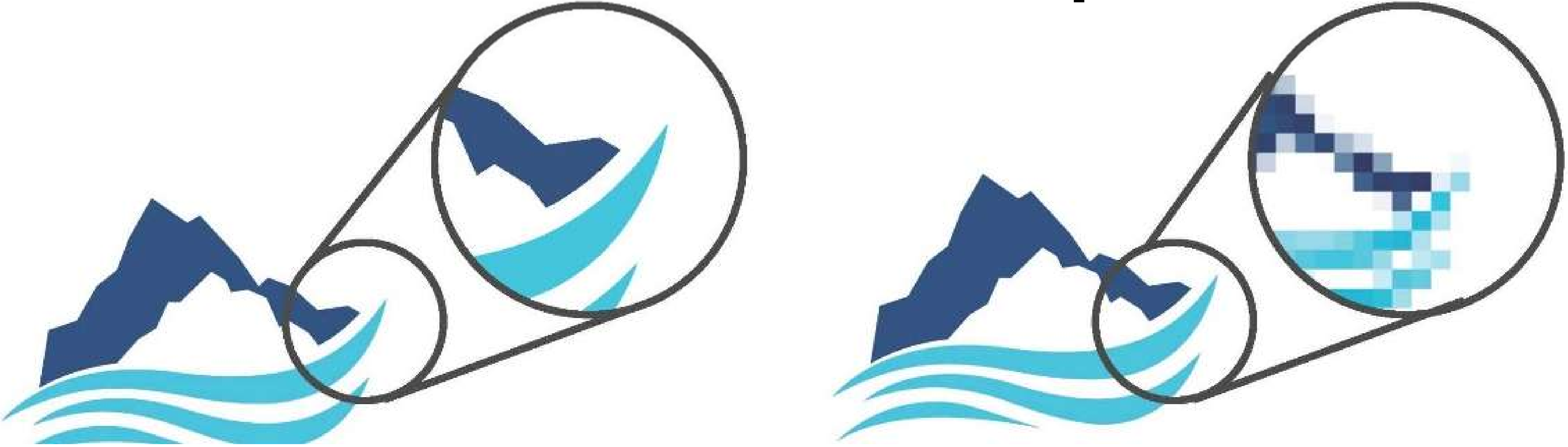
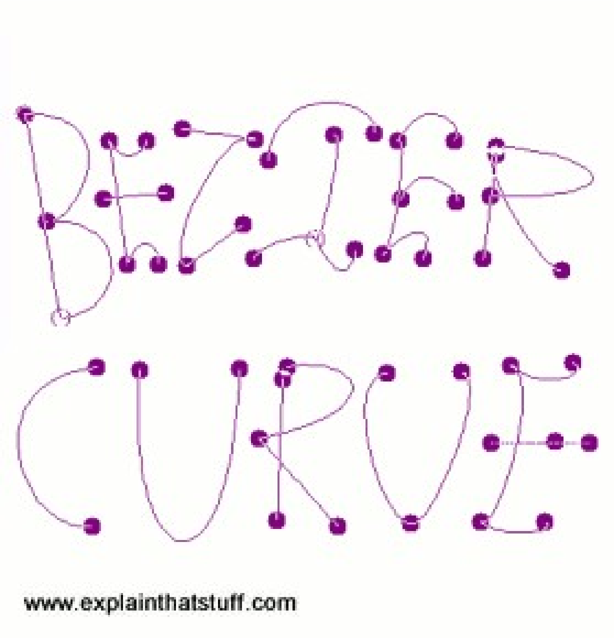
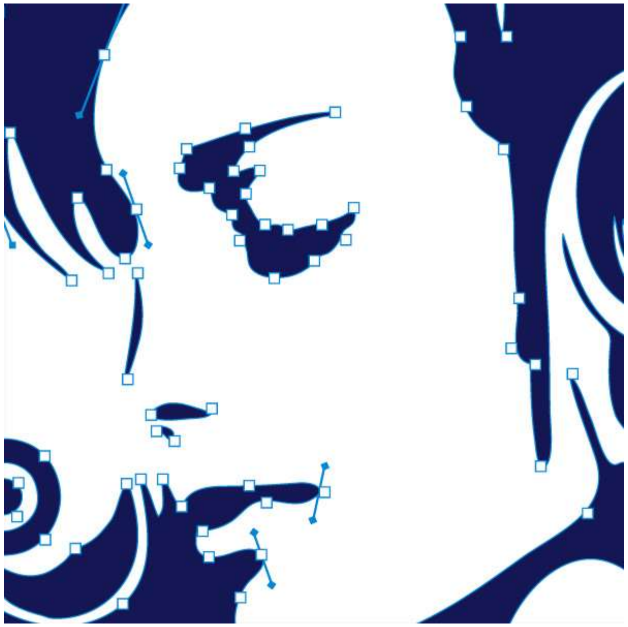
Vector vs Raster Graphics

