Generations of Consumer Computer Graphics as Seen in Demos

Markku Reunanen, Aalto University

Outline

- Consumer computer graphics
- Different generations with examples
- Conclusion
- Further reading

The microcomputer era

- First affordable home computers in the late 1970s
  - Apple II, TRS-80, Atari 400/800
- Home computer fever of the early 1980s
- Concurrent rise of consumer electronics
  - Pocket calculators
  - VCRs
  - Electronic and video games

Consumer computer graphics

- Vast developments from late 1970s to today
- We may observe “generations”
  - Increasing computing power
  - Increasing graphical capabilities
  - Different technical approaches
  - Parallel continuums, not discrete steps
- Here I approach CG through demos
  - Why not games?

The generations

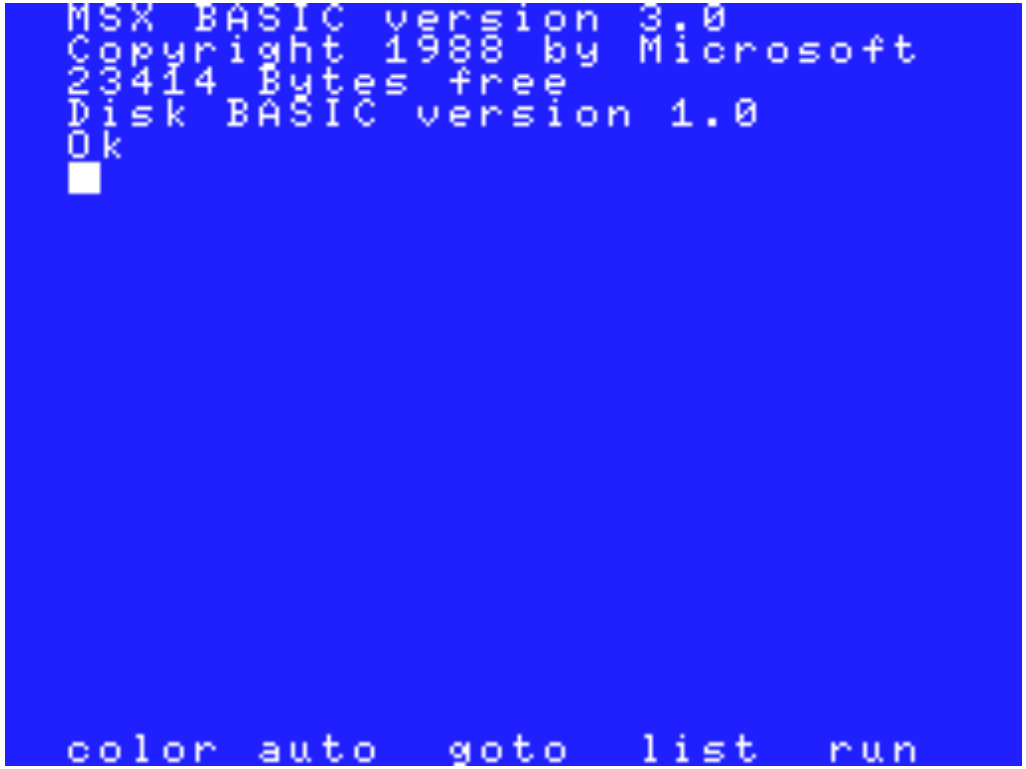
- Character graphics
- Bitmap graphics
- Some hybrids
- Chunky
- Fixed-function pipeline
- Shaders

Character graphics

Character graphics

- Typical of 8-bit computers
  - What else would you *do* than show text?
- The dominant paradigm until the mid-1980s
- Well suited for small memory
- Not all character graphics come equal
  - ROM character sets
  - User-definable fonts, effectively yielding free-form graphics
  - Commodore VIC-20 (1980), C-64 (1982), MSX (1983)

Character graphics



Typical resolutions: 32x24 and 40x25 characters of 8x8 pixels (256x192, 320x200)

Character graphics

- Fast updates:
  - Change one character, updated everywhere
  - Change all screen content with little bandwidth
- Suitable for game level blocks
- Individual pixels tedious to access
- Let’s see some examples!

Bitmap graphics

- Consist of one or more *bitplanes*
- Typical of the “16-bit generation”
  - Commodore Amiga (1985)
  - Atari ST (1984)
  - Most IBM PC graphics modes (EGA/VGA, 1984–)
- 2^bitplanes = number of colors
- User-definable *palette* as opposed to fixed colors

Bitmap graphics

Plane 0: 000010001 00001000 10001000 ...
Plane 1: 100001100 01111100 00100010 ...
Plane 2: 111000100 00010000 11100000 ...

