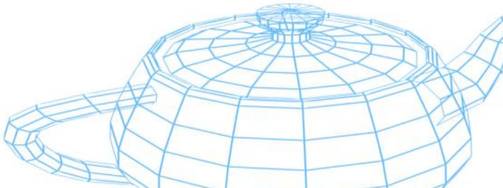
SUPSI

Computer Graphics

2D File Formats

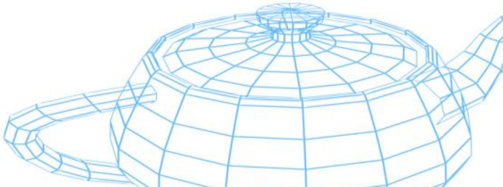
Achille Peternier, lecturer



Raster VS vector graphics

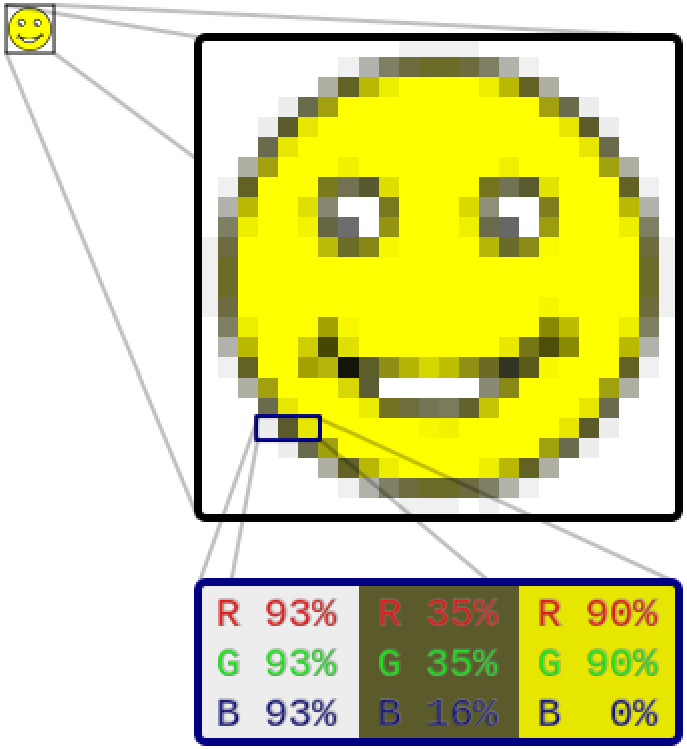
- There are two main ways for defining and storing images:
  - Using **raster** graphics (mosaic/framebuffer-like approach).
  - Using **vector** graphics (OpenGL primitive-like approach).

- Hybrid approaches also exist and combine both techniques into one same format:
  - Raster or vector graphics are used according to the needs:
    - E.g.: digital camera images = raster graphics, text paragraphs = scalable vector fonts.



Raster graphics

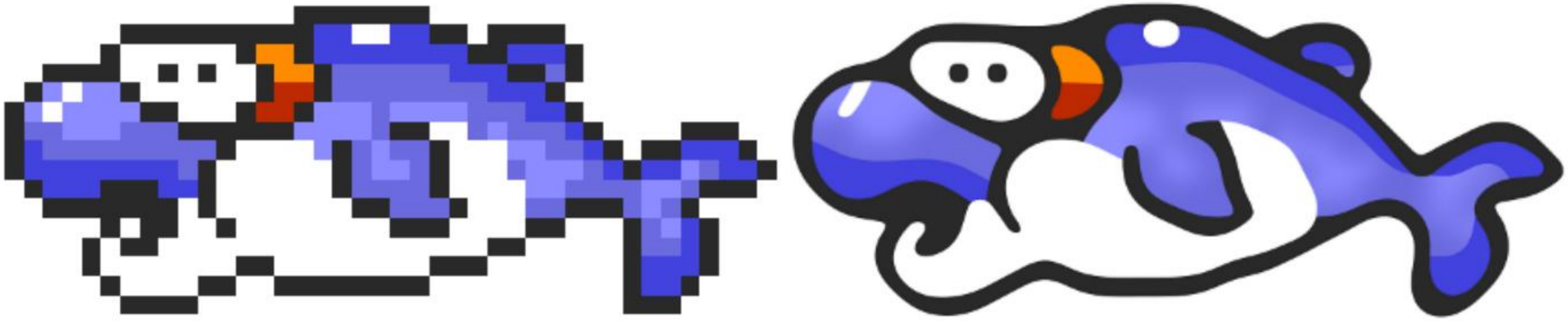
- Raster graphics are composed of a series of pixels, like the OpenGL framebuffer or a mosaic.



Raster graphics

- Typical raster images are acquired through camera sensors/scanners, or edited pixel by pixel by artists.

- Images have a fixed resolution:
  - Magnification/minification introduce aliasing and other artifacts:
    - Use filters to improve quality (e.g., linear filtering, as used in texture mapping).
    - Use specific filters for scaling pixel-art content:

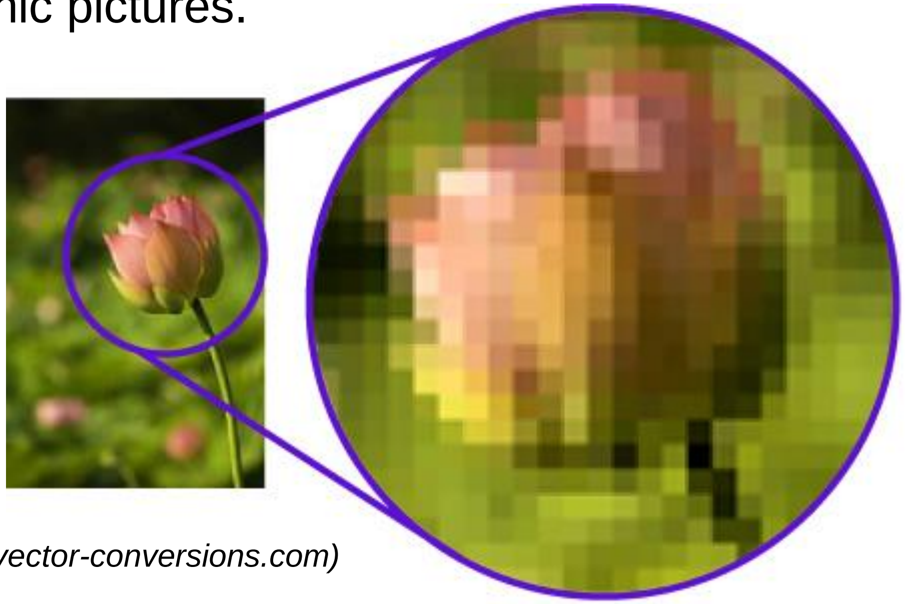


(Microsoft Research)

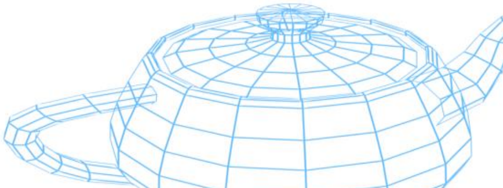
Raster graphics

- Raster data is often compressed to reduce size:
  - Uncompressed raster data corresponds to a bitmap of size $width \times height \times colorDepth$ (where *colorDepth* is the number of bytes per pixel, usually RGB with 1 byte per channel).

- Raster formats are mainly used for storing unstructured image data like photographic pictures.



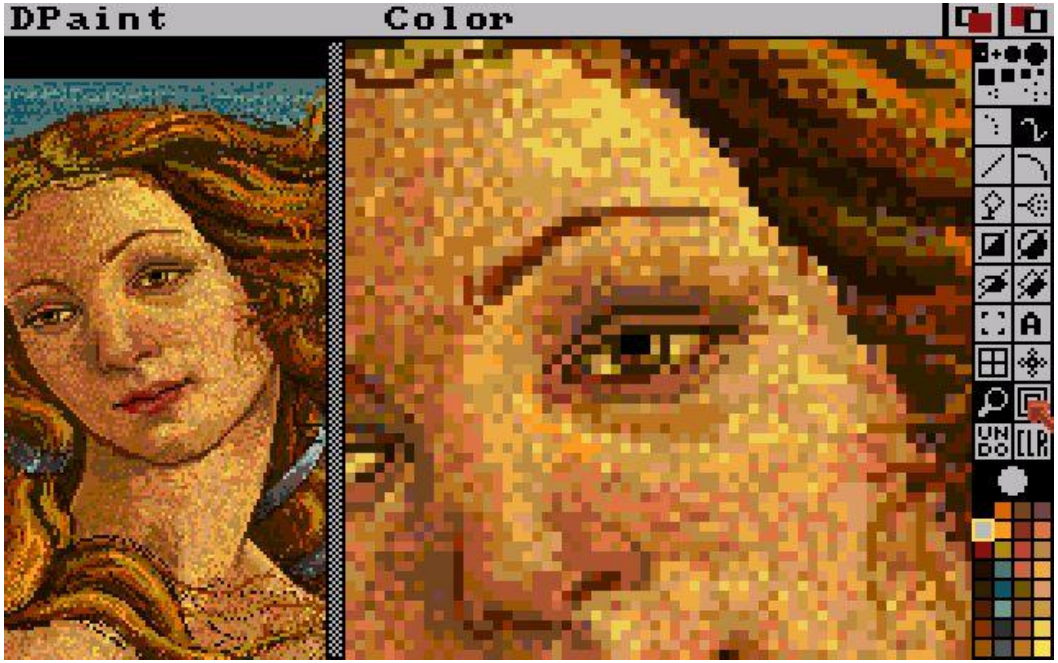
(http://vector-conversions.com)



Raster graphics

- Raster graphics image editors are Photoshop, GIMP, MS Painter, Corel Painter, Deluxe Paint, etc.

