

## Computer Graphics

Computer graphics is part of a larger field that is visual computing, this includes

- Computer graphics and computer vision.
- Image capture and image display.

Computer graphics includes:

- User interface
- 3D renders
- AR/VR
- Special effects, including image editing

## Image as a 2D Array

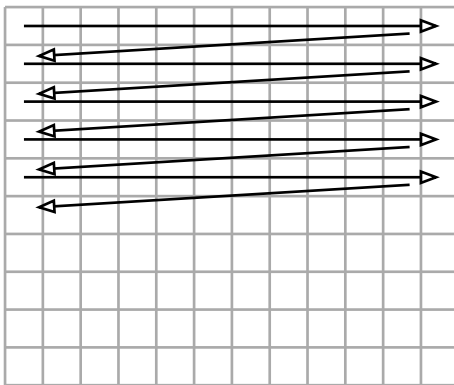
From a discrete perspective, an image is a **2D array of pixels**. The memory is not two-dimensional, how do we store it in memory?

### Row Major

The first row is stored before the second row.

The index of a pixel

$$i(x, y) = x + y \cdot n_{\text{cols}}$$

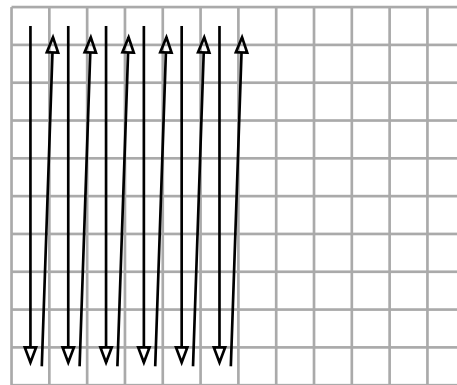


### Column Major

The first column is stored before the second column.

The index of a pixel

$$i(x, y) = x \cdot n_{\text{rows}} + y$$

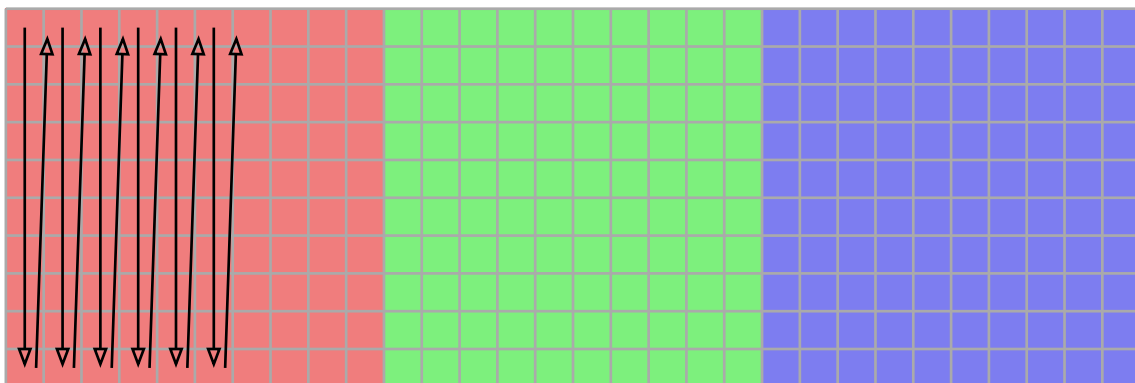


## RGB Representations

- **Interleaved, row major** stores all colours of a pixel next to each other, before moving on to the next pixel.

$$i(x, y, c) = 3x + 3y \cdot n_{\text{cols}} + c$$

- **Planar, column major** lays the R, G and B image next to each other, then use column major on the combined image.



$$i(x, y, c) = x \cdot n_{\text{rows}} + y + cs_c$$

The general formula is  $i(x, y, c) = xs_x + ys_y + cs_c$ , where  $s_x$  is the number of pixels for each step in the  $x$  direction.

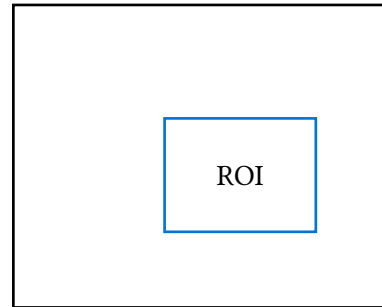
### Padded Image

An algorithm that operates on neighbouring pixels need a padded image so every pixel in the image has neighbouring pixels.

The pixels in the region of interest are given by

$$i(x, y, c) = i_{\text{first pixel}} + xs_x + ys_y + cs_c$$

Where  $i_{\text{first pixel}}$  is the index of the top left pixel of the ROI.



### Pixels

Pixel is short for **picture element**.

Each pixel consist of 3 values R, G, B describing the colour.

- 0-255 for each colour, because it is convenient to use 1 byte per colour.
- It is also the number of colours we need to have no visible artifacts.

### Colour Banding

#### Definition

**Colour banding** is when there are not enough bits to represent colour.

This is very visible to our eyes due to the mach band/chevreul illution where our eyes enhances the contrast of an edge, making the band more visible than it is.

#### Definition

**Dithering** adds noise to reduce banding: randomly add a value to the pixel value, the probability determined by the colour of the pixel.

### Image as a Continuous Function

From a continuous perspective, an image is a continuous 2D function. This allows mathematical functions to run on the image.

#### Definition

A **pixel** is a point with no dimension.

We don't have continuous memory in computers, so we need to

- **Sample** the points, and
- **Quantise** the level of allowed values.

## Rendering

### Depth Perception

Our eyes will use anything to infer the depth of a scene, including:

- Occlusion: where one thing covers another because it is in front of the other thing.
- Relative size
- Distance to the horizon: the closer to the horizon the further away it is.
- Infer the shape of an object from shading.
- Red objects look closer than blue objects: chromatic aberration focus it further back in retina.
- Atmosphere/focus
- Perspective, this is easier when there are parallel lines.

In CG we want to use all the cues we can give.

### Raytracing

We identify points on the image surface and calculate illumination for each pixel.

1. Shoot many rays from the camera.
2. Then calculate the colour of the pixel from where the ray hits, which is the closest intersection point (of the ray) to the camera.

Raytracing can easily handle reflection, refraction, shadows and motion blur, but is computationally expensive.

### Finding Intersection

$$\begin{aligned}\text{ray : } \mathbf{r} &= \mathbf{o} + s\mathbf{d} \\ \text{plane : } \mathbf{r} \cdot \mathbf{n} + d &= 0\end{aligned}$$

After solving

$$s = -\frac{d + \mathbf{n} \cdot \mathbf{o}}{\mathbf{n} \cdot \mathbf{d}}$$

To find an intersection with a polygon

1. Find intersections with the plane
2. Check whether the point is inside the polygon, which is just 2D geometry.