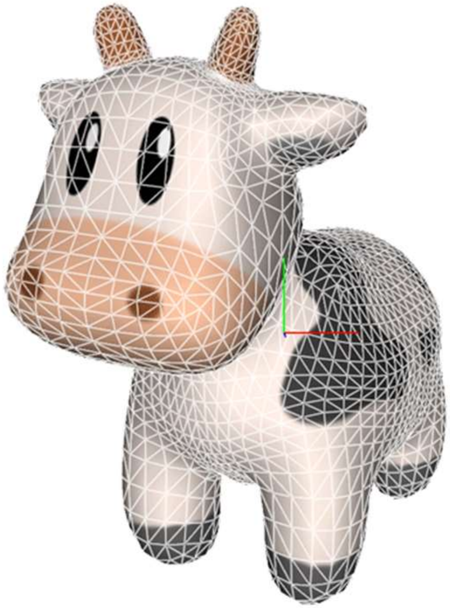


Vector Graphics

- Geometrical (mathematical) representation

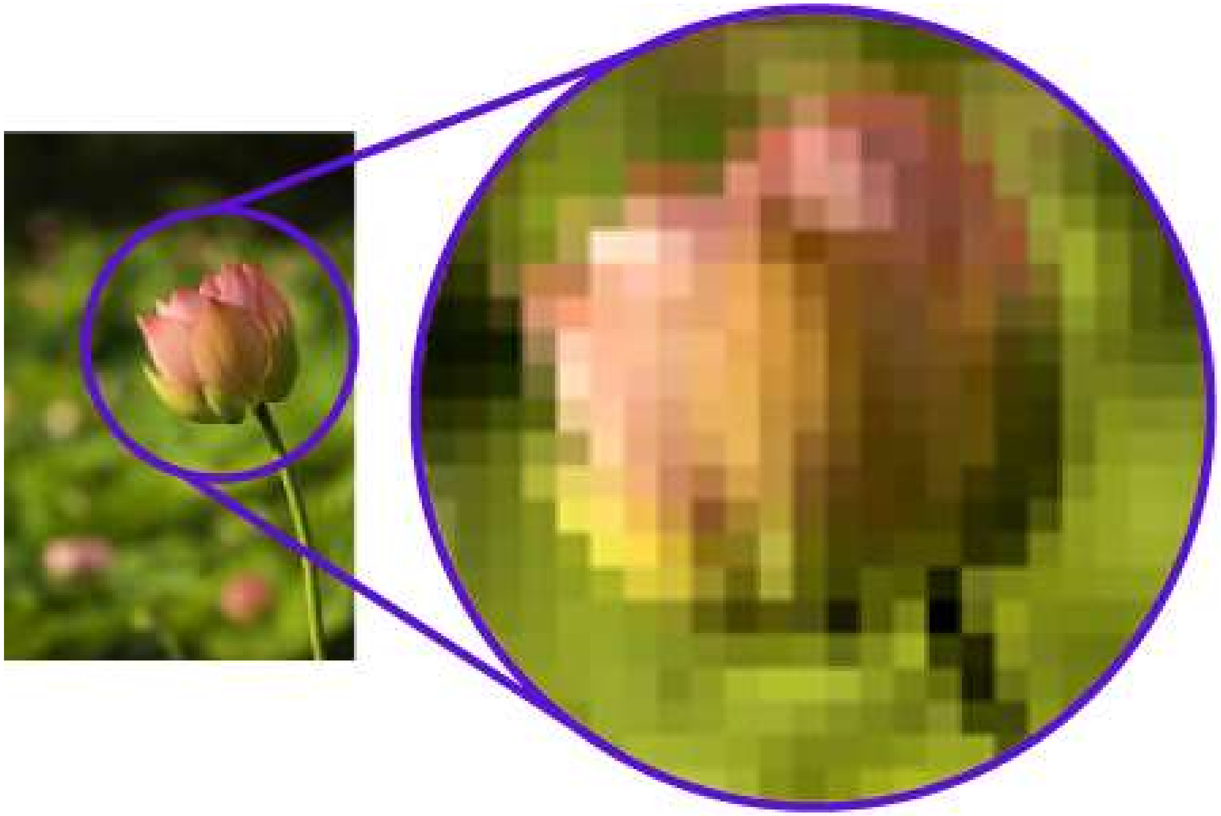




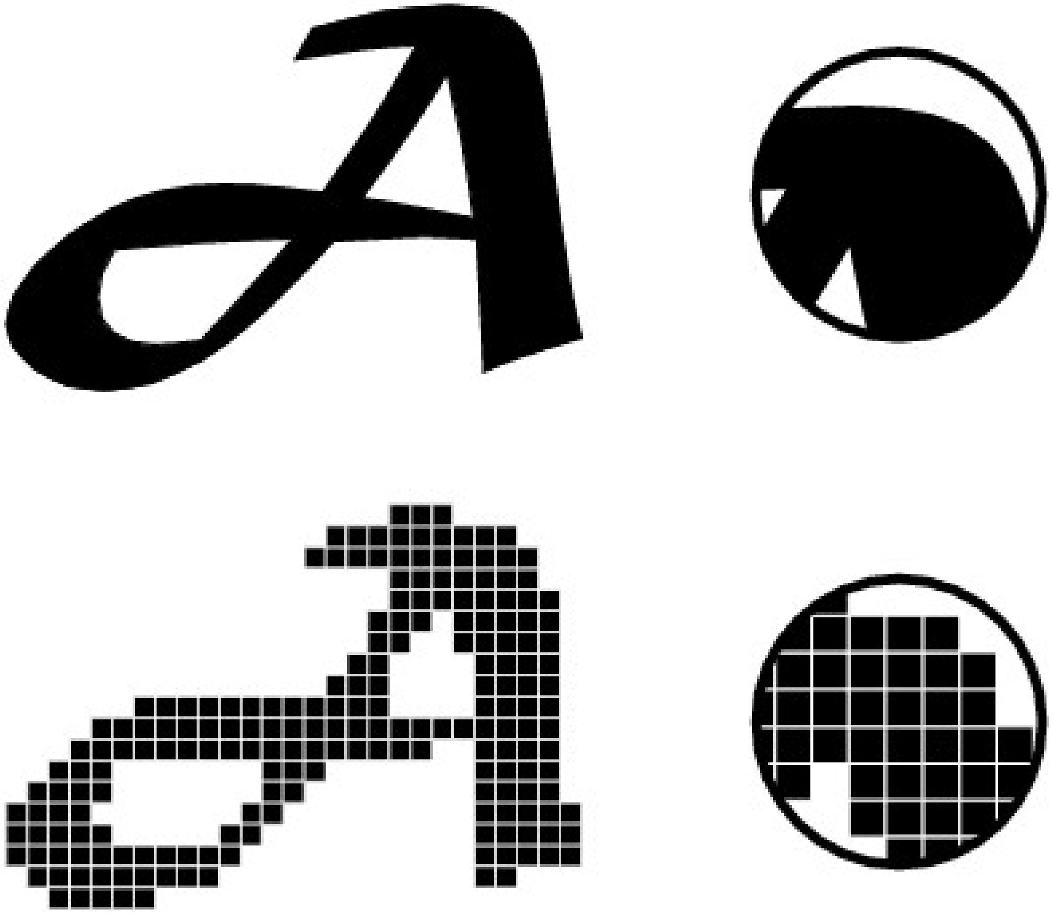


Raster Graphics

- Rectangular grid of colored elements

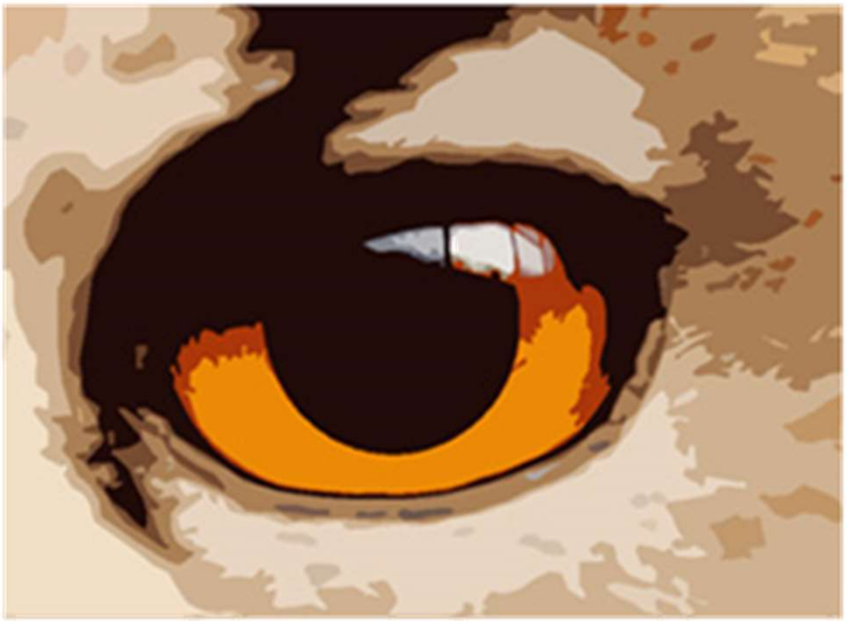


Zoom

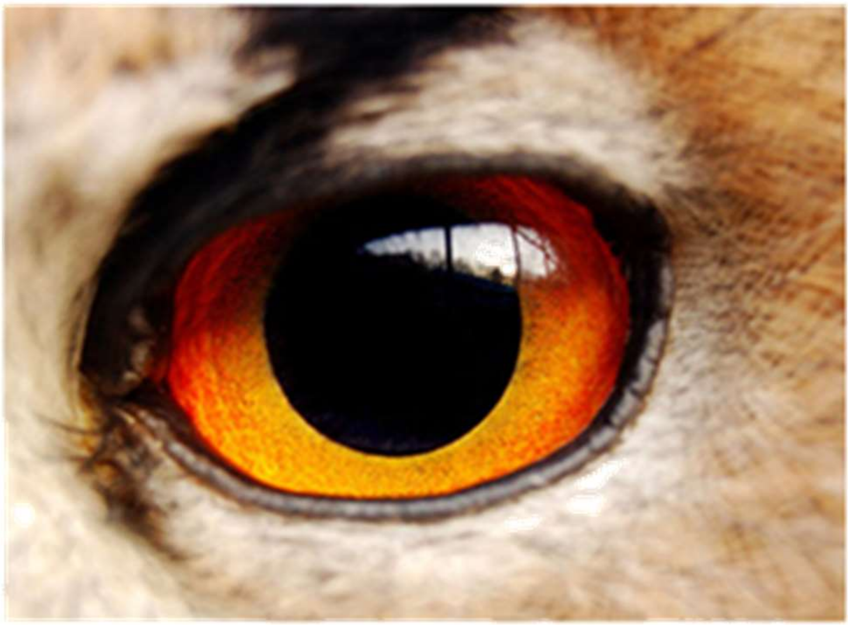


Arbitrary Content

- Vector graphics is hard to make
  - General and fast

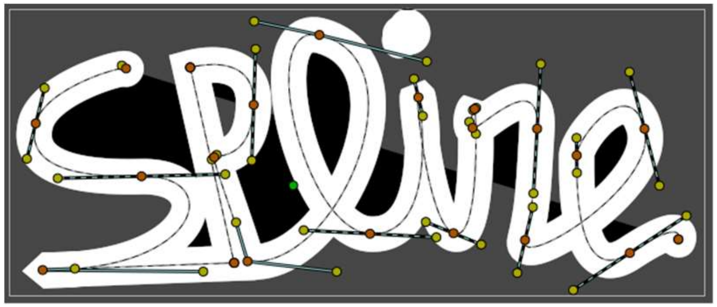


- Raster graphics is hard
  - To edit meaningfully
  - To store efficiently



Vector Graphics is used by Software

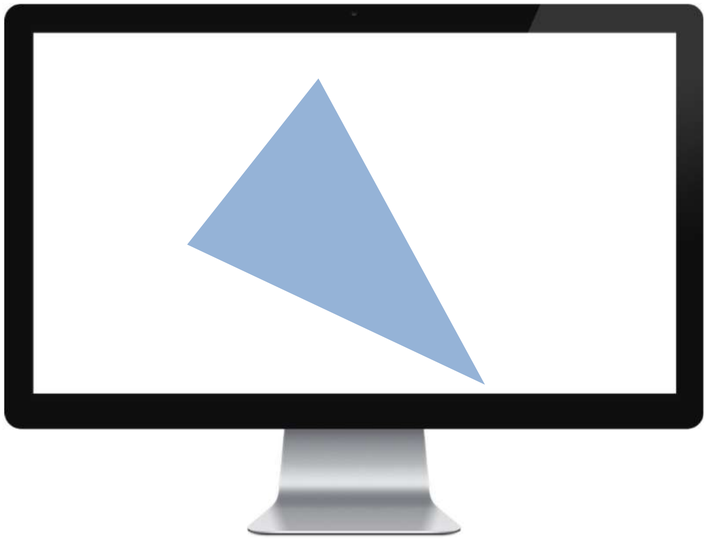
- True Type Fonts



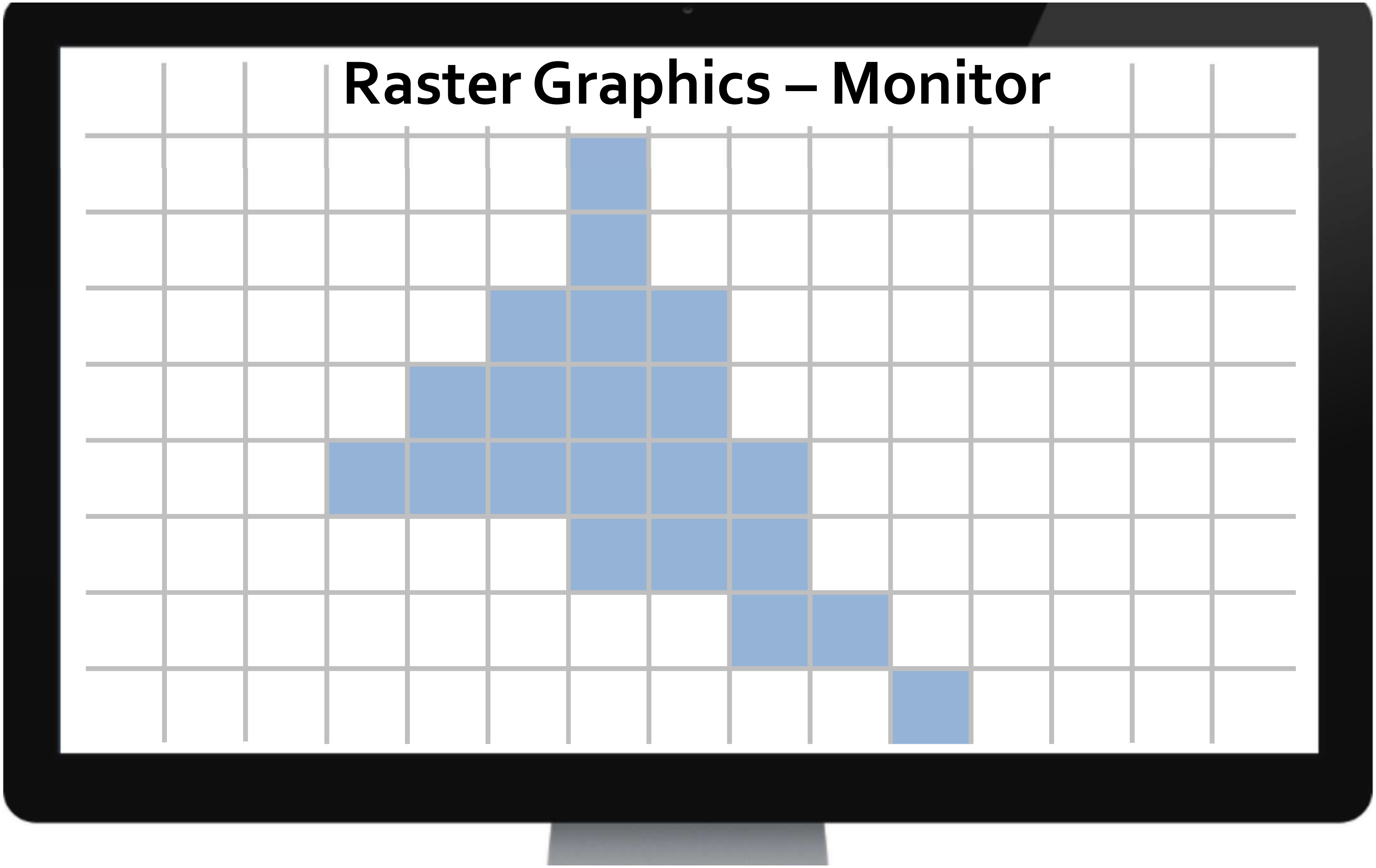
