

ICS370 Computer Graphics

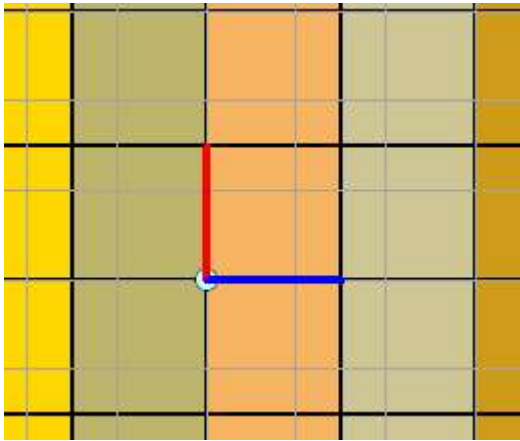
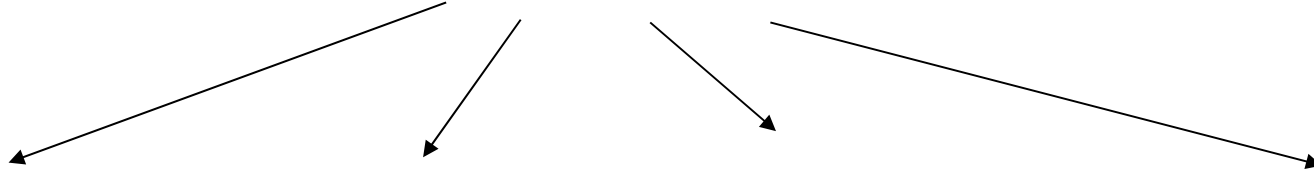
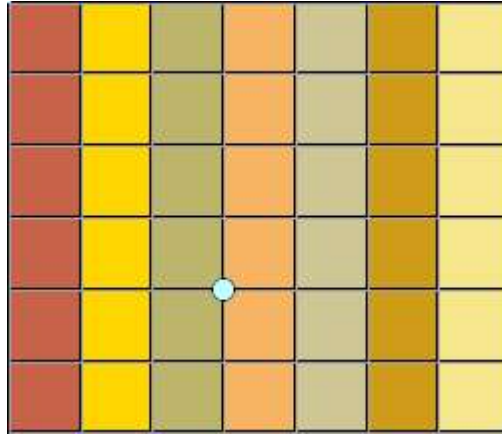
Instructor: Toby Daniel

Lecture 4

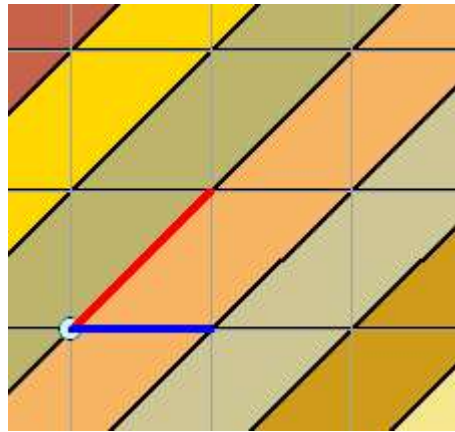
Affine Transforms

- An **Affine Transform** performs a linear mapping from 2D coordinates to other 2D coordinates.
- They have two very important properties:
 - Straight lines remain straight
 - Parallel lines remain parallel
- Affine transformations are translations, scales, flips, rotations, and shears.

