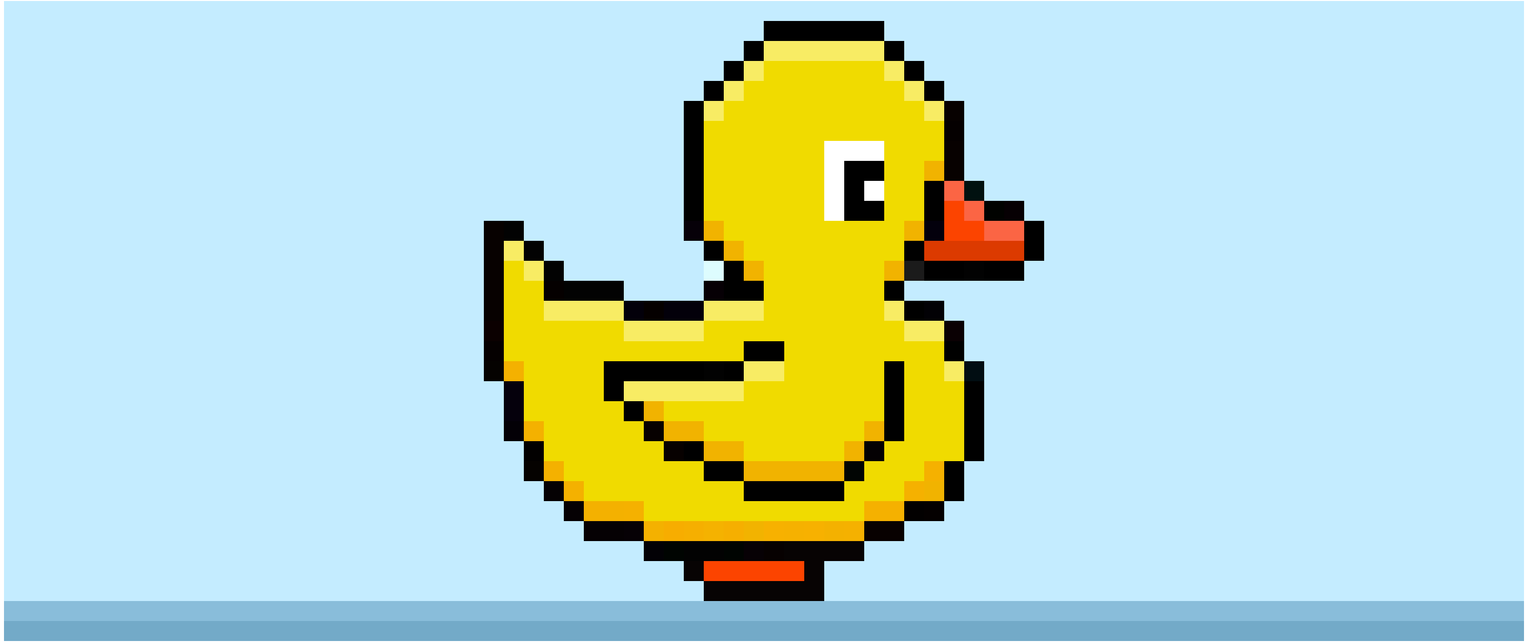
Image and Video Basics

Tushar B. Kute,
http://tusharkute.com

Pixel

- In digital imaging, a pixel (short for "picture element") is the smallest addressable element in a raster image or the smallest addressable element in a dot matrix display device.
- In simpler terms, it's the basic building block of any digital image you see on screens, smartphones, computers, TVs, etc.

Pixel



Pixel

- Here's a closer look at what makes a pixel tick:
- Color:
  - Each pixel can represent a specific color.
  - This color information is usually stored in three channels: red, green, and blue (RGB).
  - By adjusting the intensity of these channels, we can create an almost infinite range of colors and hues.

Pixel

- Brightness:
  - Pixels can also hold information about their brightness, ranging from pure black to pure white. This allows for detailed shading and depth in images.
- Resolution:
  - The number of pixels in an image is called its resolution. The higher the resolution, the more detailed and sharper the image will be.
  - For example, a high-definition image has millions of pixels, while a lower-resolution image might only have thousands.

Pixel: Why?