- Illustrator

- Maya

Raster Graphics – Monitor

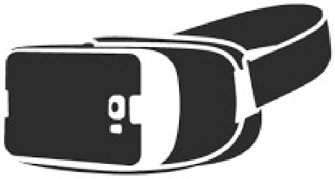


Raster Graphics – Monitor



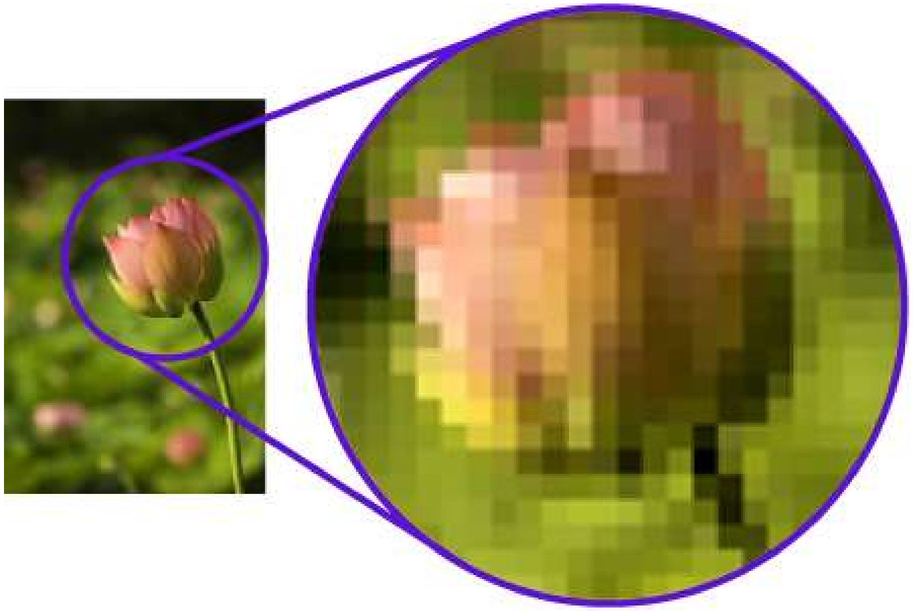
Raster Graphics is used by Hardware

- Monitor
- Handy
- TV
- Digicam
- Printer
- Scanner
- VR/AR
  - Google Glass
  - Holo Lens
- Mouse
- …



Why is it used by hardware?

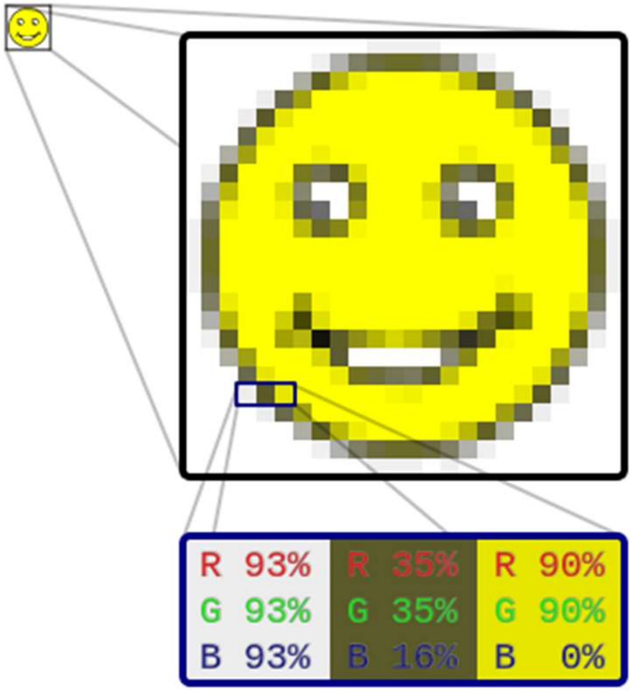
- Easy and cheap to produce
- Very fast
- Arbitrary content

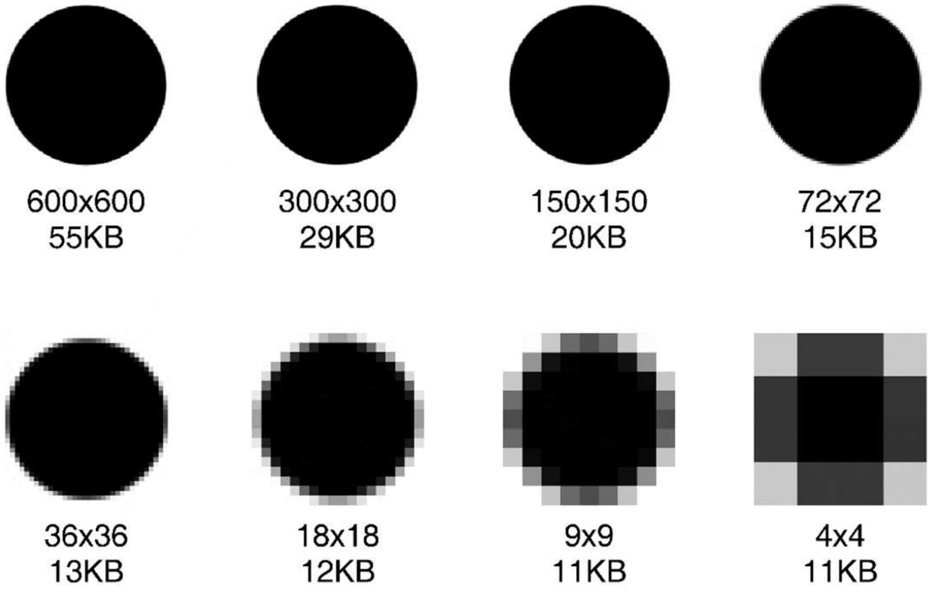


Storing Raster Graphics

Raster Graphics

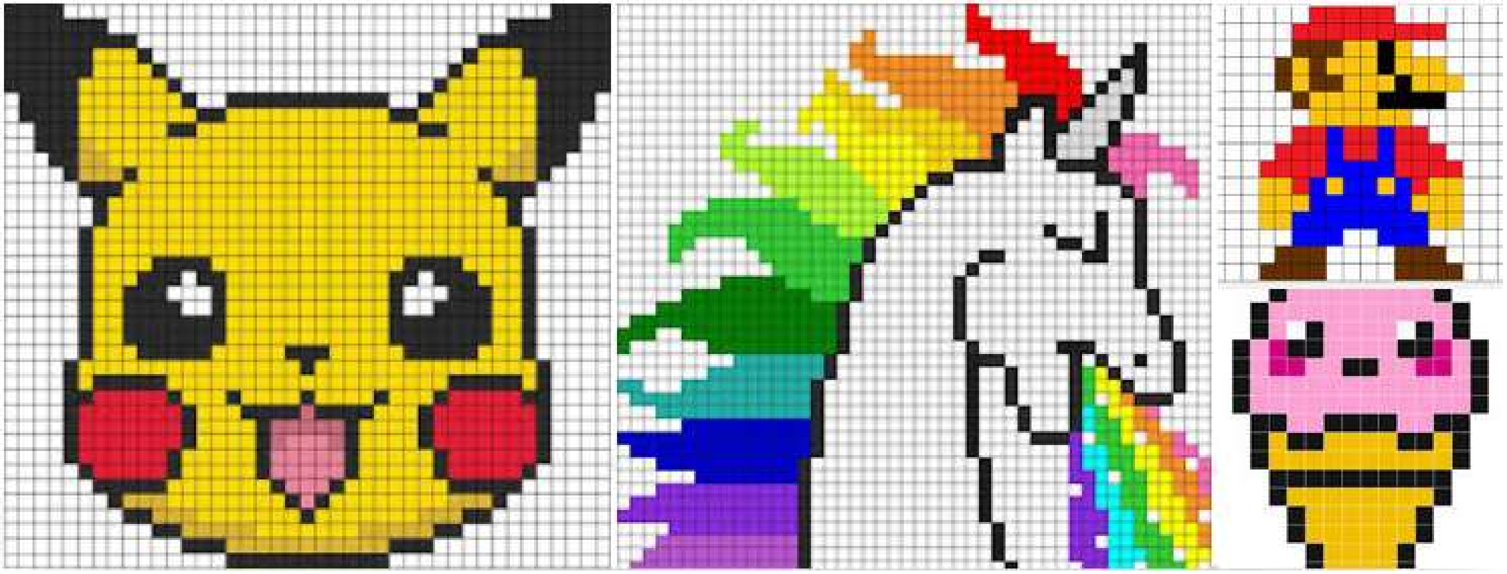
- Raster Image = rectangular grid of colored elements
- Higher realisme = higher memory requirements