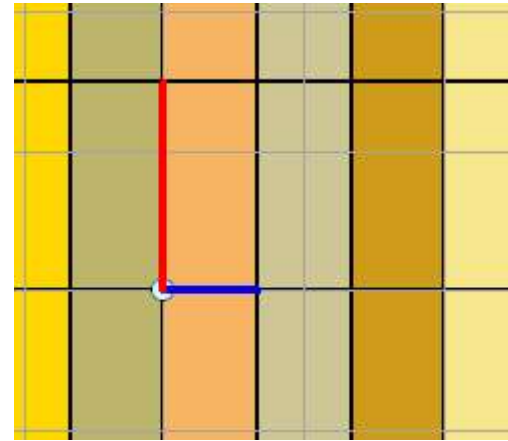
Original



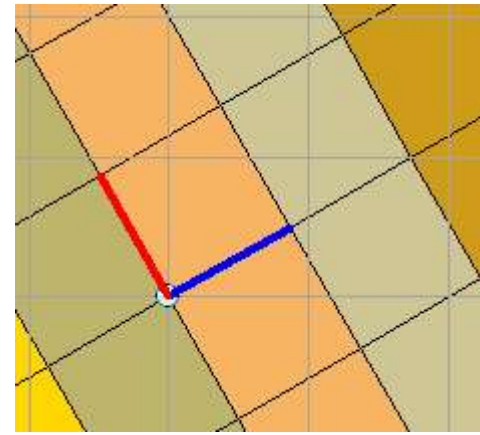
Scale



Shear



Aspect ratio



Rotate

Affine Transforms

- are **linear transforms**, so the transformation can be expressed in the matrix notation of linear algebra.
- an arbitrary Affine Transform can be mathematically expressed by six numbers in a matrix like this:

$$\begin{bmatrix} sx & shx & tx \\ shy & sy & ty \end{bmatrix}$$

- sx and sy are factors that accomplish scaling.
- tx and ty are factors that accomplish translation.
- shx and shy are factors that accomplish shear.

Computing New Coordinates

$$\text{newX} = \text{sx} * \text{x} + \text{shx} * \text{y} + \text{tx}$$

$$\text{newY} = \text{shy} * \text{x} + \text{sy} * \text{y} + \text{ty}$$

sx sy - scaling.

tx ty - translation.

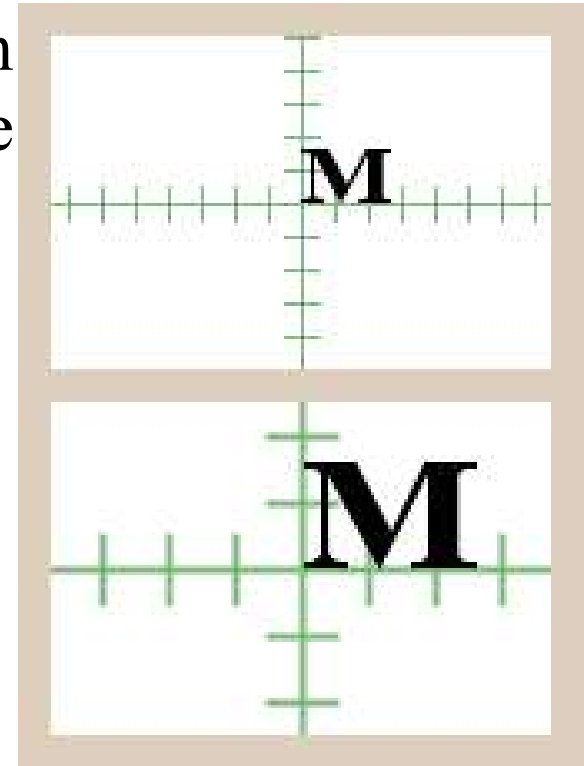
shx shy - shear.

Scaling

- A scaling transformation alters the size of an object. This operation can be carried out by multiplying the coordinate values (x, y) of each vertex by scaling factors s_x and s_y to produce the transformed coordinates (x', y')

$$x' = x \cdot s_x \quad y' = y \cdot s_y$$

- s_x is a numeric multiplier that can be either positive or negative.
- Unequal values for s_x and s_y result in a differential scaling or *aspect ratio* changes.
- We can control the location of a scaled object by choosing a position, called the **fixed point**, that is to remain unchanged after the scaling transformation.



