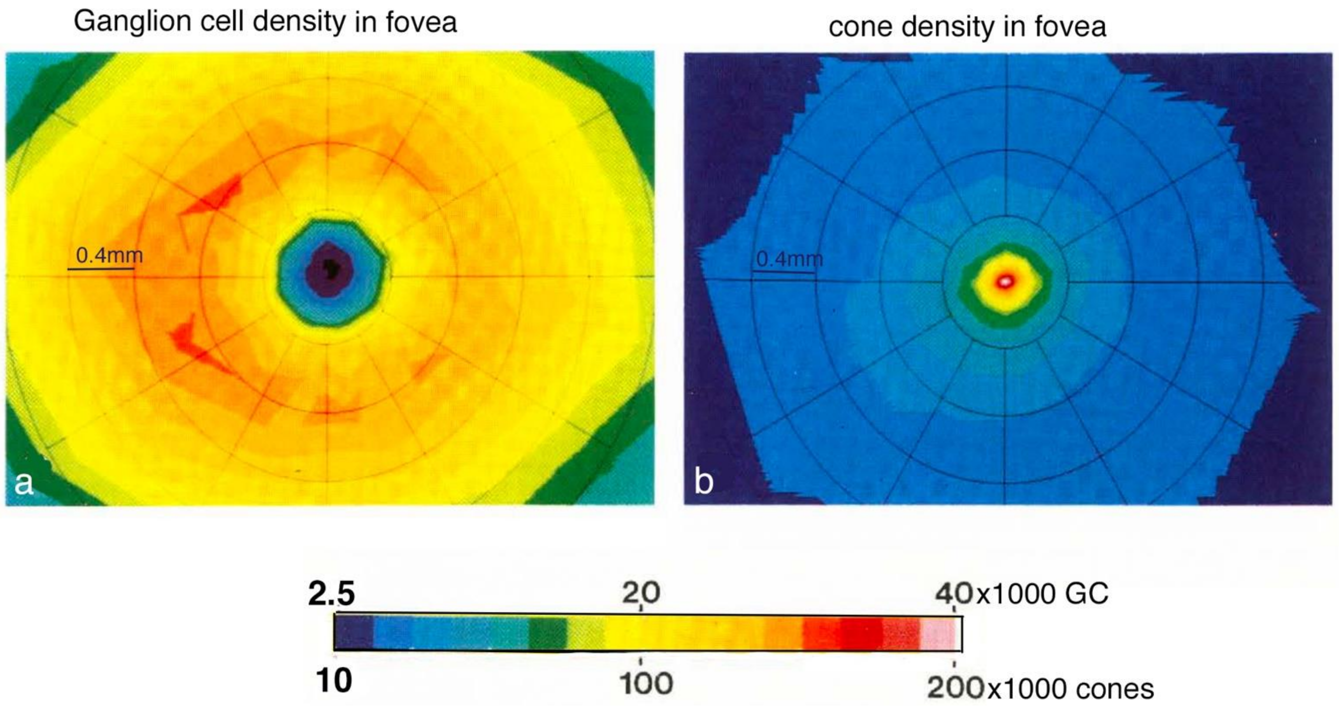






Glossary:

Resolution



From Curcio and Allen 1990

- Visual acuity is highest at the center of the eye (fovea)
- Here, the limit of the human eye is about 60 pixels / degree (1 arc-minute)

  e.g. 3600 pixels (60 x 60) in a $1^o \times 1^o$ area
- Depends on distance from object

Glossary:

Resolution

60 px/degree

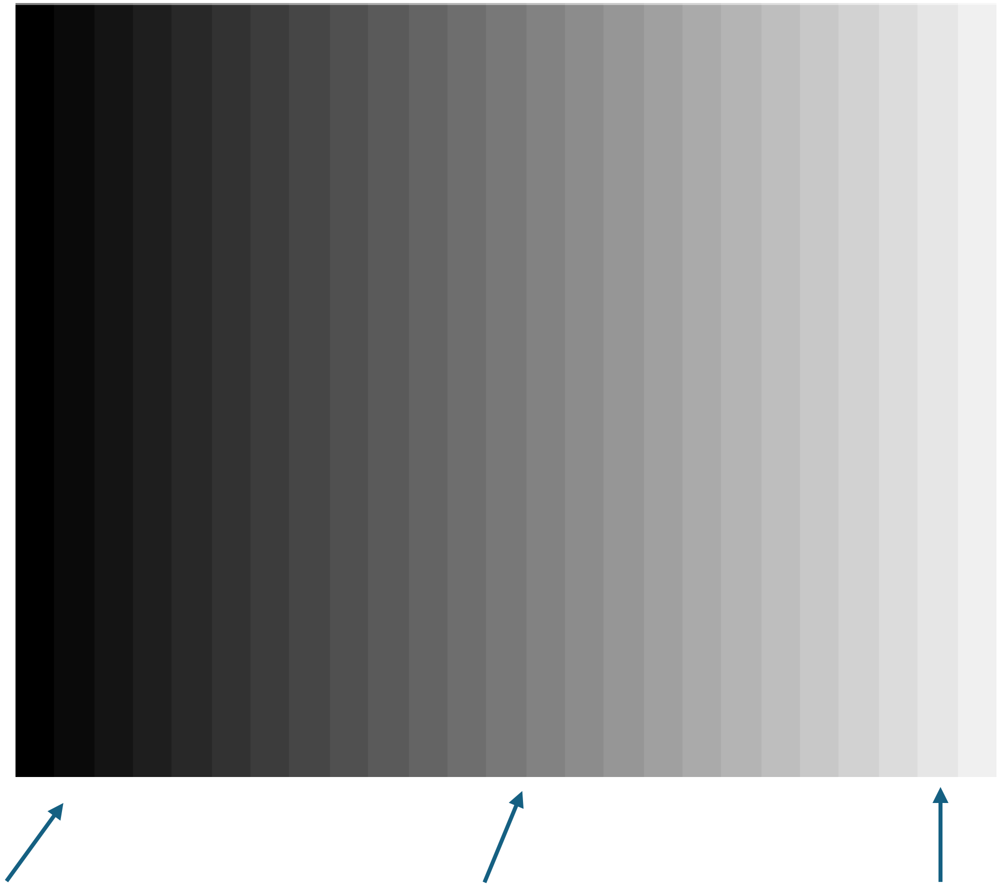
30 px/degree

15 px/degree

7.5 px/degree

Glossary:

- Intensity
  - How large is the pixel value?
- Brightness
  - How does the eye perceive intensity
- Contrast
  - Range of intensities of pixels



Gamma correction (and similar) designed to adjust intensity → brightness (next lecture)

Glossary:

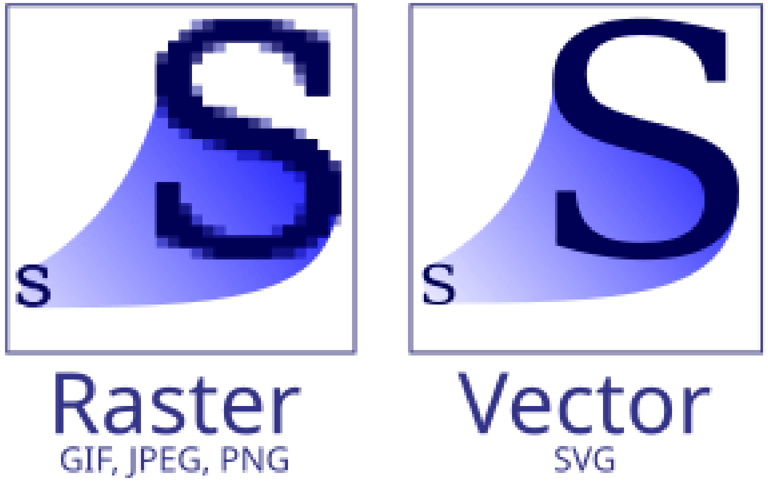
<u>Raster images</u>

-Use pixels.

-Lose information when resizing/zooming

-E.g.

JPEG, PNG, TIFF

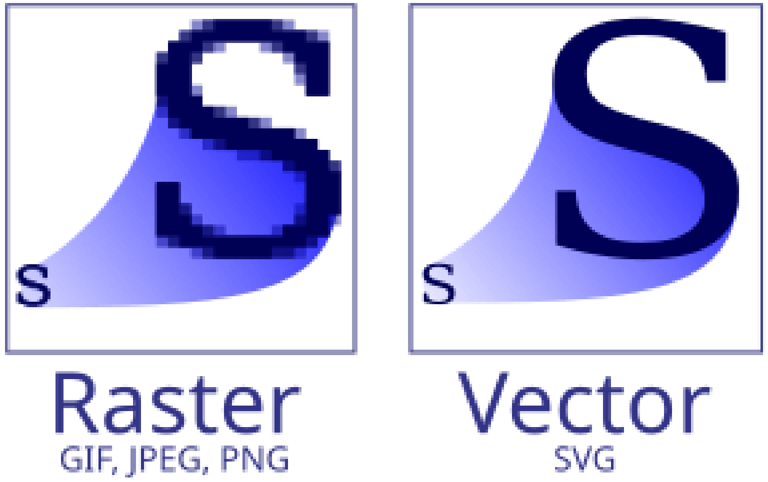


Glossary:

<u>Vector images</u>

-Define the image using piecewise formulas.
-Lossless when resizing/zooming
-Converted to raster for display

-E.g.
SVG, EPS, PDF, AI, PSD



Glossary:

<u>Tiff (Tag Image File Format)</u>

- Raster image format
- Most versatile for colors
- Supports RGB, CMY, Grayscale, and more
- Uncompressed (large file sizes, but lossless)

<u>JPEG (Joint Photographic Experts Group)</u>

- Raster image format
- Compressed <u>image</u> data (smaller size, but degradation)

<u>GIF (Graphics Interchange Format)</u>

- Raster image format
- Compressed <u>color</u> maps (smaller size, less color contrast)

OpenCV (https://opencv.org/)

- Developed by Intel in 1999
- Open-source (https://github.com/opencv)
- Apache License 2

Features:

- Image processing and visualization
- Object recognition
- Segmentation
- Facial and gesture recognition
- Image homography/registration
- Depth perception
- Deep learning methods

Code:

- Written in C++
- Bindings in Python, Java, Matlab
- CUDA support (since 2010)
- Currently version 4