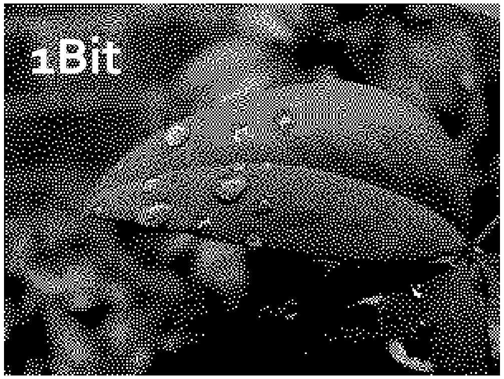
Pixel

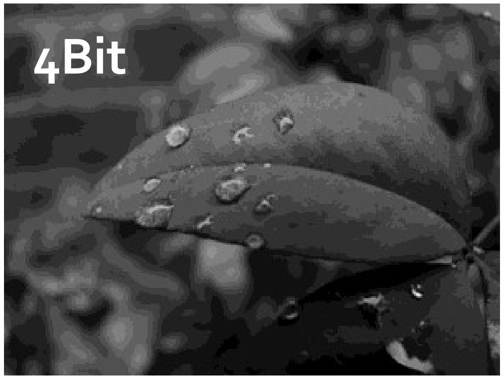
- "Picture element"
- Physical point in a raster image
- Certain amount of bits per pixel

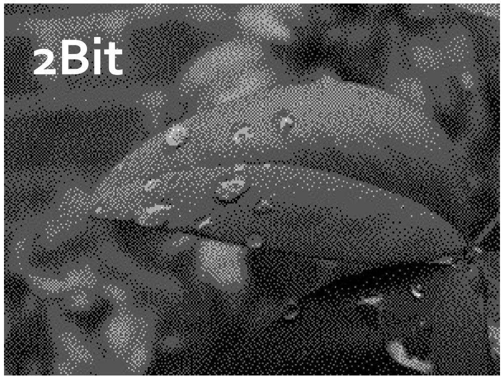


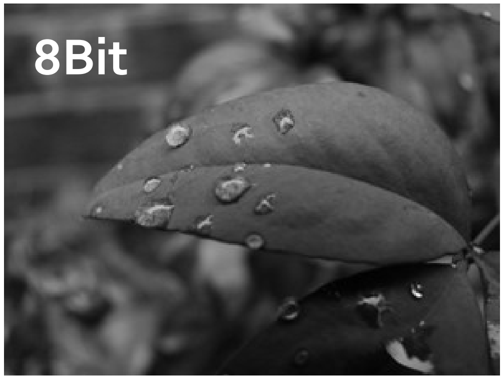
Bits per Pixel (Bpp)

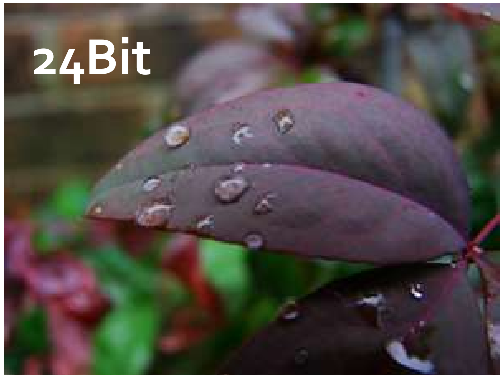
- Amount of bits used to store color information









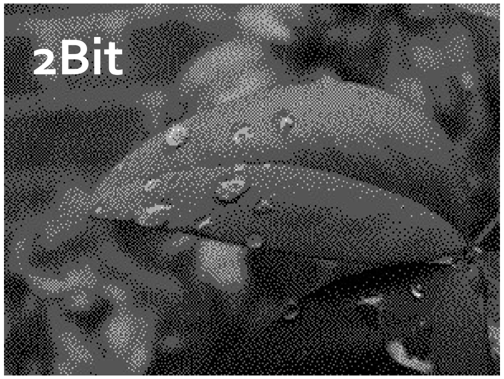


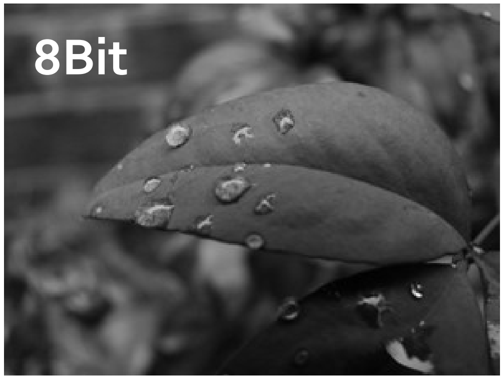
Bits per Pixel (Bpp)

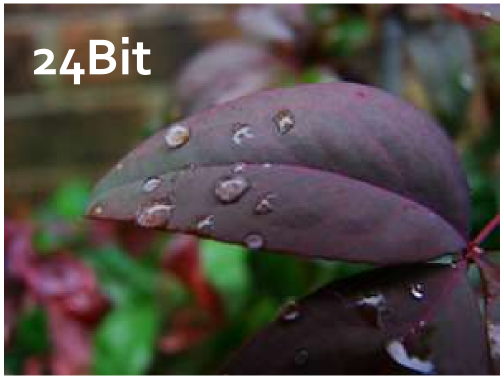
0
1
2
3

0
5
230
255

8bit	8bit	8bit
0	0	0
255	0	0
0	255	255
...	...	...







Indexed Colors / Color Tables

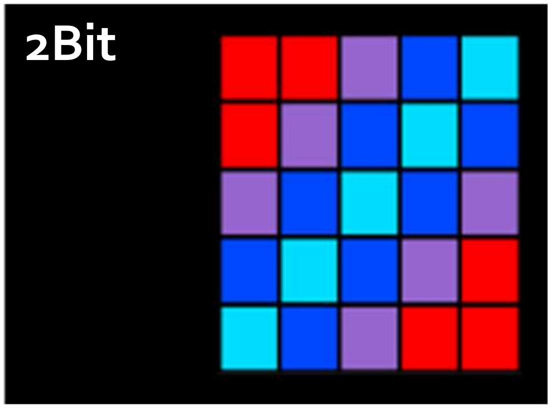
0	0	1	2	3
0	1	2	3	2
1	2	3	2	1
2	3	2	1	0
3	2	1	0	0

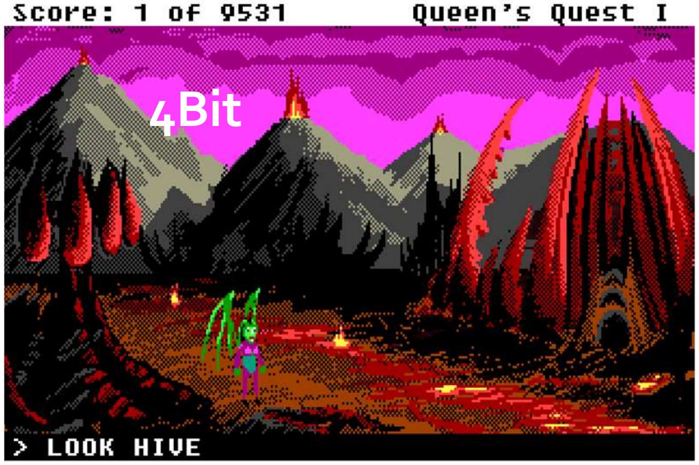
0 = 🟥

1 = 🟪

2 = 🟦

3 = 🟦





Score: 1 of 9531 Queen's Quest I

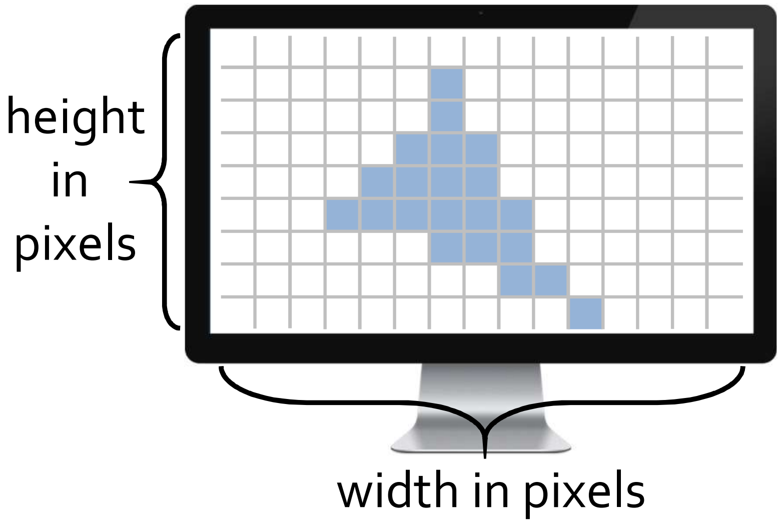
> LOOK HIVE

Color Table Animations

- Cycle through color table entries over time

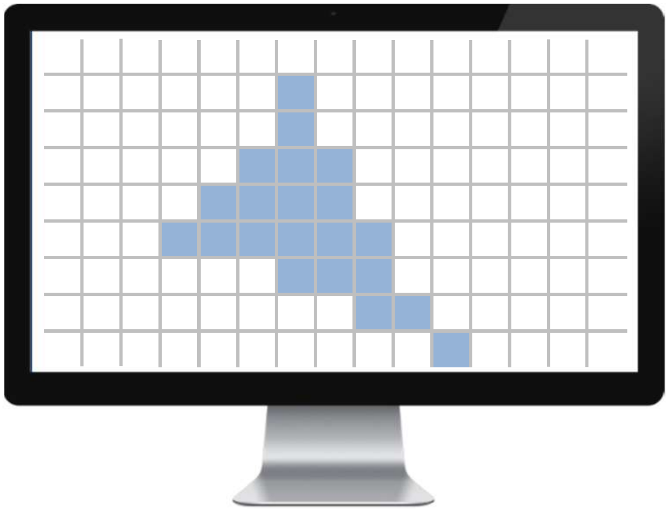
Frame Buffer

- A.k.a. frame store
- Raster image of monitor input
- Portion of RAM (often in video memory)
- Resolution
  - Width x height of pixels
  - VGA = 640*480
  - XGA= 1024*768
  - HD=1280*720
  - FullHD = 1920*1080