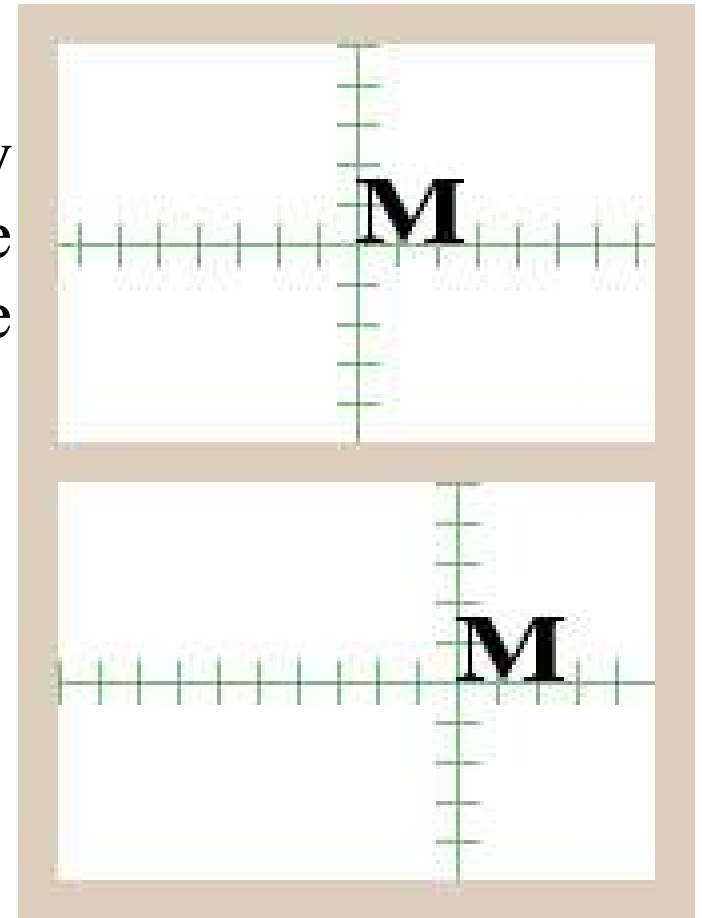
Translation

- A translation is applied to an object by repositioning it along a straight-line path from one coordinate location to another.
- We translate a two-dimensional point by adding translation distances t_x and t_y to the original coordinate position (x, y) to move the point to a new position.

$$\mathbf{x}' = \mathbf{x} + \mathbf{tx}$$

$$\mathbf{y}' = \mathbf{y} + \mathbf{ty}$$

- The translation distance pair (t_x, t_y) is called a *translation vector* or *shift vector*.
- Translation is a rigid-body transformation that moves objects without deformation.