- Understanding pixels is crucial because they are the foundation of how computers and devices process and display visual information.
- From capturing images with cameras to manipulating them in editing software, everything revolves around manipulating and interpreting these tiny squares of color.

Pixel: Why?

- Here are some additional points to consider:
  - Size:
    - Pixels are incredibly small, often measured in micrometers.
    - Therefore, the individual pixels are usually not visible to the naked eye unless you zoom in very close.

Pixel: Why?

- File size:
  - The number of pixels in an image directly affects its file size.
  - More pixels mean more information, which translates to a larger file.
- Applications:
  - Pixels are used in various fields beyond just displaying images, such as computer vision, scientific visualization, and medical imaging.

Pixel coordinates

- This is the most fundamental system, with each pixel in the image assigned a row and column number.
- These discrete numbers determine the location of each pixel within the image grid.
  - Origin: Typically starts at the upper left corner, with the first row being 0 and the first column also being 0.
  - Direction: Rows increase downwards, while columns increase rightwards.
  - Example: Pixel at row 5, column 3.

Pixel coordinates

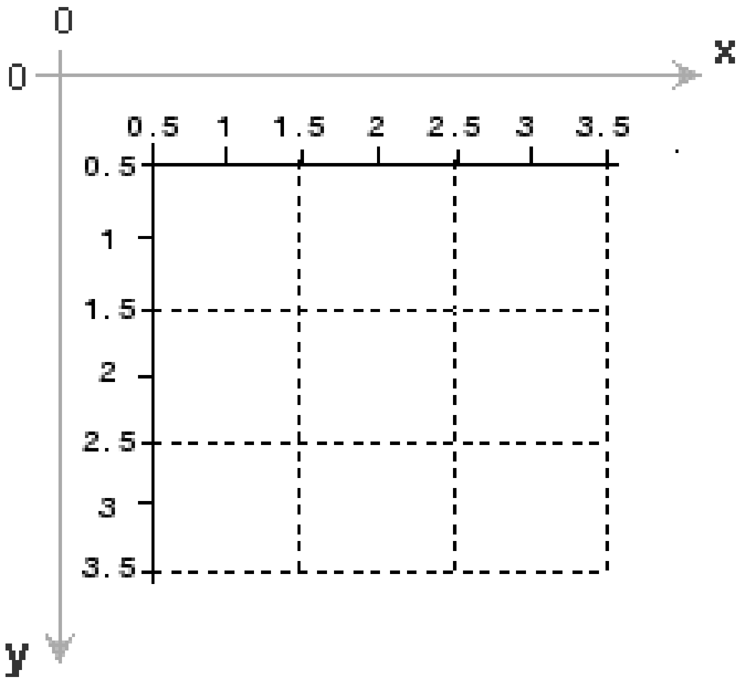


Image coordinates

- This system builds upon pixel coordinates but utilizes continuous real numbers rather than integers.
  - Origin: Can vary depending on the context, but often aligns with the pixel coordinate origin (upper left corner).
  - Units: May be expressed in pixels, millimeters, or other units depending on the desired scale and application.
  - Example: Coordinates (3.2, 5.7) representing a point between pixels (3, 5) and (4, 6).

Intrinsic coordinates

- This system defines locations within the image based on its intrinsic properties, independent of pixel or screen dimensions.
  - Origin: Often coincides with the center of the image or a specific feature within it.
  - Units: Normalized values ranging from 0 to 1 for both horizontal and vertical axes.
  - Example: Point located at (0.5, 0.8) represents the center of the image on the vertical axis and 80% along the horizontal axis.

World coordinates

- In applications like image registration or scene mapping, images might be mapped to real-world coordinates.
  - Origin: Aligns with the chosen reference point in the real world.
  - Units: Real-world units like meters, kilometers, or geographic coordinates.
  - Example: Point at (100, 200) on the image corresponds to a location with specific latitude and longitude in the real world.

Accessing and Manipulating pixels

- Programming Languages and Libraries:
  - Python: Popular libraries like OpenCV, Pillow, and Matplotlib offer convenient functions for reading and writing pixel values, iterating through pixels, and applying transformations.
  - C++: OpenCV also provides access to low-level image manipulation in C++, suitable for performance-critical applications.
  - JavaScript: Canvas API can manipulate pixels within HTML elements, enabling interactive web experiences.