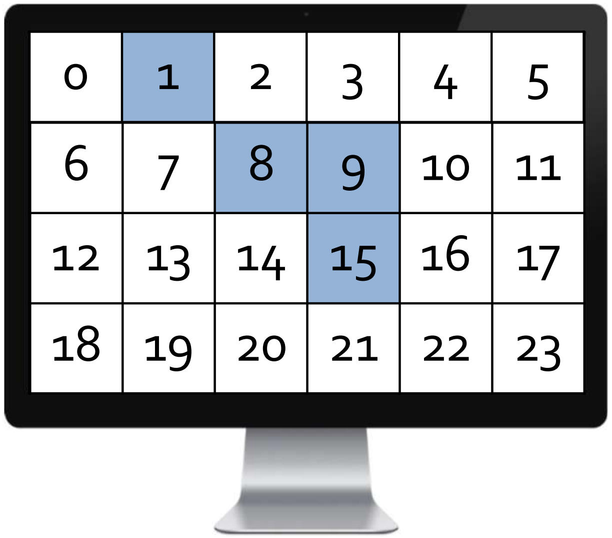
Frame Buffer Resolution

- Width x height of pixels
- VGA = 640*480, 8bit per pixel
  - 640*480*1 = **307KB**
- XGA= 1024*768, 16bit per pixel
  - 1024*768*2= **1,5MB**
- HD=1280*720, 24bit per pixel
  - 1280*720*3 = **2,6MB**
- FullHD = 1920*1080, 32bit per pixel
  - 1920*1080*4= **8MB**
- 4k = 3840*2160, 32bit per pixel
  - 3840*2160 *4= **32MB**



Frame Buffer

- A.k.a. frame store
- Raster image of monitor input
- Portion of RAM (often in video memory)
- RAM is usually 1 dimensional and linear



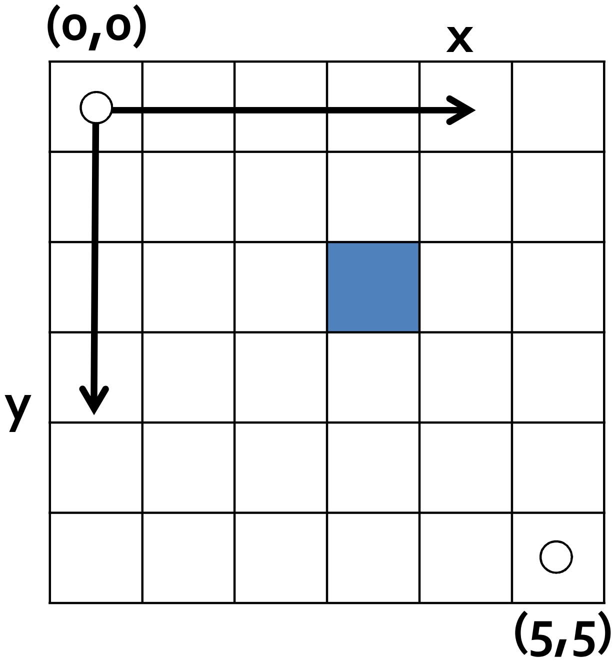
0	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	17
18	19	20	21	22	23

RAM (8Bit color)

#	Data
0	255
1	120
2	255
3	255
4	255
5	255
6	255
7	255
8	120
...	

Drawing a Pixel

- Given is a pixel by coordinates and color
  `DrawPixel(x, y, color)`



Drawing a Pixel

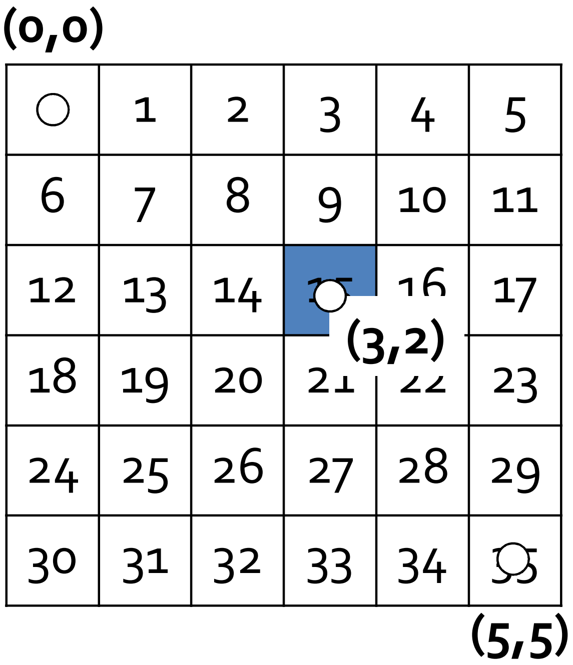
- Color assignment to location (memory address) in frame buffer

```
frameBuffer[addr] = BLUE;
```

- Calculate address?

```
addr = y * width + x
```

- Works for 8 bits per pixel
  - 1 pixel = 1 byte
- Otherwise multiply with size
  - 16bpp – 1 pixel = 2 byte
  - 24bpp – 1 pixel = 3 byte

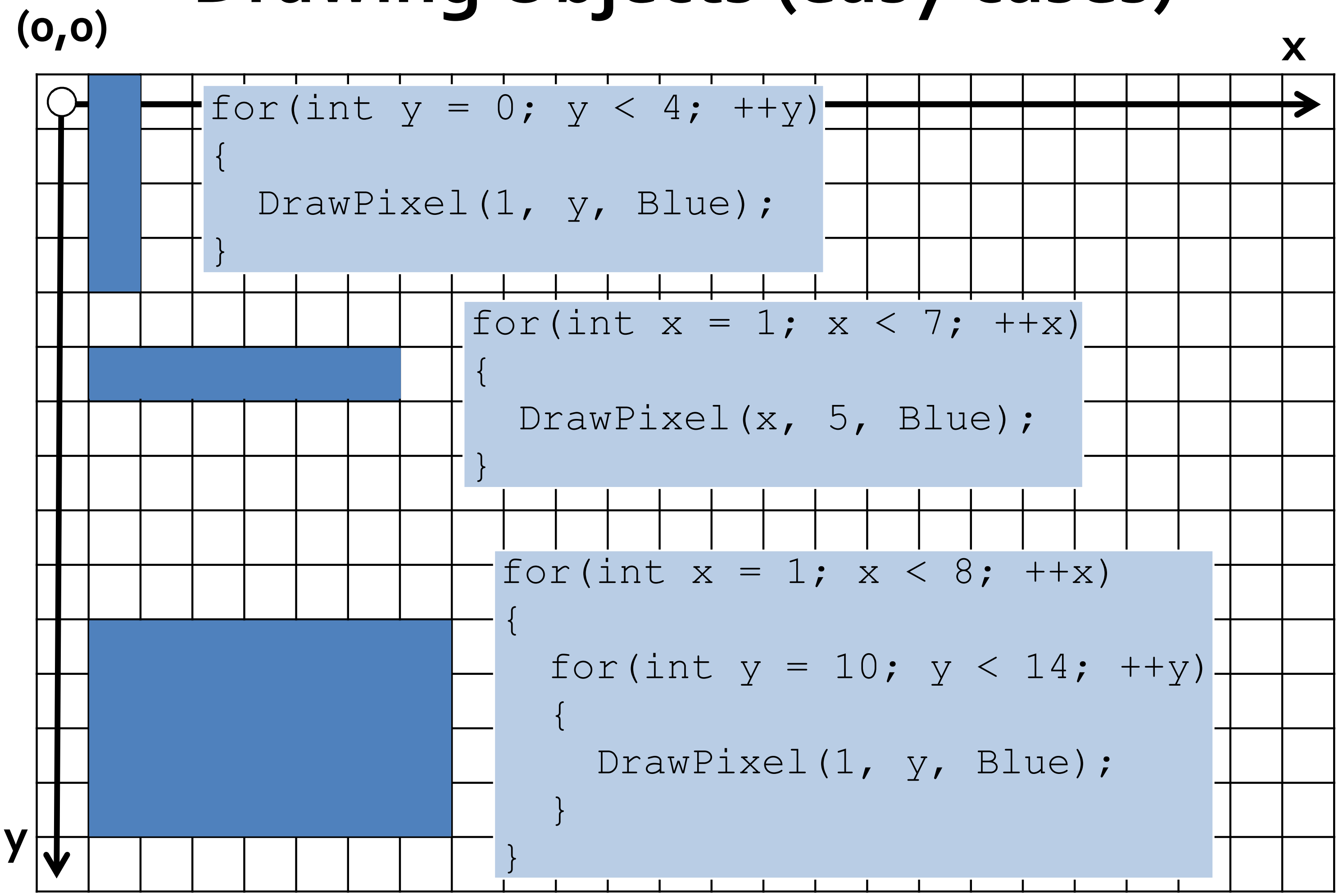


○	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	17
18	19	20	21	22	23
24	25	26	27	28	29
30	31	32	33	34	35

RAM (8Bit color)

#	Data
0	255
1	120
2	255
3	255
4	255
5	255
6	255
7	255
8	120
…	

Drawing Objects (easy cases)