- Examples of raster-based formats are BMP, JPEG, PNG, TGA, etc.



Palette

- To reduce the number of bytes required for storing each pixel, a (limited) palette of colors can be used.
- The palette works like a look-up table:

Image (no palette)

🟩 0 255 0	🟦 0 0 255	🟥 0 255 0	🟥 0 255 0	🟥 0 255 0	🟥 0 255 0

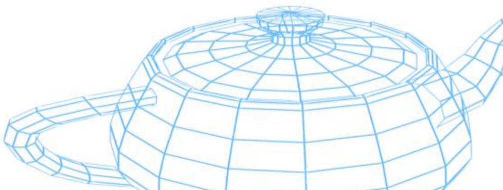
18 bytes

Image (using palette)

🟩 2	🟦 1	🟥 3	🟥 3	🟥 3	🟥 3

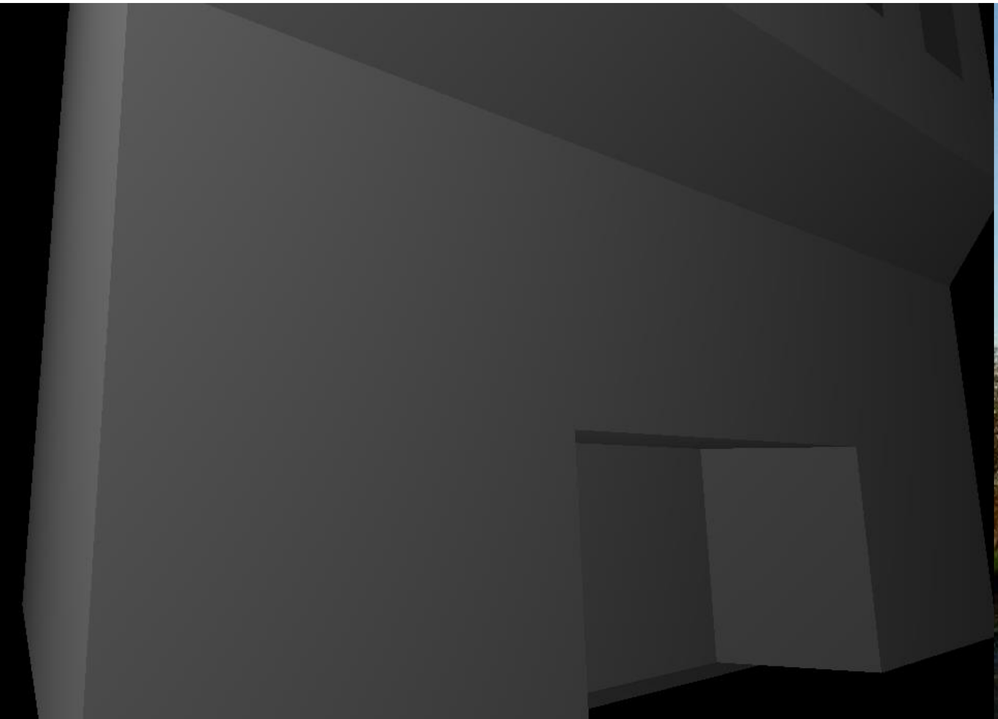
6 bytes

Palette ID	Color
1	0 0 255
2	0 255 0
3	255 0 0



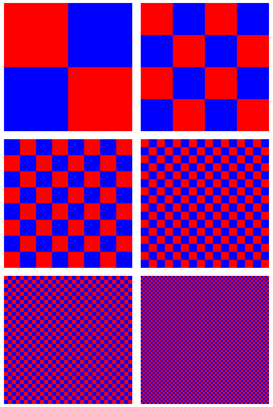
Color banding

- Problem due to the lack of precision in the level of gradients used to shade a color (e.g., when 8-16 bit color depth images are generated).



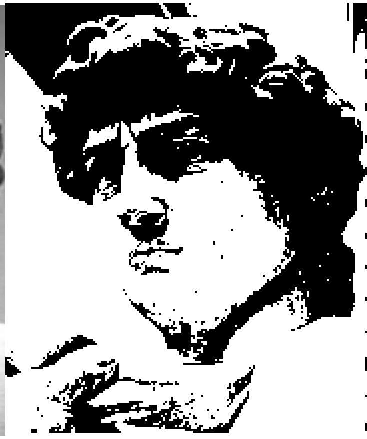
Dithering

- Dithering is a form of controlled distributed noise to reduce artifacts introduced by the lack of resolution:
  - For example, when a lower color depth is used for converting a grey scale image to black and white, or for simulating colors that are not available in the palette.