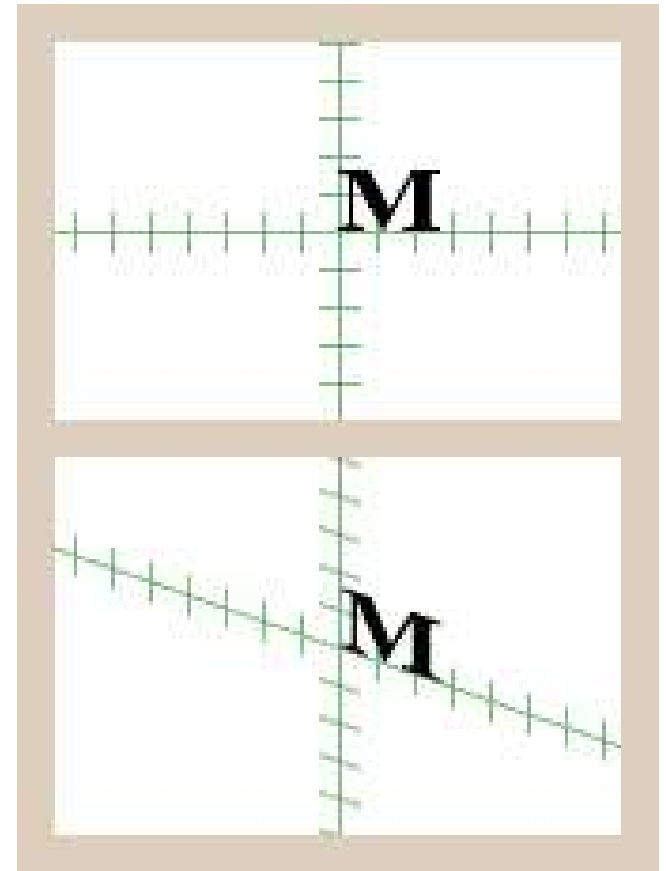
Translation

- A straight line is translated by applying the transformation to each of the line endpoints and redrawing the line between the new endpoint positions.
- Polygons are translated by adding the translation vector to the coordinate position of each vertex and regenerating the polygon using the new set of vertex coordinates
- For a circle or ellipse, we translate the center coordinates and redraw the figure in the new location.
- We translate other curves (for example, splines) by displacing the coordinate positions defining the objects, then we reconstruct the curve paths using the translated coordinate points.

Shear

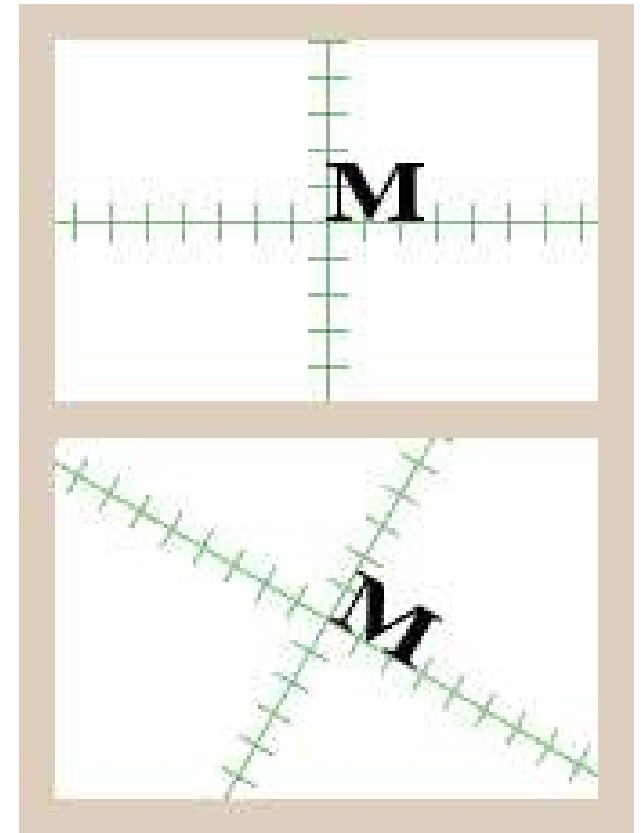
- Shear is basically a diagonal transformation of one axis of the coordinate system
- It has the effect of displacing the object as if it were made up of layers or sheets
- For a horizontal shear

$$x' = x + shx \cdot y \quad y' = y$$



Rotation

- In the process of transforming a graphic you can cause it to be rotated by a specified angle around a specified coordinate position.
- Some systems use degrees
- Some systems use radians
- $\text{PI radians} = 180 \text{ degrees}$
 - $\text{PI}/2 = 90 \text{ degrees}$
 - $\text{PI}/4 = 45 \text{ degrees}$
 - $\text{PI}/8 = 22.5 \text{ degrees}$