2^3= eight possible colors

First pixel: 110, color number 6

Bitmap graphics

- Hard to set an individual pixel to a certain color
  - Need to touch multiple bitplanes, even eight
  - Need to fiddle with individual bits
- Notable strengths, too
  - An individual bitplane can be redrawn fast
  - Bitplanes are independent – transparent and translucent layers easy to do
- And then examples

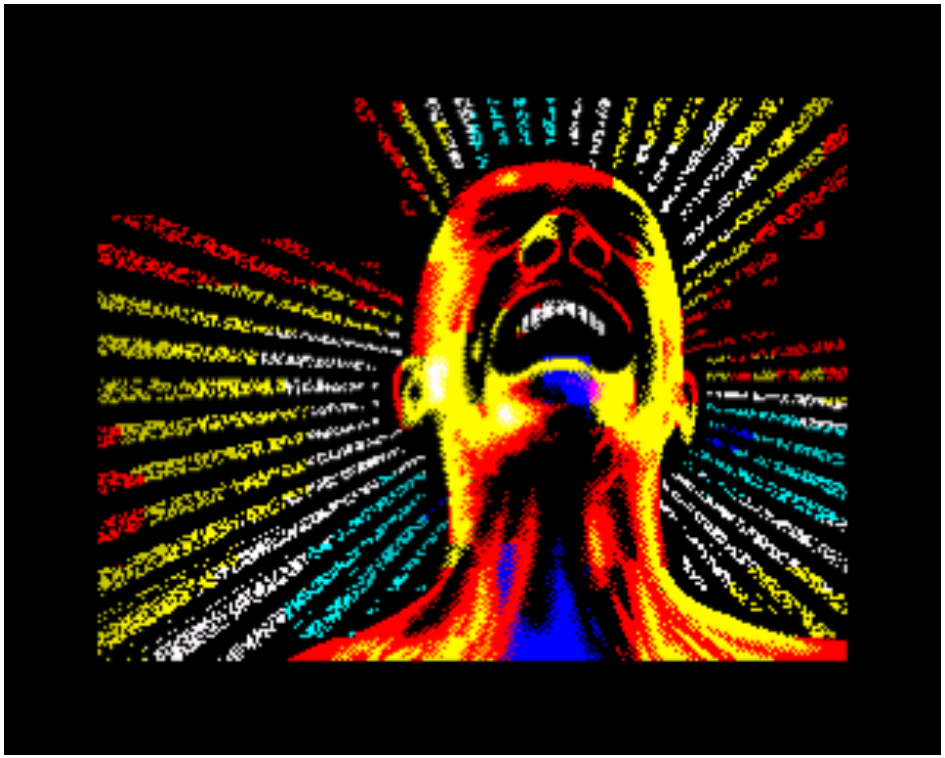
Bitmap graphics

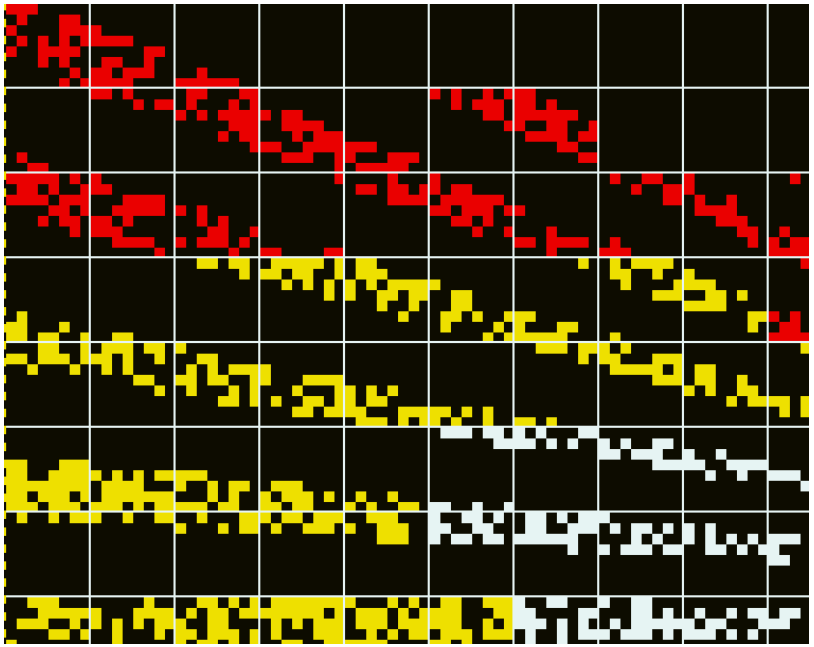
“Glenz vectors” by Edward Melia

Some hybrids

- Some 1980s' computers don't fit these categories
  - Sinclair ZX Spectrum a hybrid between bitmap and character mode
  - Oric-1 and Oric Atmos more like teletype
  - Amstrad CPC and Sinclair QL interleaved bitmaps in the memory
- Quite many machines featured *sprites* that could be moved around independently