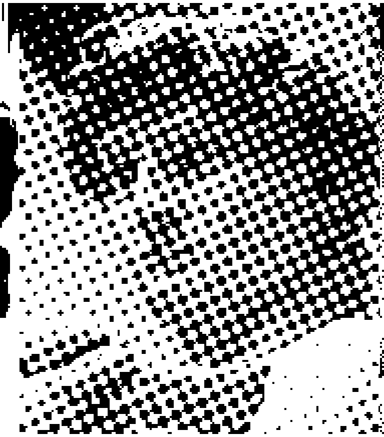


red and blue pattern

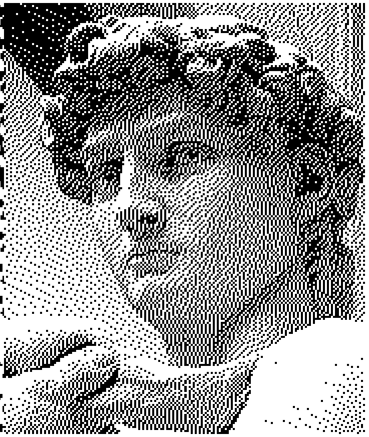
Original



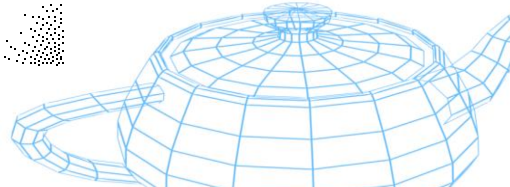
Threshold



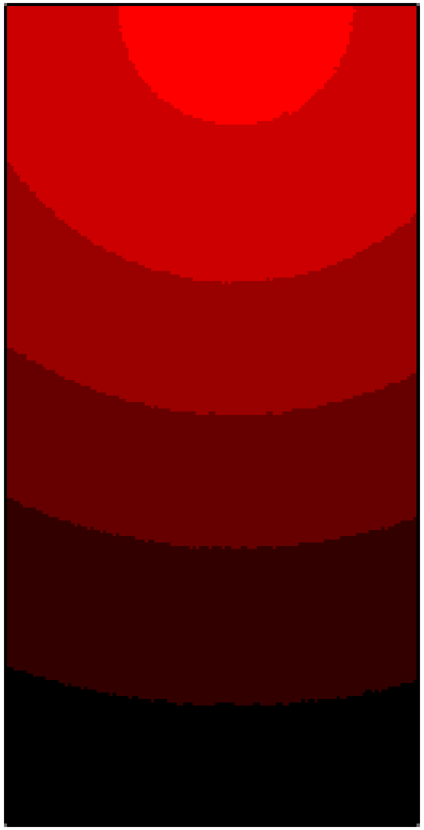
Halftones



Sierra

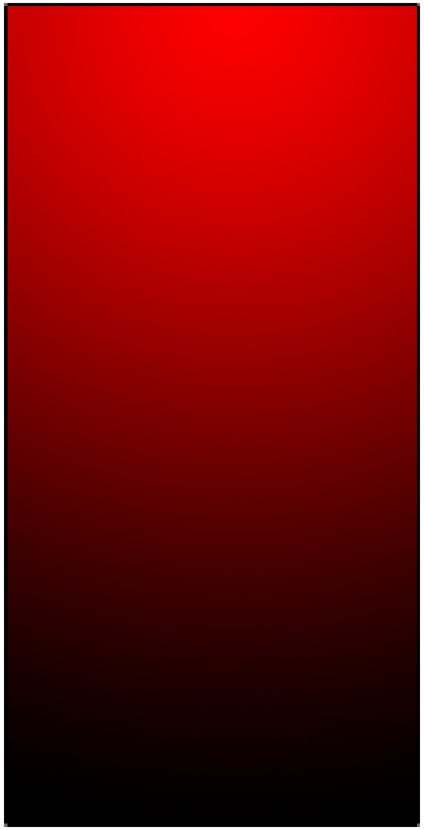


Dithering



low res

low res + dithering

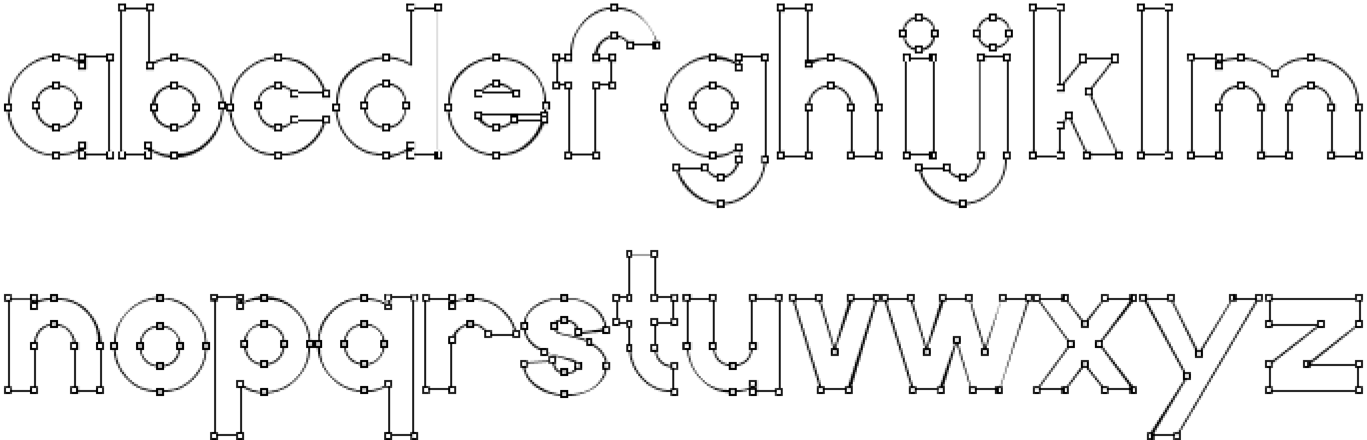


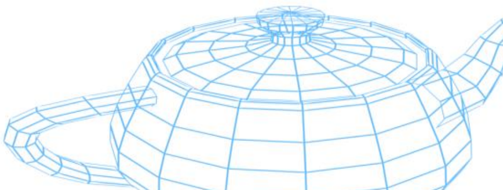
high res

Vector graphics

- Represented by geometric primitives like points, lines, curves, etc.

- Conceptually similar to OpenGL: you pass a series of primitives that are evaluated and their final resolution depends on the screen size and the various matrices involved:
  - Images are dynamically generated according to the current resolution and other parameters.
  - Perfect match between screen (150-300 DPI) and printer (>1200 DPI) resolution.

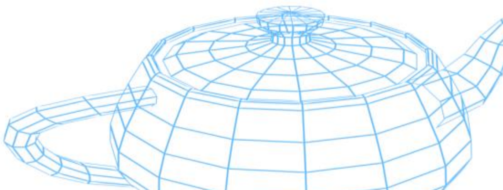




Vector graphics

- Vector graphics is particularly suitable for storing and defining structured data like blueprints, technical drawings, glyphs, clip art, logos, maps, etc.

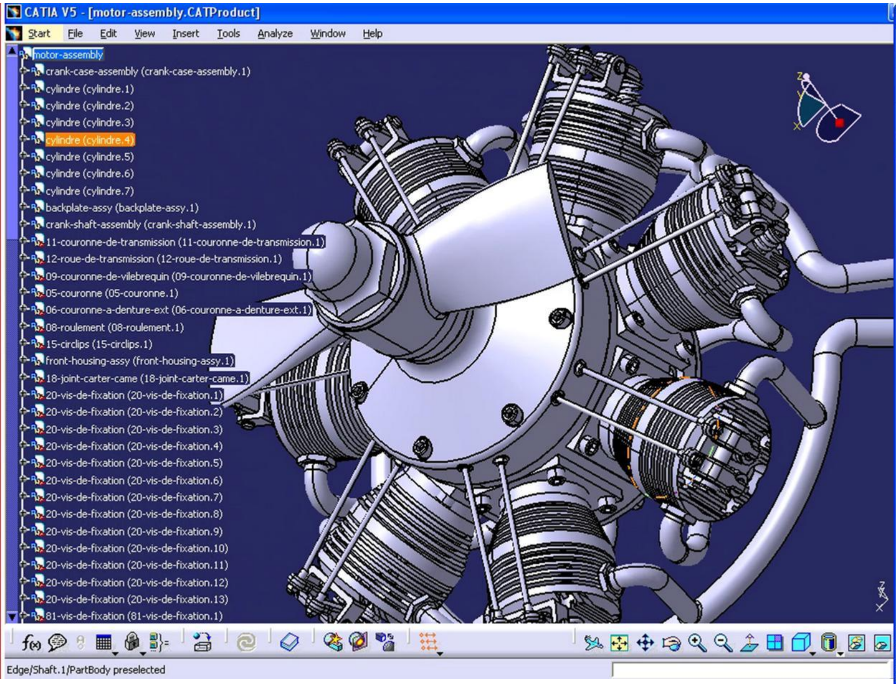
- Good information/storage ratio:
  - No need for compression, although it is always possible.
  - Bigger/smaller images have exactly the same size.

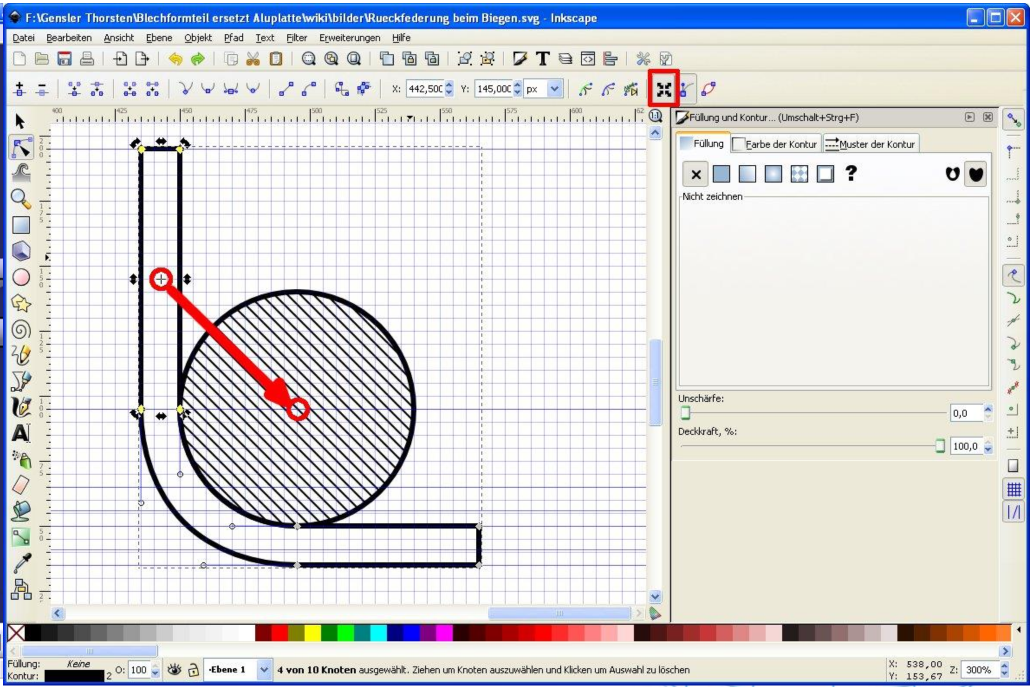


Vector graphics

- Vector graphics editors are CAD applications (Catia, AutoCAD), Inkscape, Corel DRAW, Adobe Illustrator, Latex, etc.

- Examples of vector graphics file formats are SVG, EPS, CGM, DXF, etc.

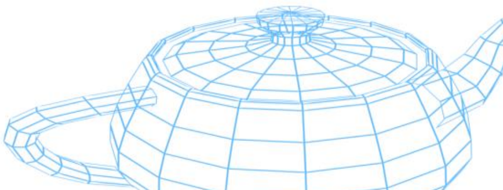

CATIA V5 - [motor-assembly.CATProduct]
Start File Edit View Insert Tools Analyze Window Help
motor-assembly
crank-case-assembly (crank-case-assembly.1)
cylindre (cylindre.1)
cylindre (cylindre.2)
cylindre (cylindre.3)
cylindre (cylindre.4)
cylindre (cylindre.5)
cylindre (cylindre.6)
cylindre (cylindre.7)
backplate-assy (backplate-assy.1)
crank-shaft-assembly (crank-shaft-assembly.1)
11-couronne-de-transmission (11-couronne-de-transmission.1)
12-roue-de-transmission (12-roue-de-transmission.1)
09-couronne-de-vilebrequin (09-couronne-de-vilebrequin.1)
05-couronne (05-couronne.1)
06-couronne-a-denture-ext (06-couronne-a-denture-ext.1)
08-roulement (08-roulement.1)
15-circlips (15-circlips.1)
front-housing-assy (front-housing-assy.1)
18-joint-carter-came (18-joint-carter-came.1)
20-vis-de-fixation (20-vis-de-fixation.1)
20-vis-de-fixation (20-vis-de-fixation.2)
20-vis-de-fixation (20-vis-de-fixation.3)
20-vis-de-fixation (20-vis-de-fixation.4)
20-vis-de-fixation (20-vis-de-fixation.5)
20-vis-de-fixation (20-vis-de-fixation.6)
20-vis-de-fixation (20-vis-de-fixation.7)
20-vis-de-fixation (20-vis-de-fixation.8)
20-vis-de-fixation (20-vis-de-fixation.9)
20-vis-de-fixation (20-vis-de-fixation.10)
20-vis-de-fixation (20-vis-de-fixation.11)
20-vis-de-fixation (20-vis-de-fixation.12)
20-vis-de-fixation (20-vis-de-fixation.13)
81-vis-de-fixation (81-vis-de-fixation.1)
Edge/Shaft.1/PartBody preselected


F:\Gensler Thorsten\Blechformteil ersetzt Aluplatte\wiki\bilder\Rueckfederung beim Biegen.svg - Inkscape
Datei Bearbeiten Ansicht Ebene Objekt Pfad Text Filter Erweiterungen Hilfe
X: 442,500 Y: 145,000 px
Füllung und Kontur... (Umschalt+Strg+F)
Füllung Farbe der Kontur Muster der Kontur
Nicht zeichnen
Unschärfe:
0,0
Deckkraft, %:
100,0
Füllung: Keine
Kontur:
O: 100
Ebene 1
4 von 10 Knoten ausgewählt. Ziehen um Knoten auszuwählen und Klicken um Auswahl zu löschen
X: 538,00 Y: 153,67 Z: 300%

Hybrid approaches

- Several formats use the most suitable technique depending on the case:
  - E.g., using raster data for storing photos and vector data for storing fonts.