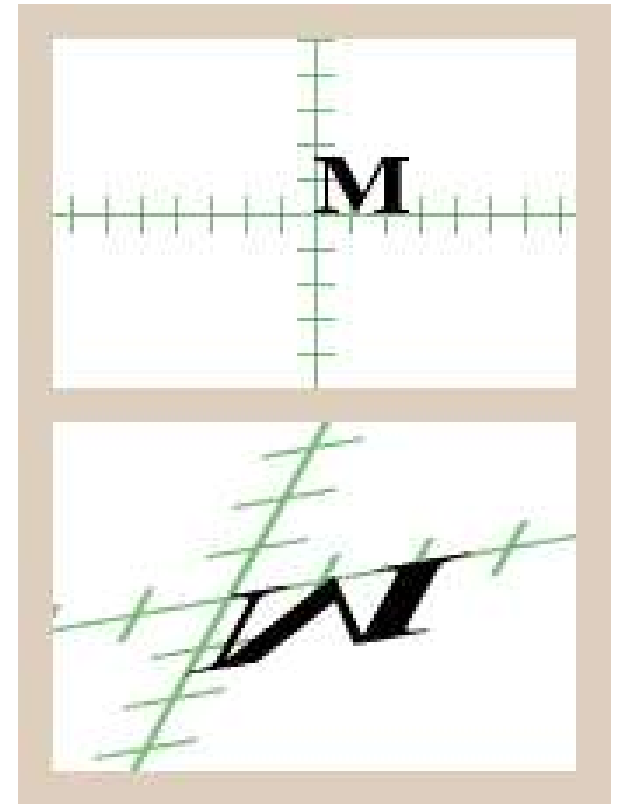
Rotation

- A two-dimensional rotation is applied to an object by repositioning it along a circular path in the xy plane. To generate a rotation, we specify a rotation angle θ and the position (xr yr) of the rotation point (or pivot point) about which the object is to be rotated.
 - Positive values for the rotation angle are anti-clockwise rotations
 - Negative values rotate objects in the clockwise direction.
- Rotations are rigid-body transformations that move objects without deformation. Every point on an object is rotated through the same angle.

Combination Transforms

Question time

- What combination of transforms have been applied to this image?



Another Question

- Does the order that transforms are performed make a difference to the final image?

Exercise

On paper draw a square with coordinates (1,1) (1,2) (2,1) (2,2)
then

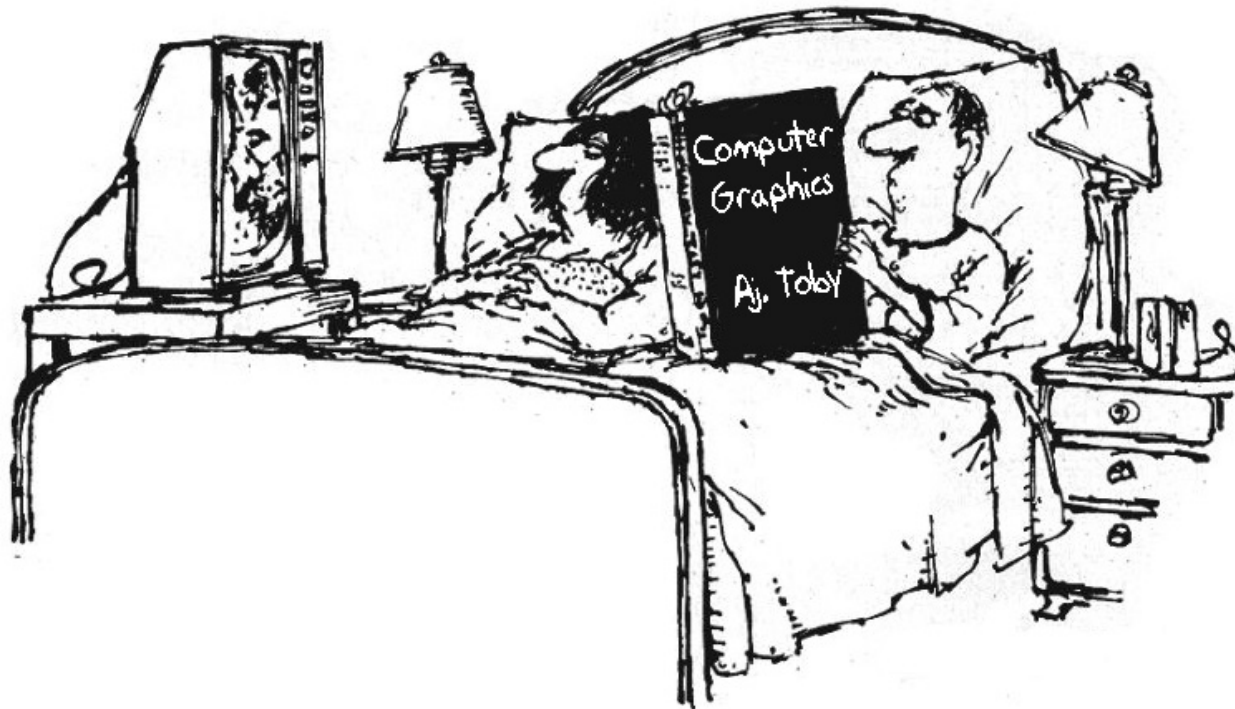
- a) rotate by +90 degrees with the pivot point as the origin
- b) translate by +1 on the x axis