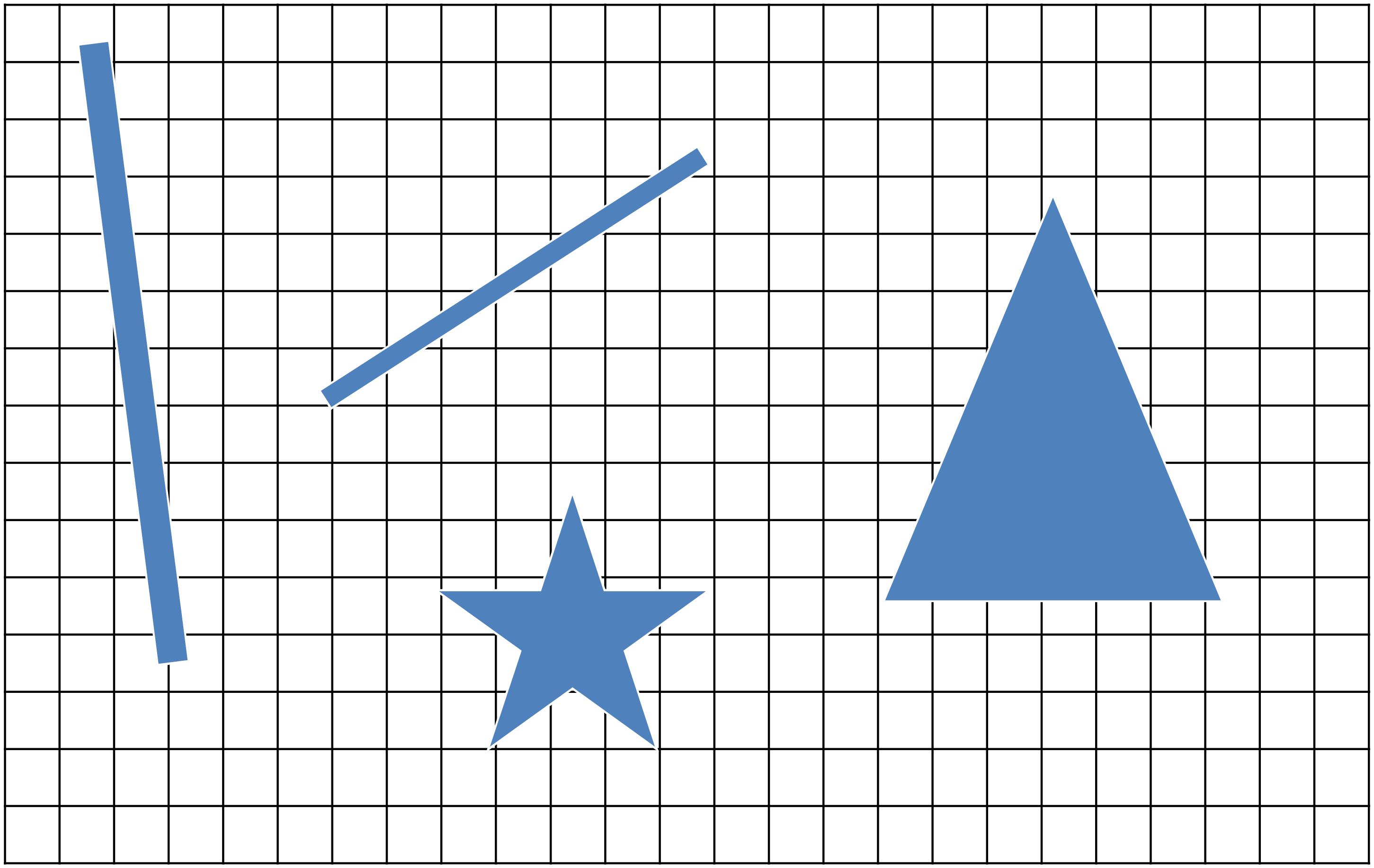


```
for(int y = 0; y < 4; ++y)
{
  DrawPixel(1, y, Blue);
}
```

```
for(int x = 1; x < 7; ++x)
{
  DrawPixel(x, 5, Blue);
}
```

```
for(int x = 1; x < 8; ++x)
{
  for(int y = 10; y < 14; ++y)
  {
    DrawPixel(1, y, Blue);
  }
}
```

Drawing Objects (normal cases)

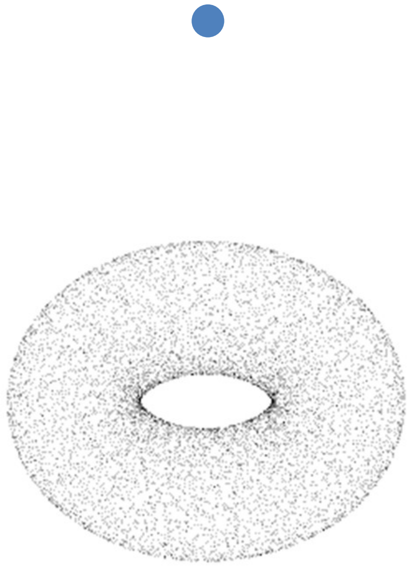


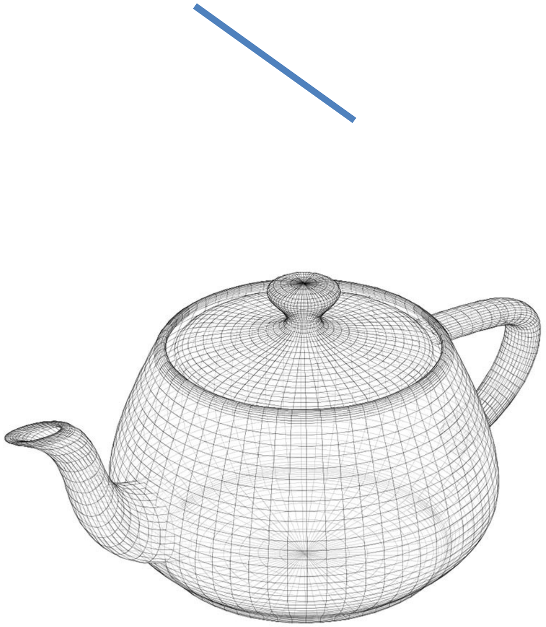
From Software to Hardware

From Vector Graphics to Raster Graphics

Primitives

- Hardware renders only certain types of geometric primitives
  - Often point, line, triangle

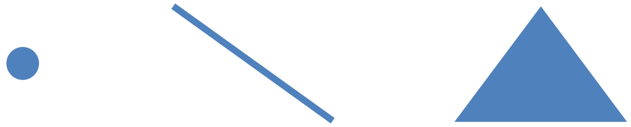




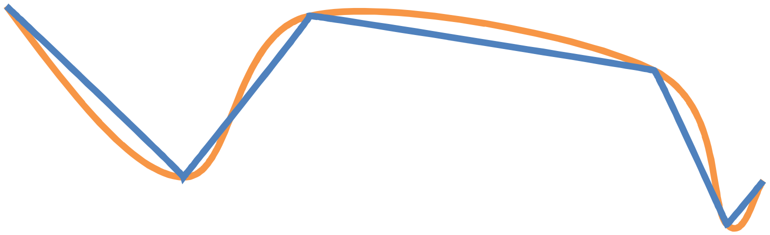
Why Only Certain Primitives?

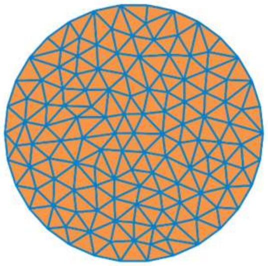
- Useful properties

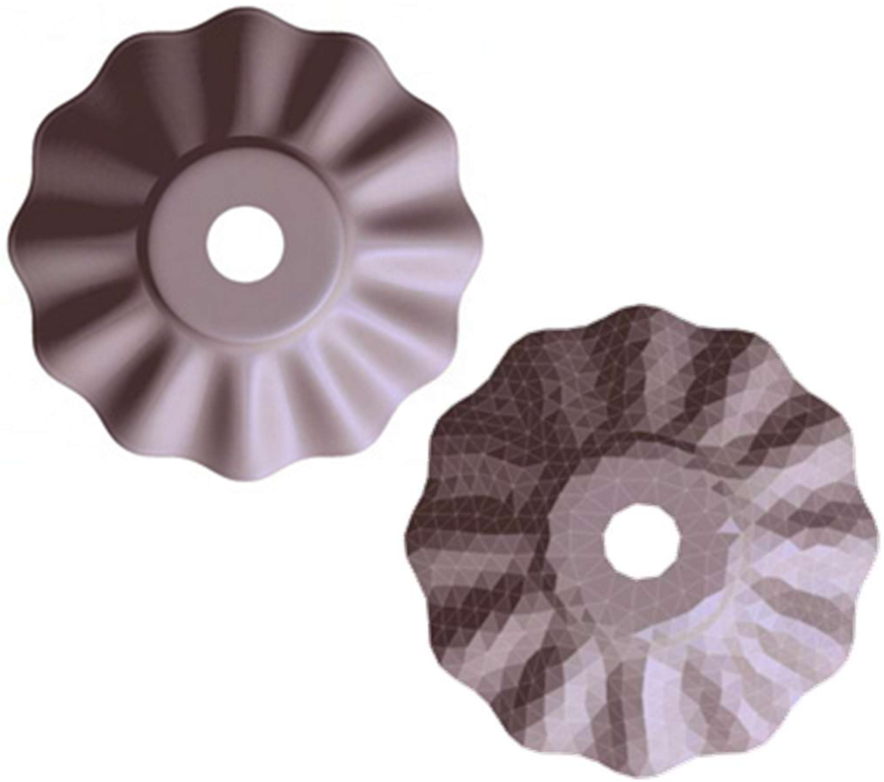
1. Easy to specify
2. Always convex(?) and planar(?)
3. Exist in 2D and 3D
4. Each primitive costs transistors on chip ⟶ limits number
5. All curves can be approximated by lines
6. All polygons can be broken into triangles