- Examples of hybrid file formats include PDF, PS, etc.

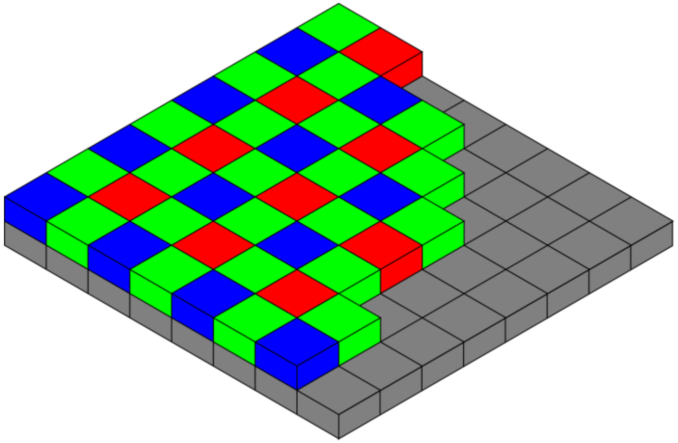


Pure raw data

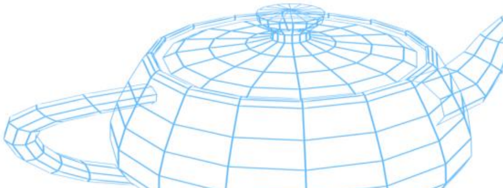
- It’s an unformatted piece of raw graphics memory dumped to file, e.g., framebuffer memory dumped to disk (e.g., **`glReadPixels()`**), camera sensor intensity values, etc.

- No portability, limited usage (only as native format of the application that generated the file).

- Very difficult to encode/decode without some documentation or reverse engineering.

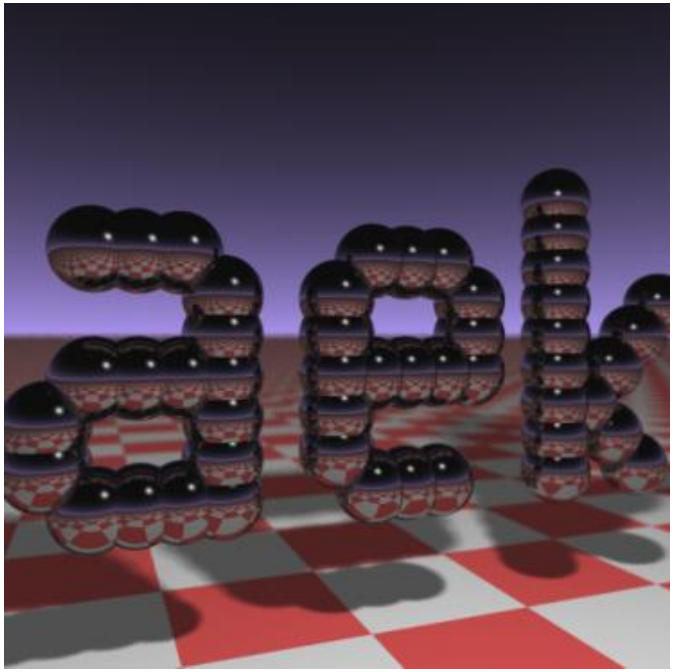


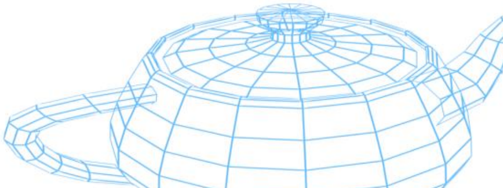
Bayer arrangement on a camera image sensor



PNM

- **P**ortable a**N**y **M**ap file formats:
  - Include PBM, PGM and PPM.
  - Part of the *Netpbm* open package.
- Introduced by Jef Poskanzer in the ‘80s.
- Extremely simple ASCII and binary formats:
  - We already used it in the ray-tracing demo.

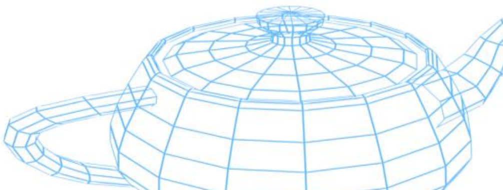




PNM

- 3 ASCII and 3 binary formats defined:
  - Bitmap (.pbm): black and white content
    - 1 bit per pixel.
  - Graymap (.pgm): single channel used for gray shades
    - 8 bits per pixel.
  - Pixmap (.ppm): three channels used (RGB values)
    - 24 bits per pixel.

Header	Type	Content
P1	Portable bitmap	ASCII
P2	Portable graymap	ASCII
P3	Portable pixmap	ASCII
P4	Portable bitmap	Binary
P5	Portable graymap	Binary
P6	Portable pixmap	Binary

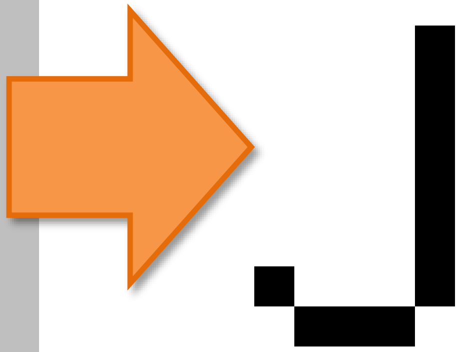


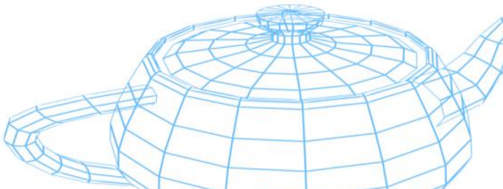
PBM

- ASCII .pbm file example:

```
P1
# This is an example bitmap of the letter "J"
6 10
0 0 0 0 1 0
0 0 0 0 1 0
0 0 0 0 1 0
0 0 0 0 1 0
0 0 0 0 1 0
0 0 0 0 1 0
1 0 0 0 1 0
0 1 1 1 0 0
0 0 0 0 0 0
0 0 0 0 0 0
```