Perform these again but in the opposite order (b first then a)

Take your original square and apply a horizontal shear with $shx = 1$

Homework - bedtime reading

Computer Graphics
Hearn & Baker
Chapter 5. Pages 183 - 208



Coordinate Spaces

- You have already seen how coordinate spaces work during Lab 2. The origin usually being at the top left hand corner.
- In computer graphics it is useful to have more than one coordinate space when drawing images.
- These are known as the **User Space** and the **Device Space**.

User Space

- The User Space refers to the coordinate space that is used by applications (not *users* as the name implies)
- This is a virtual space that is device independent. This means that it is not constrained by the screen size, resolution, paper type etc.
- User Space is where primitives are created and transformations performed with all operations using user space coordinates
- Before graphics can be drawn to your monitor or printed the coordinates in User Space are transformed into Device Space.

Device Space

- Coordinates in device space usually refer to individual device pixels and are aligned on the infinitely thin gaps between these pixels.
- A transform maps the user space coordinate system to screen or printer device coordinates such that the origin maps to the upper left hand corner of the target region of the device.
- So screen devices with 72 dpi would be set to approximately 72 user space coordinates per square inch

Rendering

- Rendering is drawing the final image onto the screen, including colour, fill, shading etc.
- The basic rendering mechanism is a system that controls **when** and **how** programs can draw.
- 2D rendering processes User Space coordinates to Device Space.
- 3D rendering processes 3D coordinates to a 2D view.

2D Rendering

The steps in the rendering process are:

- Determine what to render.
- Transform user space into device space.
- Determine colours.
- Create drawing on the destination device.

2D Rendering Operations

- Shape operations
- Text operations
- Image operations

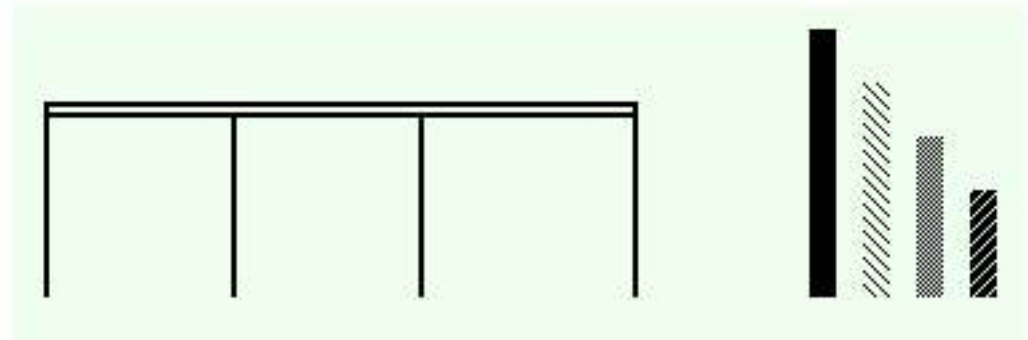
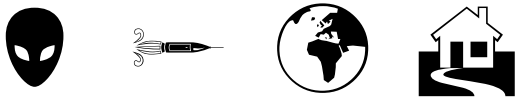
Blitting - BitBLT