Some hybrids





“Nightmares” by Noice (ZX Spectrum)

Some hybrids

Success crack intro showing sprites (C-64)

Chunky

- Also known as “packed pixel”
- Characteristic of the 1990s IBM PC demos
- First major use in VGA compatible cards (1987–)
  - Introduced in MCGA the same year
- On VGA 320x200 pixels with 256 individual colors
  - One byte in memory = one pixel
  - So-called *linear framebuffer*

Chunky

- Now individual pixels were simple to access
  - For example texture mapping easy to implement
  - Transparency/lucency still possible, even though relatively heavy
  - Various image deformations appeared
- Attempts to emulate chunky modes with character and bitmap graphics
  - Atari ST, Amiga, 8-bit computers

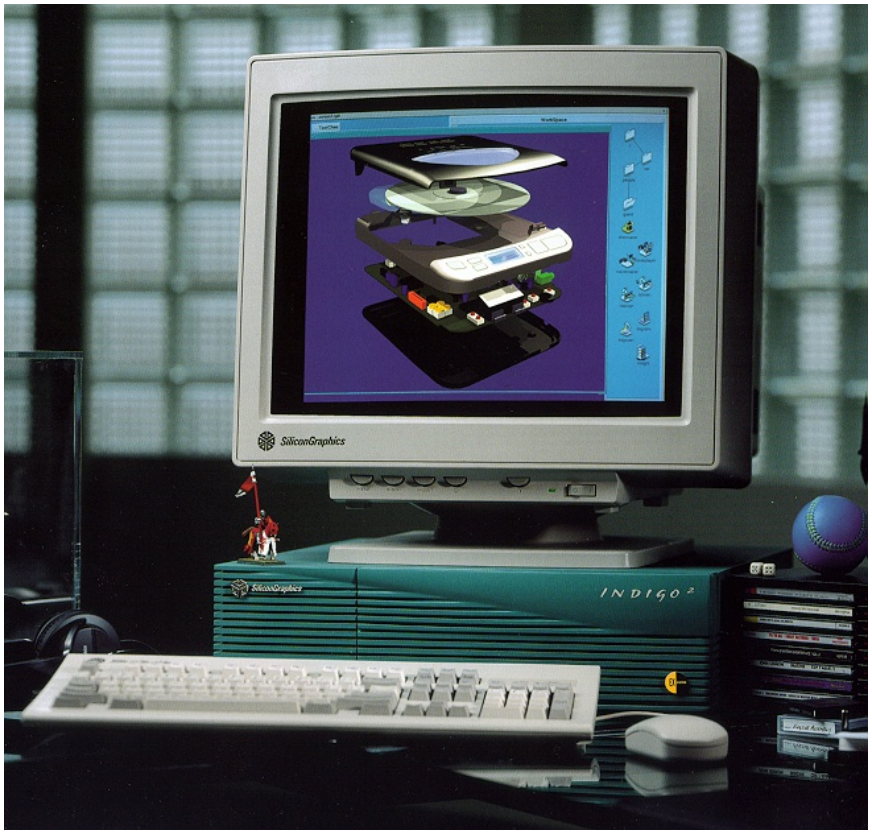
Chunky

- Finally, *true color* modes in the late 1990s
- 24 bits i.e. three consecutive bytes (R,G,B) per pixel
- 16.8 million possible colors!
  - Good color reproduction
  - Straightforward blending of images together
- Need for computing power
  - At this point, 486 and Pentium-class machines
- Let’s see some demos again...

Fixed-function pipeline

- Polygons instead of pixels
- The rise of 3D games
- 3D acceleration originally from expensive workstations, esp. Silicon Graphics
- First popularized by 3dfx Voodoo (1996–)
- ATI (AMD) and Nvidia took over soon

Fixed-function pipeline



SGI Indigo 2 (image: SGI Depot)

Fixed-function pipeline

- Everything drawn as polygons
  - Little control over individual pixels
  - Fast 3D graphics
  - Layers, lighting and texturing simple to achieve
- No more low-level access to pixels
  - Need to use programming libraries, such as *Glide*, *OpenGL* and *DirectX*
- Let’s see how it looked like