width, height



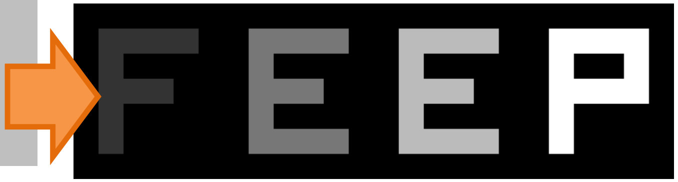


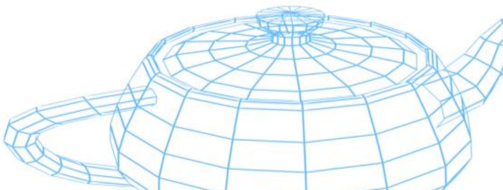
PGM

- ASCII .pgm file example:

```
P2
# Shows the word "FEEP"
24 7
9
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 2 2 2 2 0 0 4 4 4 4 0 0 7 7 7 7 0 0 9 9 9 9 0
0 2 0 0 0 0 0 4 0 0 0 0 0 7 0 0 0 0 0 9 0 0 9 0
0 2 2 2 0 0 0 4 4 4 0 0 0 7 7 7 0 0 0 9 9 9 9 0
0 2 0 0 0 0 0 4 0 0 0 0 0 7 0 0 0 0 0 9 0 0 0 0
0 2 0 0 0 0 0 4 4 4 4 0 0 7 7 7 7 0 0 9 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

max range

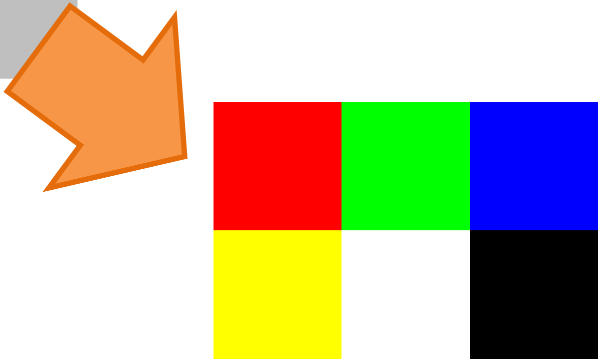


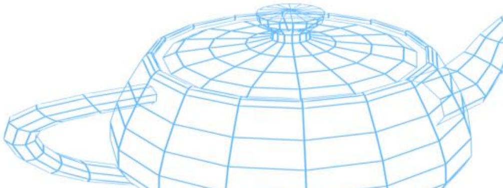


PPM

- ASCII .ppm file example:

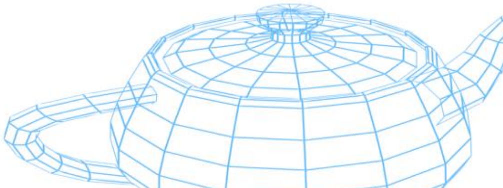
```
P3
3 2
255
255   0       0     0      255   0      0   0   255
255   255   0     255 255   255  0   0   0
```





TGA

- Introduced by Truevision Inc. (now part of Avid Technology).
- **T**ruevision [**A**dvanced **R**aster] **G**raphics **A**dapter (TGA or T[AR]GA).
- Originally, it was the native file format for the first graphics adapter for PC capable of high- and true-color graphics (16 and 24 bits per pixel).
- TGA supports 8, 15, 16, 24 and 32 bits per pixel (32 bit = 24 bit for RGB + 8 bit for alpha):
  - The 32 bit format is useful for its alpha channel to store transparent textures.



TGA

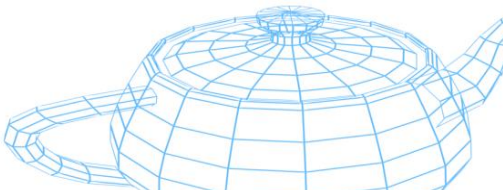
- Easy format to use.

- Each file is basically made of raw data (unless it is compressed) preceded by a compact header and an optional color map (palette) definition:
  - Header (18 bytes).
  - Image ID data (optional additional field for storing information such as image serial number, date, etc.).
  - Color map data (optional, palette).
  - Image data (raw or compressed).

- Current version is 2.0 (1989):
  - Version 1.0 back-compatible.

- Data is stored in little-endian.

WARNING



TGA header

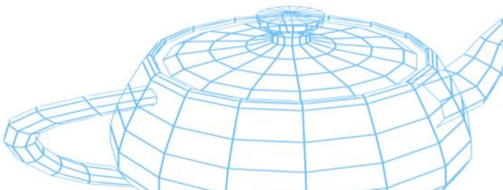
Field	Size	Description
ID length	BYTE	Length of the image ID field (zero if not used)
Color map type	BYTE	1 if included, 0 otherwise
Data type	BYTE	Code specifying the type of data used: 0 = no image data 1 = uncompressed color-mapped image 2 = uncompressed true-color image 3 = uncompressed black-and-white (grayscale) image 9 = run-length encoded color-mapped image 10 = run-length encoded true-color image 11 = run-length encoded black-and-white (grayscale) image
Color map origin	WORD	Color map starting offset
Color map length	WORD	Number of elements in the color map
Color map depth	BYTE	Number of bits used for each element

…

TGA header

…

Field	Size	Description
Origin X	WORD	Lower-left absolute X origin coordinate
Origin Y	WORD	Lower-left absolute Y origin coordinate
Width	WORD	Image width in pixels
Height	WORD	Image height in pixels
Color depth	BYTE	Number of bits per pixel
Image descriptor	BYTE	bits 3-0 give the alpha channel depth, bits 5-4 give direction



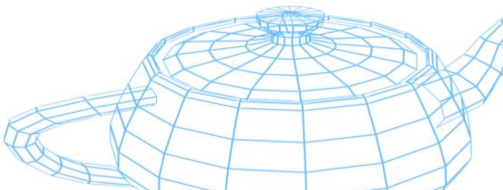
TGA optional sections

(only if *ID length* > 0)

Field	Size	Description
Image ID content	*ID length* bytes	Additional information such as image serial number, date, etc.

(only if *Color map type* = 1, repeated *color map length* times)

Field	Size	Description
Color map element	*Color map depth* bits	Color description for each element of the color map



TGA image data

(if uncompressed [*data type = 1, 2, 3*])

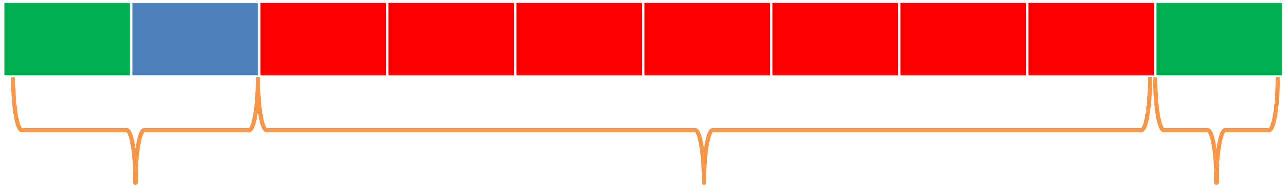
Field	Size	Description
Image data	*width * height * color depth* bytes	Uncompressed data is stored *as is*, like a memory dump. If a color map is used, each entry in the image data must be replaced by the color map element it is referring to.

(if compressed [*data type = 9, 10, 11*])

Field	Size	Description
Image data	must be processed	Data is stored using a simple compression algorithm called run-length encoding: if the first byte is 0, the next byte tells how many different pixels are specified starting from the next-next byte. If the first byte is 128, the second byte tells how many identical pixels will follow looking like the pixel specified starting from the next-next byte.

Run length encoding

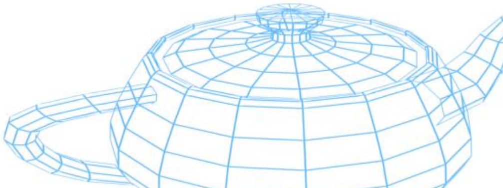
- Simple compression algorithm.
- If an image row contains N identical pixels in sequence, just store the pixel color and the number of times it is repeated.



Chunk ID: 0
Length: 2
Pixel: 0 255 0
Pixel: 0 0 255

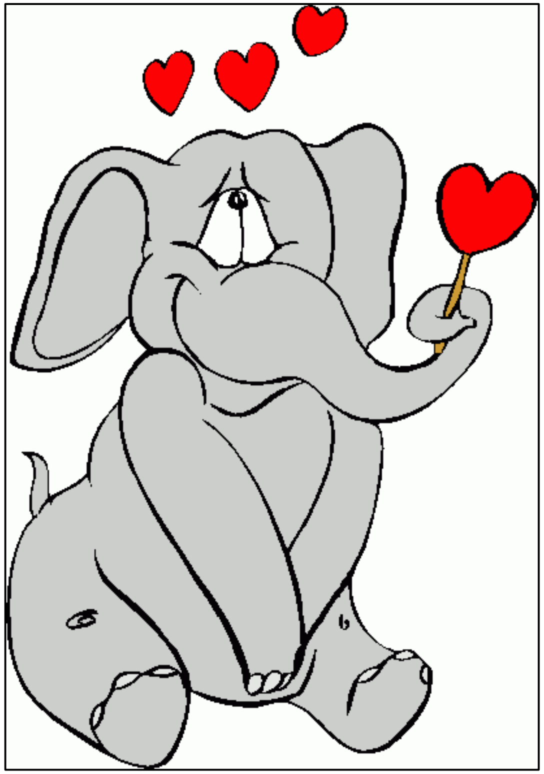
Chunk ID: 128
Length: 7
Pixel: 255 0 0

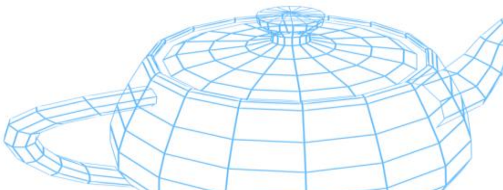
Chunk ID: 0
Length: 1
Pixel: 0 255 0



Run length encoding

- Works well for cartoon-like, manually painted images (e.g., pixel art).
- Very inefficient for real images (photos, scans) and for pictures using many gradients.





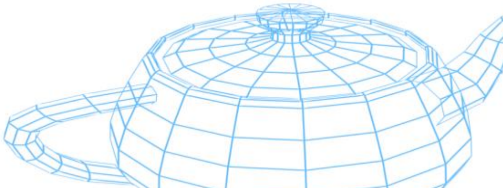
BMP

- **B**it**M**a**P** image file.

- Format introduced by Microsoft and widely used under Windows and other platforms.

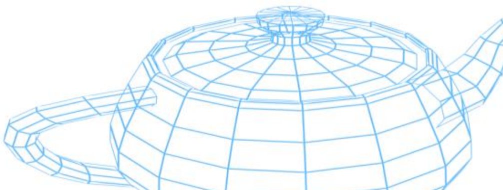
- Patent-free, well documented, natively supported by the Windows API.

- Relatively simple to use:
  - Not very different from TGA.