- Bit blit is a computer graphics operation in which two bitmap patterns are combined into one.
- The name derives from the BitBLT machine instruction for the Xerox Alto computer, standing for "Bit Block Transfer".
- 'Blitting' was an important process in 2D graphics - especially games.
- Bit blit operations were key in the evolution of computer displays from using **character graphics**, to using **bitmap graphics** for everything.

Character Graphics

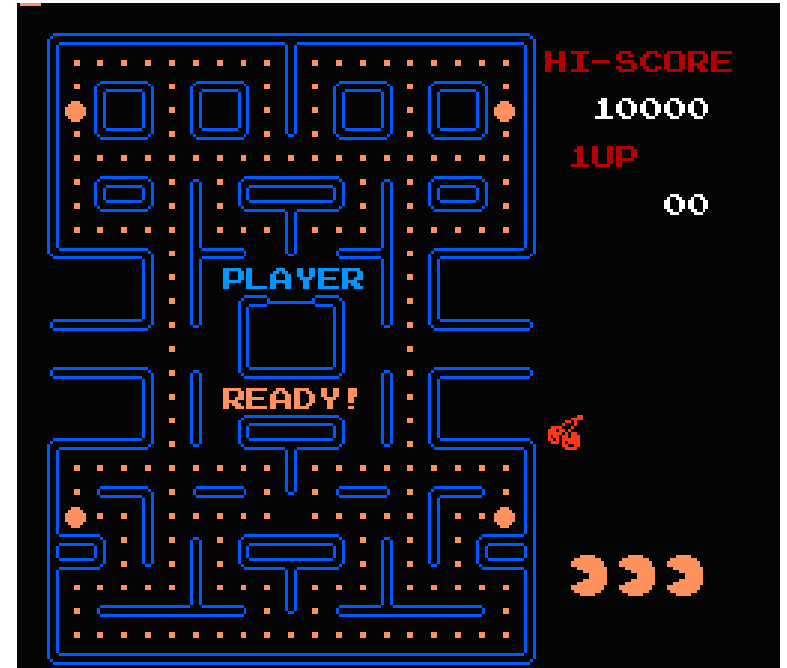
- A set of special symbols strung together like letters of the alphabet to create elementary graphics and forms
- Extended ASCII character sets in the original DOS font allowed elementary forms and bar charts to be printed

Font: Webdings



Use of Blitting

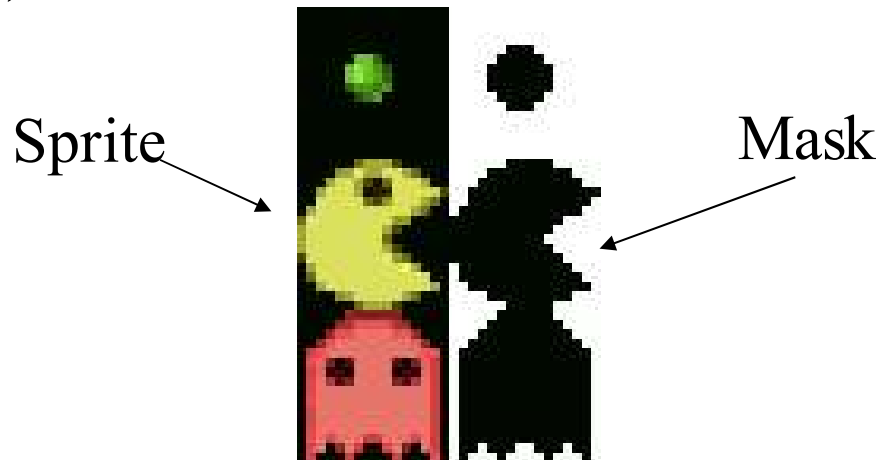
- A classic use for blitting is to render **sprites** onto a background image.