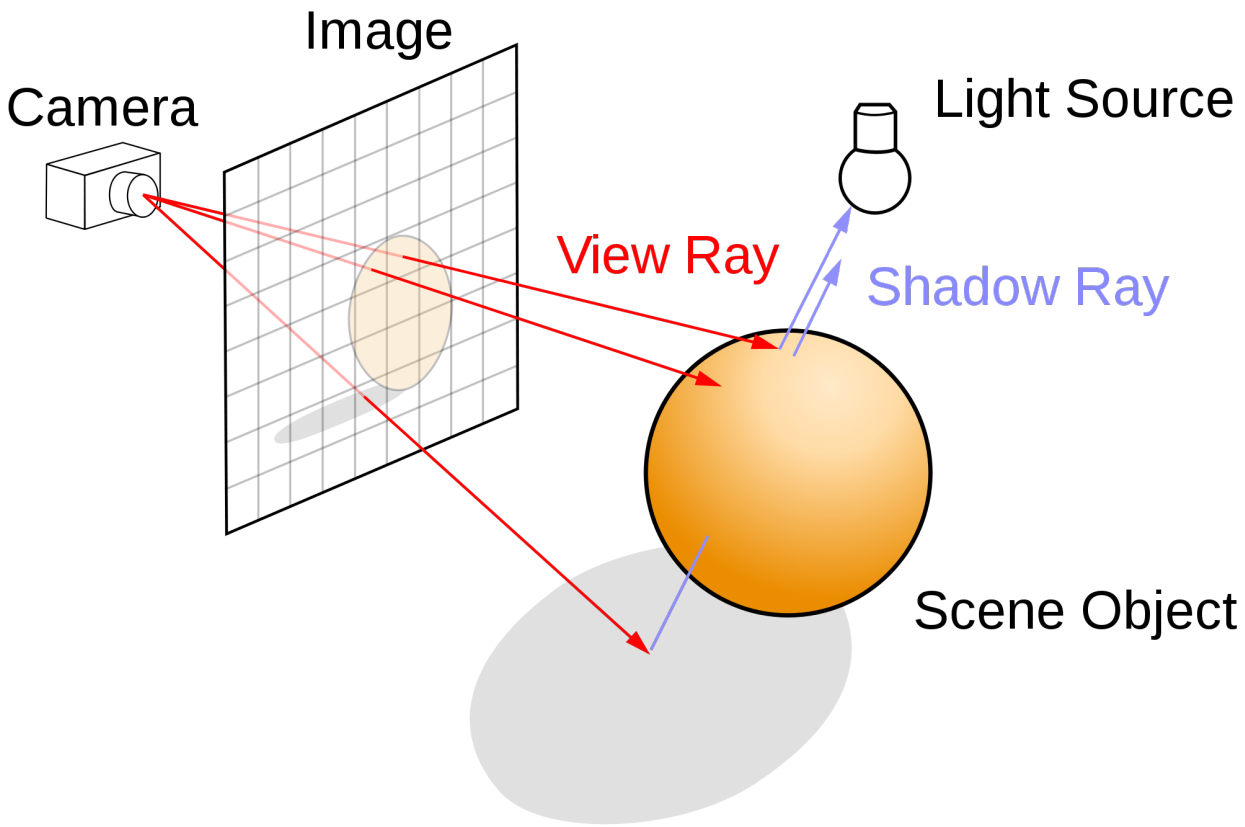
Shaders

- The current generation
- Programmability is back
- *Geometry*, *vertex* and most notably *pixel* shaders
  - Short pieces of code run fast in parallel
  - Pixel-level access is back
- Previously unseen computing power

Shaders

- Two major approaches:
  - Draw polygons like before and use shaders for surface materials, lighting and so on
  - Do calculations for each pixel on the screen
- The latter can be split to:
  - Raymarching (related to raycasting and -tracing)
  - Just come up with some cool-looking mathematical formula
- http://www.shadertoy.com/
- And then demos.

Shaders



Raytracing (image: Wikipedia)

Conclusion

- Thousand- or millionfold increases in computing power, bandwidth and colors
- Games often the driver
- The underlying platform and community preferences together dictate the outcome
- Demoscene’s hardware-pushing ethic
  - … but not just that
- Cultural adoption of technology – how a group of people has found use for computers

Further reading

- Boris Burger et al. (2002), *Realtime Visualization Methods in the Demoscene*
- Canan Hastik (2014), *Demo Age: New Views*
- Doreen Hartmann (2014), *Animation in the Demoscene. From Obfuscation to Category.*
- Markku Reunanen (2010), *Computer Demos – What Makes Them Tick*
- Markku Reunanen (2014), *Four Kilobyte Art*