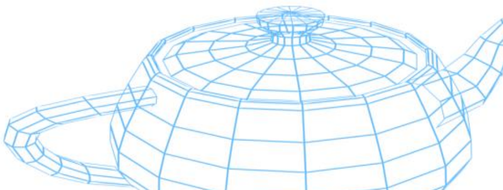
BMP

- Each BMP is structured into four sections:
    - File header.
    - Image header.
    - Image palette (optional).
    - Image data (can be compressed or uncompressed).



BMP file header

Field	Size	Description
Image file type	WORD	Image file type identifier. Should be ‘BM’ (or (0x424D) for a bitmap
File size	DWORD	File size in bytes
Reserved	WORD	[Reserved, always zero]
Reserved	WORD	[Reserved, always zero]
Image data offset	DWORD	Offset of the image data section in bytes



BMP image header

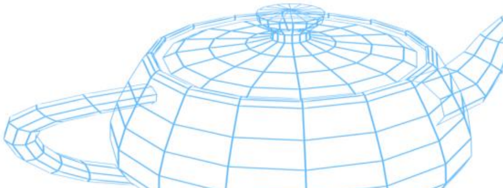
Field	Size	Description
Header size	DWORD	Header size in bytes
Image width	LONG	Image width in pixels
Image height	LONG	Image height in pixels
Number of image planes	WORD	Number of image planes (always 1)
Bits per Pixel	WORD	Color depth in bits (can be 1, 4, 8, 16, 24 or 32)
Compression method	DWORD	Compression method used: 0 = no compression 1 = Run length encoding (RLE8) 2 = Run length encoding (RLE4) 3 = Bitfields

…

BMP image header

…

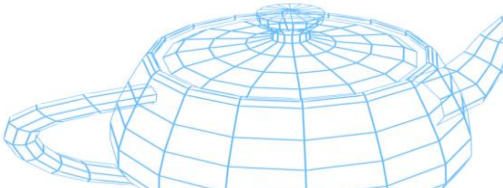
Field	Size	Description
Size of bitmap	DWORD	Bitmap size in bytes (can be 0 if not compressed)
Horizontal resolution	DWORD	Pixel per meter
Vertical resolution	DWORD	Pixel per meter
Number of used colors	DWORD	Number of used colors (if 0, it is equal to the 1 << BitsPerPixel)
Number of significant colors	DWORD	Number of important color (if 0, all the colors are important)



BMP palette

- When 1, 4, or 8 bits per pixel are used, the palette is a list of 2, 16, or 256 entries as follows:

Field	Size	Description
B	BYTE	Blue color component
G	BYTE	Green color component
R	BYTE	Red color component
Reserved	BYTE	[Reserved, always zero]

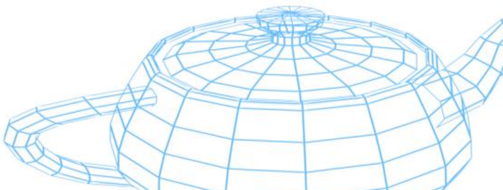


BMP palette

- When 16 or 24 bits per pixel are used and compression is set to BIT_FIELDS:

Field	Size	Description
B	BYTE	Blue color component
G	BYTE	Green color component
R	BYTE	Red color component
Reserved	BYTE	[Reserved, always zero]

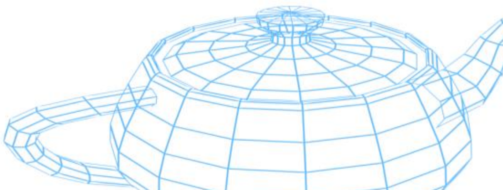
- When supported by the OpenGL context, you can use **`GL_BGR`** and **`GL_BGRA`** (or **`GL_BGR_EXT`** and **`GL_BGRA_EXT`**) to avoid inverting bytes order when passing data to **`glTexImage2D()`**.



BMP image data

- When the palette is specified:

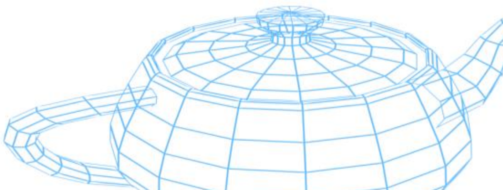
Bits per pixel	Description
1	1 byte -> 8 pixels
4	1 byte -> 2 pixels
8	1 byte -> 1 pixel
16	1 word -> 1 pixel
32	1 dword -> 1 pixel



BMP image data

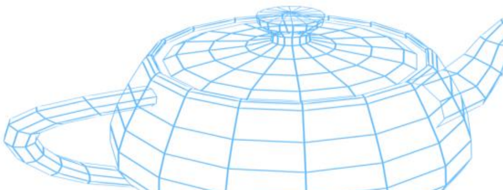
- Without palette:

Bits per pixel	Description
16	5 bits for each component (1 bit not used)
24	1 byte per channel (BGR order)
32	1 byte per channel (BGR order), 4th byte not used



BMP

- Additional properties:
  - Each line of non-compressed images is a multiple of 4.
    - When needed, an extra empty byte is added to align the line.
  - If *ImageHeight* > 0, image data begins from the bottom left corner; if < 0, from the upper left corner.



JPEG

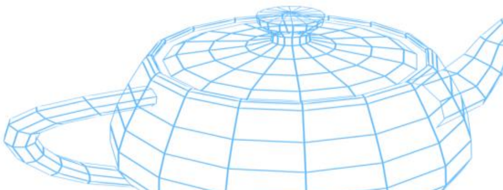
- **J**oint **P**hotographic **E**xpert **G**roup.

- Lossy compression format:
  - Some data is modified or lost to improve compression:

    decompressed image != original image

    - As it happens with MP3, MPEG, OGG, …

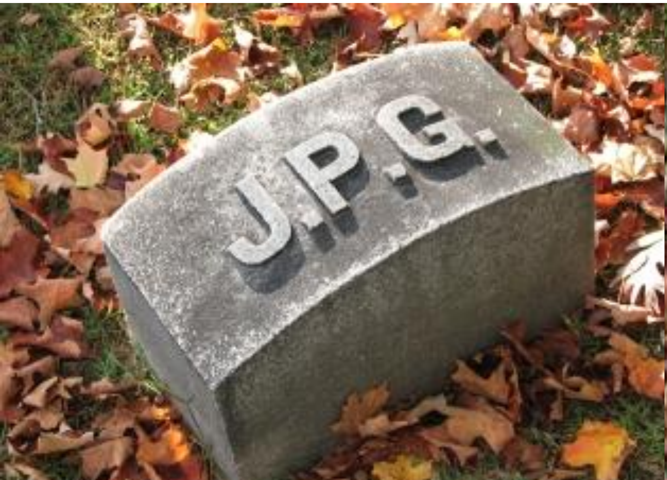
- Uses a series of complex algorithms to compress the bitmap, since the human eye is less sensitive to specific light components (e.g., high frequencies).



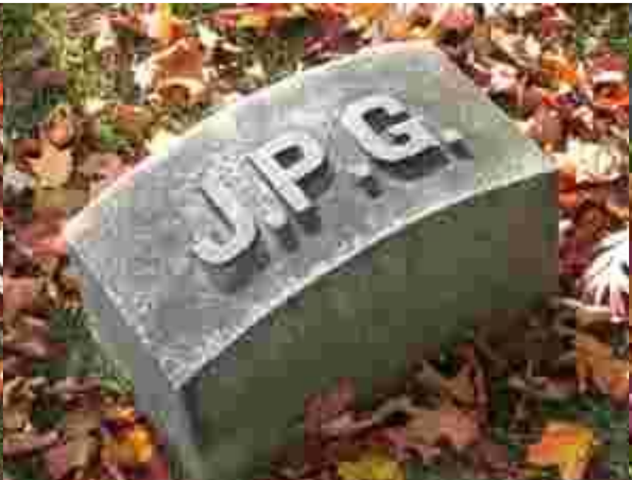
JPEG

- You should avoid/limit the usage of JPEG as the file format for your textures:
  - Lossy compression is optimized for static images seen from a specific point of view.
  - Do not use JPEG for frequently modified files but only for the final output.
  - Texture interaction with filtering, interpolation, lighting model, etc., can lead to unwanted results.

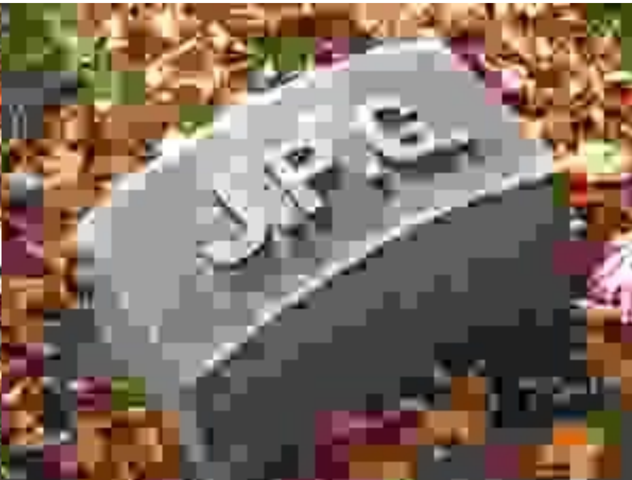
Quality = 100



Quality = 10



Quality = 1



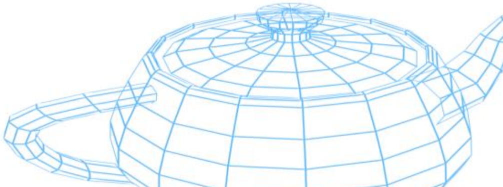
SVG file format

- **S**calable **V**ector **G**raphics.

- Open-standard introduced by the WWW Consortium in 1999.

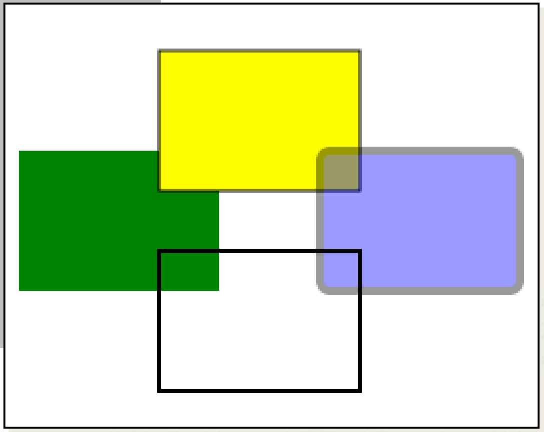
- XML-based:
  - Can be edited as text files.
  - Can be searched/parsed for specific strings/values.
  - Can be compressed (as any other text file).

- Modern Web browser can render them.



SVG example

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no" ?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 20010904//EN"
 "http://www.w3.org/TR/2001/REC-SVG-20010904/DTD/svg10.dtd">

<svg width="400" height="400" xmlns="http://www.w3.org/2000/svg">
   <title>Quads</title>
   <desc>Four quads with different attributes</desc>

   <rect x="50"  y="150" width="100" height="70"  fill="green"/>
   <rect x="120" y="100" width="100" height="70" fill="yellow"
   stroke="black" />
   <rect x="200" y="150" width="100" height="70" rx="5"
   fill="blue" opacity="0.4" stroke="black" stroke-width="4"/>
   <rect x="120" y="200" width="100" height="70" fill="none"
   stroke="black" stroke-width="2"/>
</svg>
```