- A **sprite** is a two-dimensional image or animation that is integrated into a larger scene. Originally these were 16 x 16 pixels in size.
- Nowadays the term *Sprite* is used to describe flat images that are seamlessly integrated into complicated three-dimensional scenes.

How Blitting Works

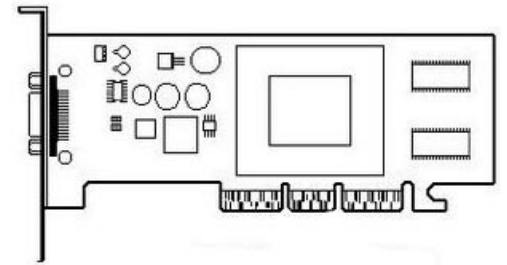
1. First a background image is displayed
2. Then a mask for the sprite is blitted onto the background to create black 'holes' where the sprites will appear (a AND operation).
3. Lastly the sprite is blitted onto the background in the black spaces created in step 2. (a OR operation)



Graphics Hardware

- The Blitter!
- A Blitter is a co-processor chip dedicated to memory data transfers, usually independently of the CPU using bitblit methods.
- Previously the computer's CPU was given the job of moving bitmaps around in memory.
- The Amiga was one of the first commercial PCs to have a Blitter co-processor.
- And so started the evolution in *CPU independent* graphics hardware.

A Brief History



1980's

- Amiga was the first mass-market computer to include a **blitter** in its video hardware.
- IBM's 8514 graphics system was one of the first **PC video cards** to implement 2D primitives in hardware.
- Games were the driving force of graphics hardware development

1990's

- Apple Macs, MS Windows and Unix Workstations required high-speed, high-resolution 2D bitmapped graphics.
- PC graphics manufacturers developed the Graphics Device Interface (**GDI**) a single programming interface.
- By the mid 1990's there was demand for hardware-accelerated 3D graphics.
- The **graphics card** was now an important part of the PC.

A Brief History



Late 1990's

- Video, 2D GUI acceleration, and 3D functionality were all integrated into one chip.
- **AGP** became standard
- **DirectX** became one of the leading 3D graphics programming interfaces
- Hardware-accelerated transform, lighting, pixel shading and vertex shading
- The **GPU** was an established feature of many computer systems

2000's

- Programmable GPU's
- Parallel GPU's
- **PCI Express** - 3rd Generation I/O bus



Input/Output Bus

- **PCI - Peripheral Component Interconnect**
 - replaced the older ISA standard, allowed for plug and play
 - parallel bus
 - 133 MB per second, 32-bit or 64-bit bus width
- **AGP - Accelerated Graphics Port**
 - dedicated pathway to the processor
 - parallel bus
 - reads directly from system RAM using the Graphics Address Remapping Table (GART)
 - 2133 MB per second (2GB/s), 32-bit bus width
- **PCIe - PCI Express**
 - uses serial physical-layer communications protocol
 - 16 channels (250 MB/s for each channel) = ?? GB/s

Graphics Cards

- Dedicated Cards

- Graphics cards that have RAM that is dedicated to the card's use
- Internal bandwidth of 15GB/sec to 40GB/sec

- Integrated Cards

- Graphics processors that utilize a portion of a computer's system RAM rather than dedicated memory.
- Relies on the the I/O bus to connect to RAM
- Bandwidth of 2GB/sec to 8GB/sec (AGP or PCIExpress 2.0)



GPU's

- A Graphics Processing Unit is a dedicated graphics rendering device on a Graphics Card.
- It takes the load off of the CPU
- Modern GPU's support 3D computer graphics and digital video
- Parallel GPUs are becoming as flexible as CPUs. They are programmable and much faster for image-array operations.