Other Geometry is Converted Into Primitives

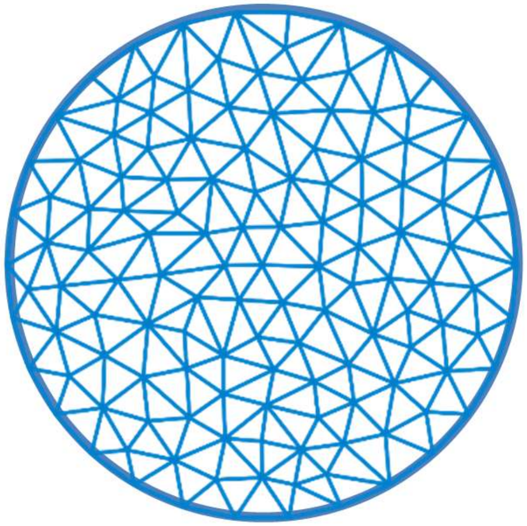


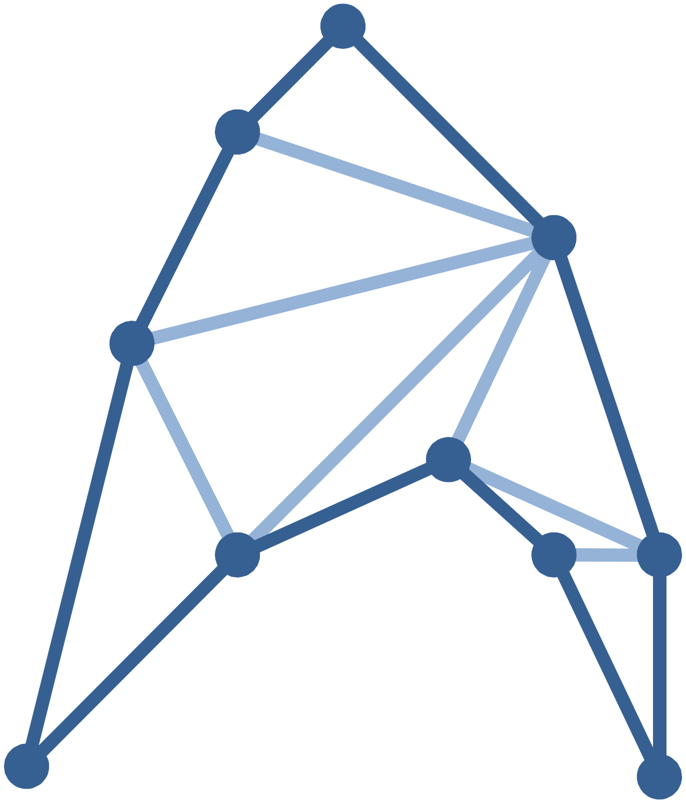




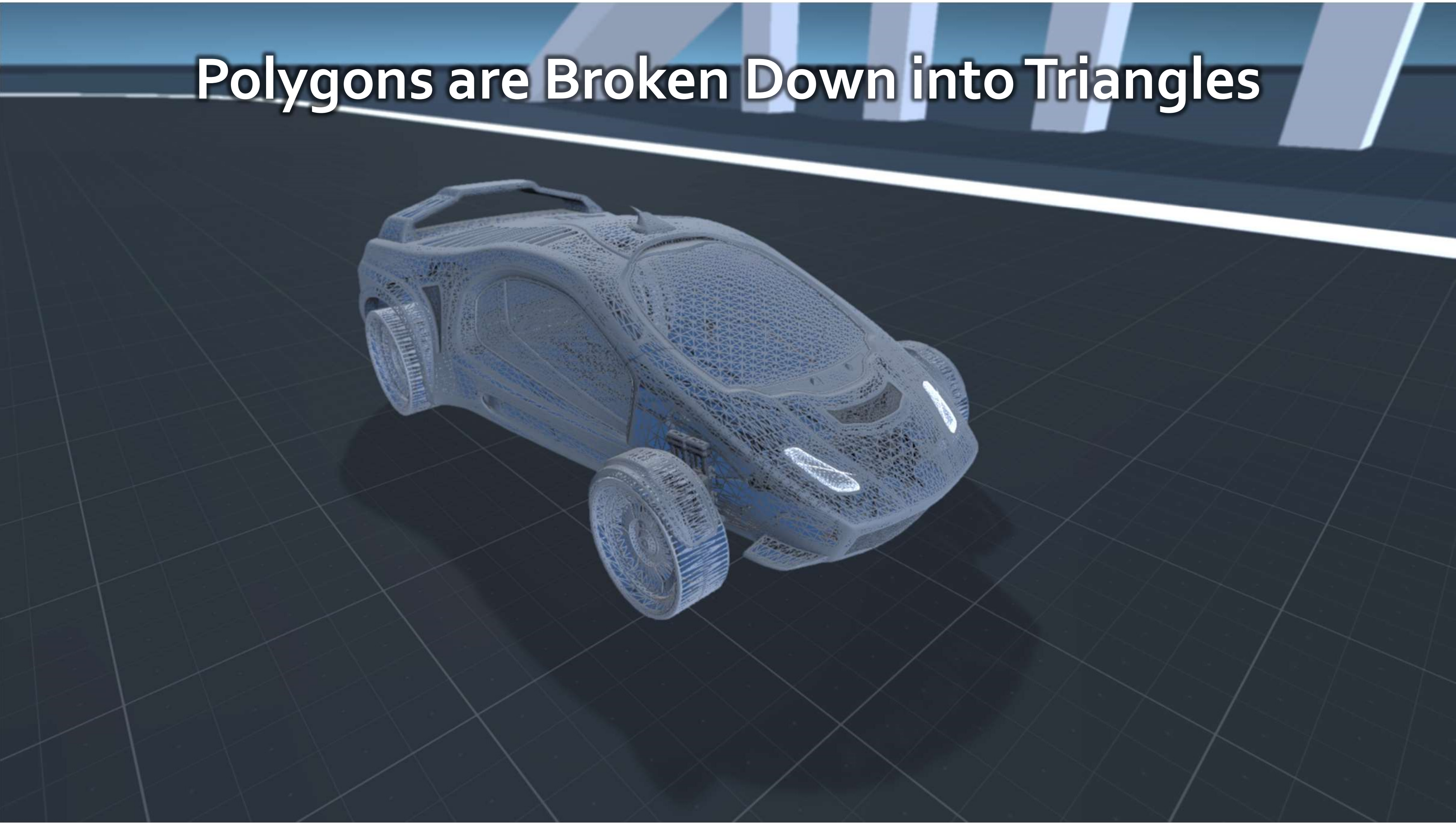
Triangulation

- Breaks a polygon into triangles
  - Delaunay-Triangulation

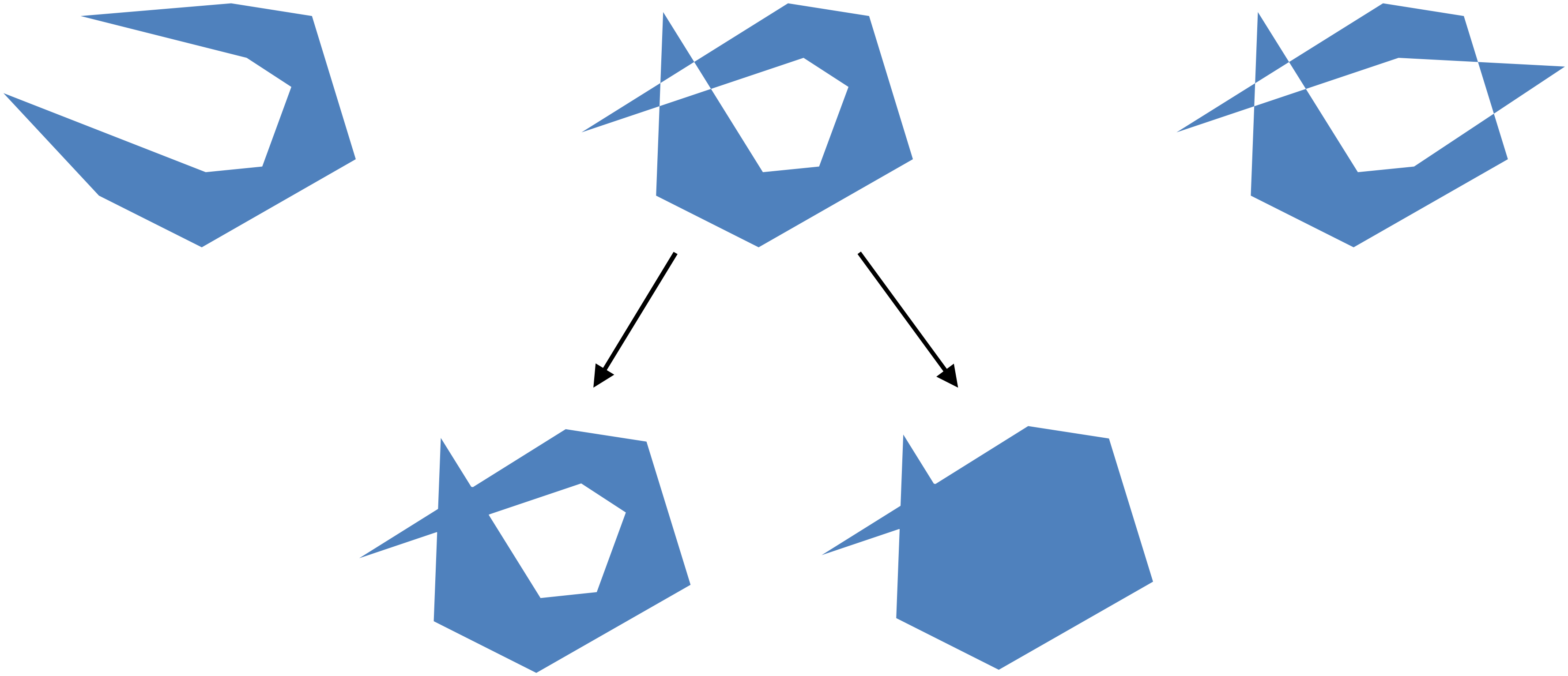




Polygons are Broken Down into Triangles

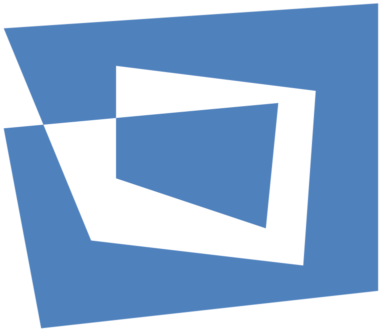


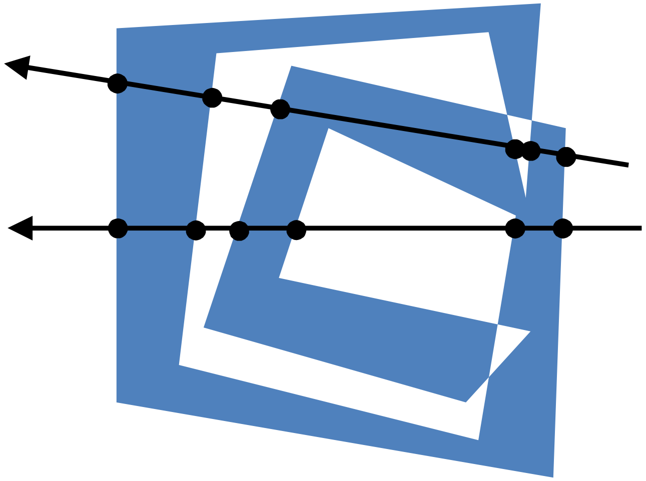
What is Inside a Polygon?