SVG example

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no" ?>
```

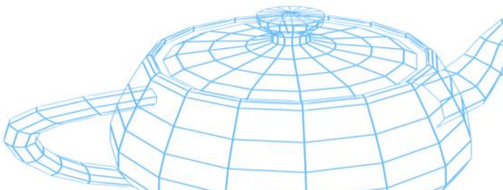
Specify the version and encoding used for this XML file

```
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 20010904//EN"
 "http://www.w3.org/TR/2001/REC-SVG-20010904/DTD/svg10.dtd">
```

Specify that we are using a SVG file using the Document Type Definition declared in file svg10.dtd

```
<svg width="400" height="400" xmlns="http://www.w3.org/2000/svg">
```

Image width, height and namespace used



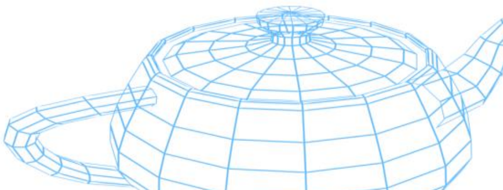
SVG example

```
<title>Quads</title>
<desc>Four quads with different attributes</desc>
```

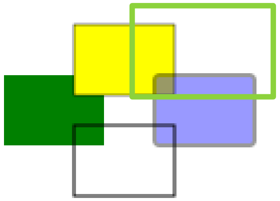
Additional details not used for rendering but useful for parsing the SVG

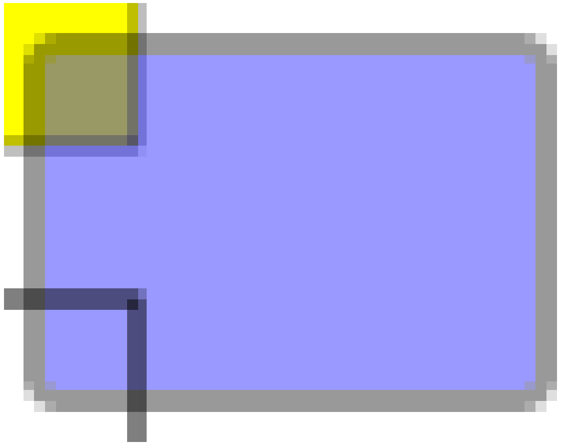
```
<rect x="50"  y="150" width="100" height="70"  fill="green"/>
<rect x="120" y="100" width="100" height="70" fill="yellow"
  stroke="black" />
<rect x="200" y="150" width="100" height="70" rx="5" fill="blue"
  opacity="0.4" stroke="black" stroke-width="4"/>
<rect x="120" y="200" width="100" height="70" fill="none"
  stroke="black" stroke-width="2"/>
```

Defines four quads with various sizes, positions and attributes (other primitives such as *ellipse*, *line*, *circle*, etc. also exist)

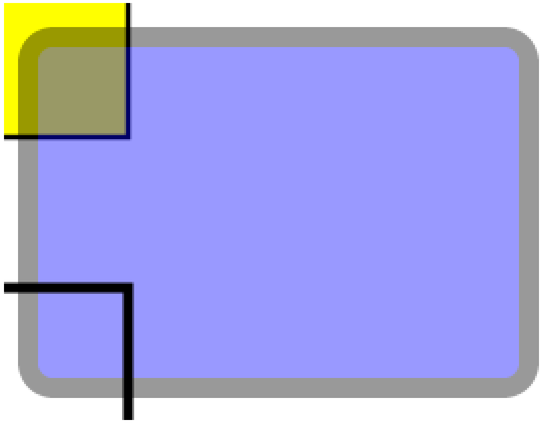


SVG example

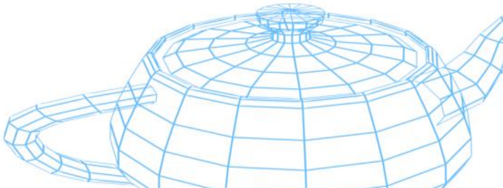




Raster



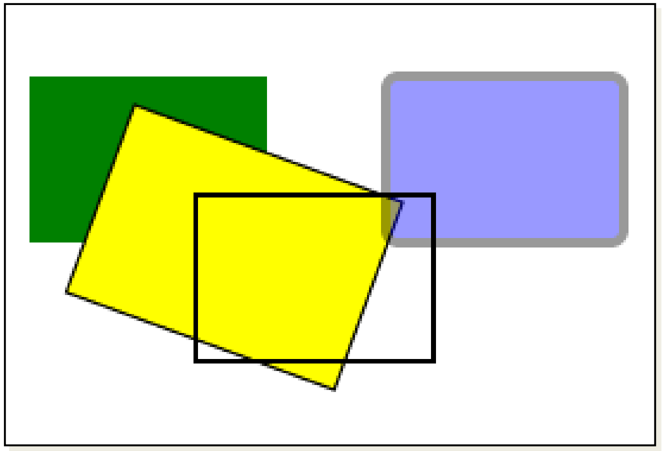
Vector



SVG

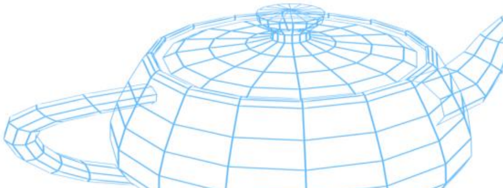
- SVG also supports transformations:

```
<rect x="120" y="100" width="100" height="70"
  fill="yellow" stroke="black" transform="scale(1.2) rotate(20)" />
```



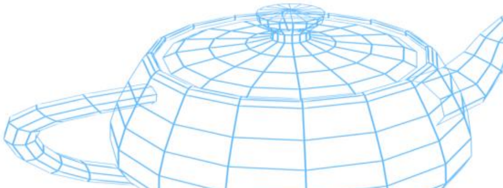
SVG

- Instead of creating SVG files manually, use an editor (e.g., Inkscape).
- SVG syntax and attributes are like the ones used in CSS files.



PDF

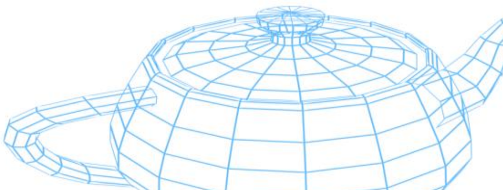
- **P**ortable **D**ocument **F**ormat.
- Introduced by Adobe Systems in 1993.
- Open, license-free standard since 2008.



PDF

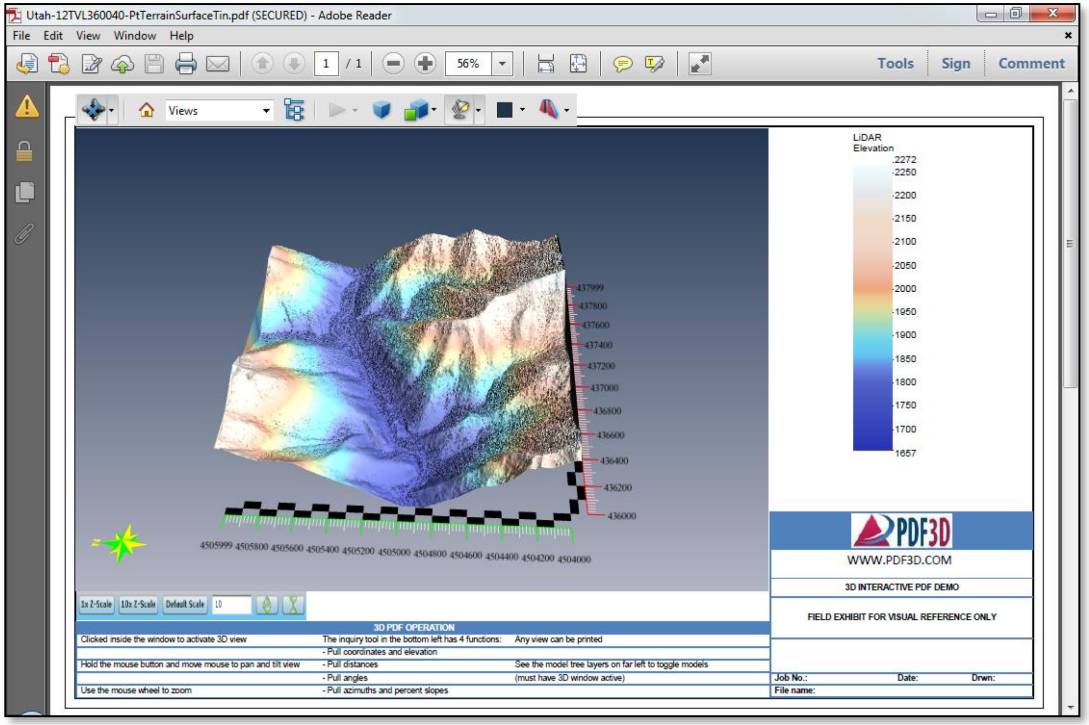
- A PDF file is a portable, hardware-independent container of objects used to define a document with a static layout:
  - Fonts, images, text, etc.

- Images can be both raster or vector graphics:
  - Vector graphics are defined in a PostScript-like way.
  - Raster graphics are stored using various internal filters that are similar to the compression techniques used in other file formats (e.g., RLE, JPEG2000, GZIP, …).



PDF

- PDF files can also contain 3D objects:
  - The PDF viewer embeds a 3D model viewer.



WWW.PDF3D.COM

3D INTERACTIVE PDF DEMO

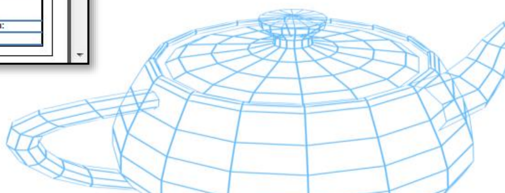
FIELD EXHIBIT FOR VISUAL REFERENCE ONLY

Job No.: Date: Drwn:

File name:

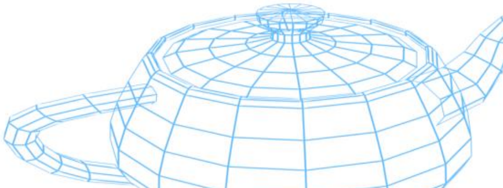
3D PDF OPERATION

Clicked inside the window to activate 3D view	The inquiry tool in the bottom left has 4 functions:	Any view can be printed
	- Pull coordinates and elevation	
Hold the mouse button and move mouse to pan and tilt view	- Pull distances	See the model tree layers on far left to toggle models
	- Pull angles	(must have 3D window active)
Use the mouse wheel to zoom	- Pull azimuths and percent slopes	



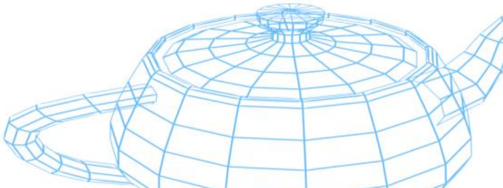
PDF

- PDF files are not often used in 3D graphics but are more suitable for generating portable, device-independent documents (e.g., using Microsoft Word, LibreOffice, Latex, etc.) and for printing.



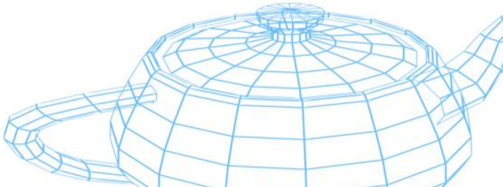
Documentation

- Prof. Spinedi’s class notes.
- http://www.fileformat.info/ (updated often).
- http://www.wotsit.org/ (it was good, but now it has a lot of broken links…).
- James D. Murray, William van Ryper, *Encyclopedia of Graphics File Formats*, 2nd Edition, May 1996, ISBN: 1-56592-161-5



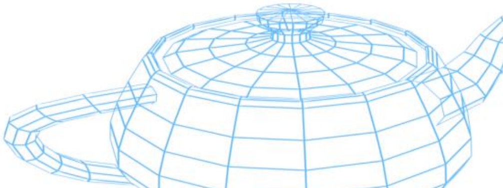
Libraries

- Encoding/decoding graphics file formats can be a time-consuming and complex task:
  - Use external libraries when possible (libJPEG, libPNG, etc.).
  - Higher-level libraries also exist (FreeImage, DevIL, SOIL, etc.):
    - Multiple file formats supported.
    - Additional functions provided (color depth conversion, resizing, etc.).



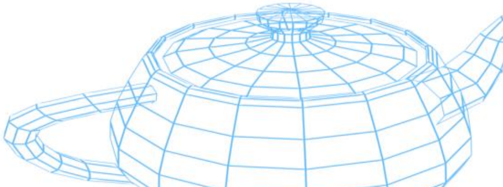
FreeImage

- Open-source library supporting popular graphics image formats like PNG, BMP, JPEG, TIFF and more.
- Easy to use, fast, thread-safe, and cross-platform.
- Available at: http://freeimage.sourceforge.net/



FreeImage

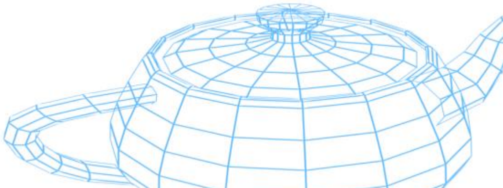
```
#include <FreeImage.h>

// Init FreeImage:
FreeImage_Initialise(); // Static lib only, also #define FREEIMAGE_LIB

// Load an image from file:
FIBITMAP *bitmap = FreeImage_Load(FreeImage_GetFileType("teapot.tga", 0),
                                  "teapot.tga");

// Load image into OpenGL:
glTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA,
             FreeImage_GetWidth(bitmap), FreeImage_GetHeight(bitmap),
             0, GL_BGRA_EXT, GL_UNSIGNED_BYTE, // FreeImage uses BGR
             (void*) FreeImage_GetBits(bitmap));

// Release bitmap and FreeImage:
FreeImage_Unload(bitmap);
FreeImage_DeInitialise(); // Static lib only
```



FreeImage

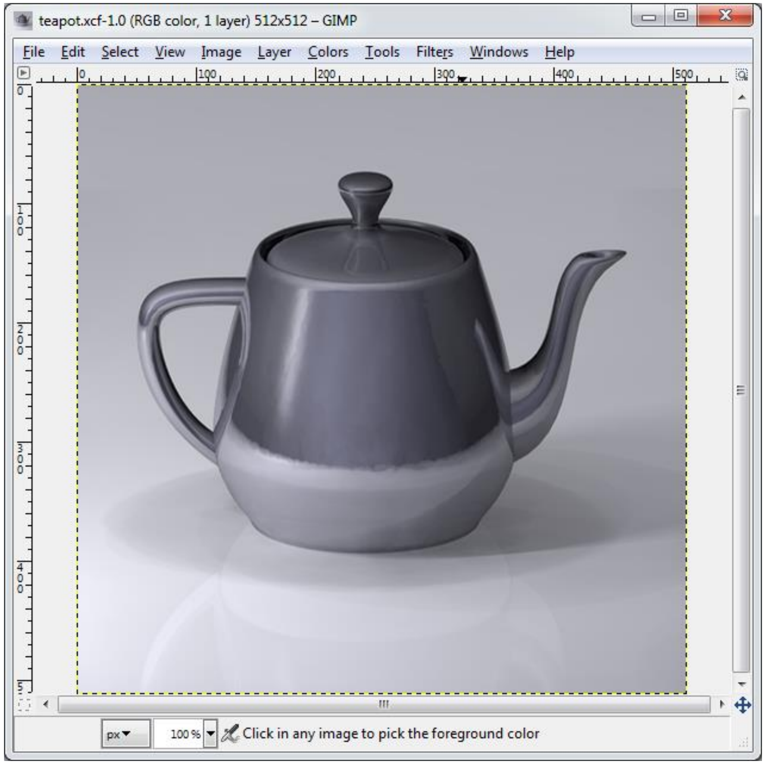
- On Windows:
  - Download and compile the solution.
  - Make sure to change the *Runtime library* flag of **all the projects** in the solution to “Multi-thread [debug] DLL” (under project properties -> Code generation).
  - If you want to use the **static version**, define **`FREEIMAGE_LIB`** in your code or project properties:
    - The series solutions will only use the static lib.

- On Ubuntu:
  -
    ```
    sudo apt install libfreeimage-dev
    ```



GIMP

- **G**NU **I**mage **M**anipulation **P**rogram.
- Free Photoshop-like software:
  - Multiplatform.
  - Supports several file formats.
- Useful for converting/resizing images to be used later as textures.
- http://www.gimp.org/



teapot.xcf-1.0 (RGB color, 1 layer) 512x512 – GIMP

Click in any image to pick the foreground color