Accessing and Manipulating pixels

- Accessing Pixel Values:
  - Indexing: Individual pixels can be accessed by their row and column indices, similar to accessing elements in a 2D array.
  - Slicing: Subsets of pixels can be selected using slicing notations to apply operations to specific regions.
  - Iterating: Loops can be used to iterate through all pixels in the image, enabling pixel-by-pixel modifications.

Accessing and Manipulating pixels

- Manipulating Pixel Values:
  - Changing Color: You can directly assign new RGB values to individual pixels or modify channels (red, green, blue) to adjust hue, saturation, and brightness.
  - Applying Filters: Predefined filters like blur, sharpen, or edge detection can be applied to the entire image or specific regions to alter its visual appearance.
  - Custom Transformations: Custom functions can be implemented to process pixel values and achieve specific effects, like object detection or color quantization.

Accessing and Manipulating pixels

- Important Things to Consider:
  - Image Format: Different image formats (e.g., JPEG, PNG) have different color representations and compression levels, which can affect pixel access and manipulation.
  - Data Types: Ensure you understand the data types used for storing pixel values to avoid errors and unexpected behavior.
  - Memory Efficiency: Large images can consume significant memory during manipulation. Consider optimizing algorithms and data structures for efficiency.

Accessing and Manipulating by Python

- OpenCV (cv2):
  - Highly efficient for real-time image processing, video analysis, and computer vision tasks.
- Pillow (PIL):
  - Simple and intuitive for basic image manipulation and editing.
- NumPy:
  - While not specifically for image processing, it provides powerful array manipulation capabilities that can be applied to images.

Common Manipulations

- Changing pixel values:
  - Assign new values to individual pixels or regions.
- Applying filters:
  - Use mathematical operations or convolutions to modify pixel values based on their neighbors.
- Creating masks:
  - Define regions of interest for selective processing.
- Edge detection:
  - Identify boundaries between objects.
- Object segmentation:
  - Isolate specific objects within the image.

Video

- A video, in its essence, is an electronic medium for recording, copying, playing back, broadcasting, and displaying moving visual media.
- In simpler terms, it's a series of consecutive images (frames) displayed in rapid succession, creating the illusion of movement and capturing a dynamic scene.

Video : Components

- Frames:
  - Individual still images that make up the video. Frame rate, measured in frames per second (fps), determines the smoothness and realism of the motion.
- Codec:
  - An algorithm that compresses and decompresses video data for efficient storage and transmission. Different codecs offer varying levels of compression and quality.

Video : Components

- Audio:
  - Optional but often complementary, audio adds another layer of information and immersion to the video experience.
- Metadata:
  - Additional information about the video, such as title, creation date, duration, resolution, etc.

Video : Analysis

- Object tracking:
  - Following objects' movements across frames, understanding interactions and trajectories.
- Activity recognition:
  - Classifying human actions like walking, running, jumping, or even complex interactions like sports or traffic behavior.
- Anomaly detection:
  - Identifying unusual events or deviations from expected patterns in video footage.

Video : Understanding

- Optical flow:
  - Estimating the motion of pixels between frames, revealing patterns of object movement and scene dynamics.
- 3D reconstruction:
  - Building 3D models of scenes from multiple video frames, enabling virtual world interactions and robotic perception.
- Background subtraction:
  - Identifying foreground objects (e.g., people, vehicles) moving against a static background.

Video : Applications

- Video surveillance:
  - Analyzing security footage for anomaly detection, object tracking, and intrusion detection.
- Self-driving cars:
  - Processing real-time video to navigate roads, identify obstacles, and understand traffic signals.
- Medical imaging:
  - Analyzing medical videos for disease diagnosis, surgical planning, and treatment monitoring.
- Sports analytics:
  - Extracting player movements, performance metrics, and tactics from game footage.

Video Processing

- Practical

Thank you

@mitu_skillologies

@mITuSkillologies

@mitu_group

@mitu-skillologies

@MITUSkillologies

kaggle

@mituskillologies

Web Resources
https://mitu.co.in
http://tusharkute.com

@mituskillologies

contact@mitu.co.in

tushar@tusharkute.com