Odd-Even Rule

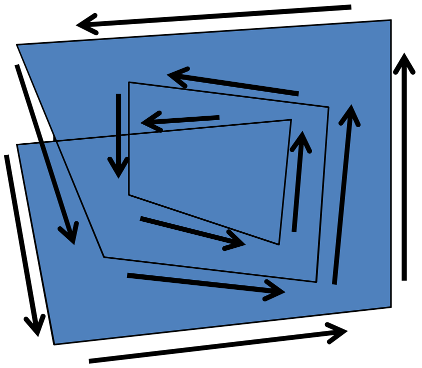
- Inside/outside switches at every edge
- Straight line to the outside:
  - Even # edge intersections = outside
  - Odd # edge intersections = inside

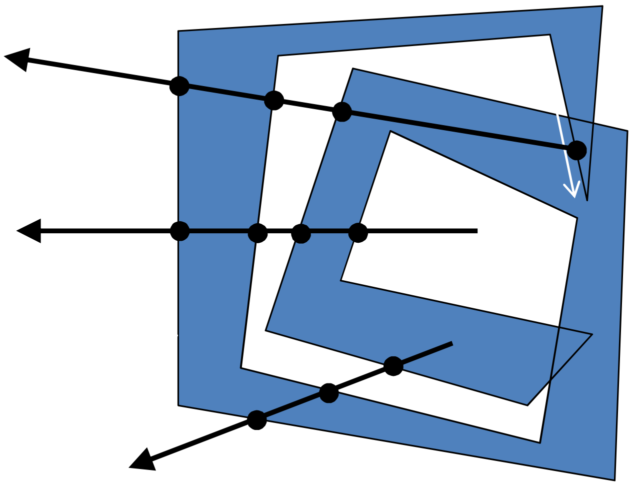




Nonzero Winding Number

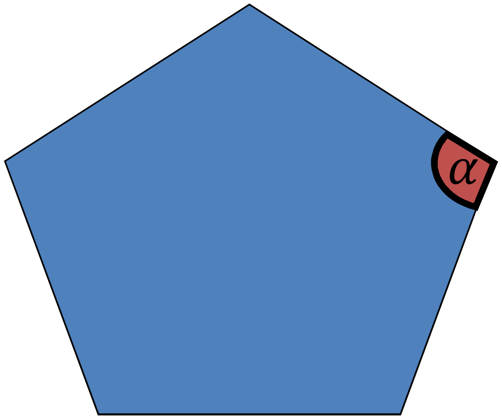
- Point is inside if polygon surrounds it
- Straight line to the outside:
  - same # edges up and down = outside
  - different # edges up and down = inside



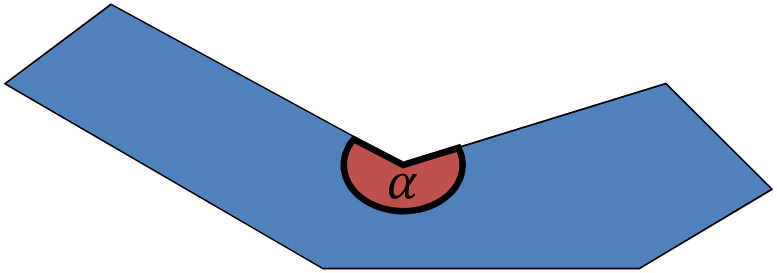


Polygon Classification

- **Convex**: no interior angle > 180°



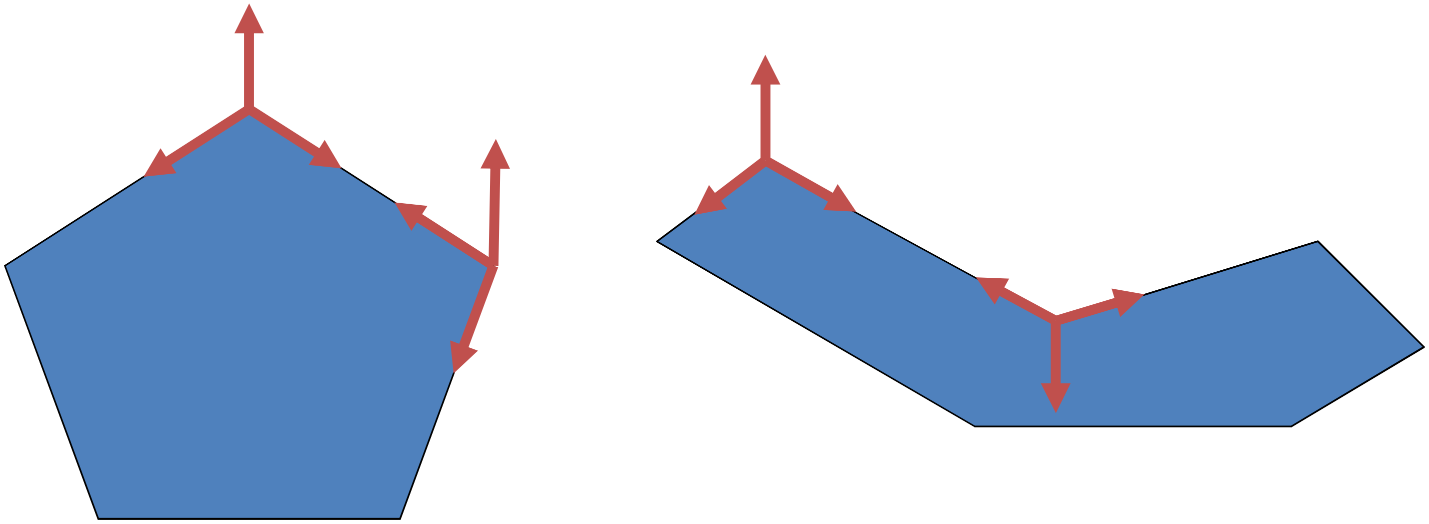
- **Concave**: not convex



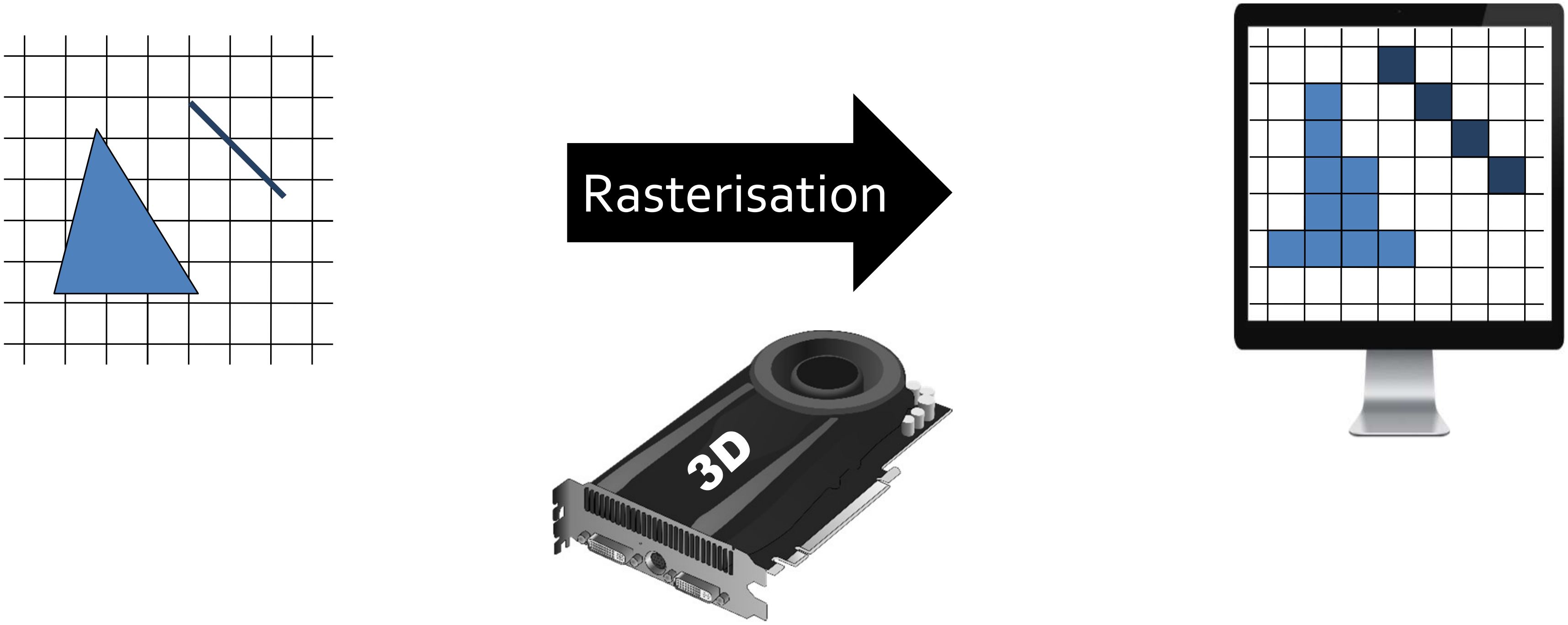
Polygon Classification – Vector Test

- All vector cross products have the same sign

  ⟹ convex



From Primitives To Pixels



Rasterisation