Cell Array, Character Generation & Anti Aliasing in Computer Graphics Unit 1

A cell array is a data structure that allows for the storage of data in an array-like format. It is similar to a regular array, but with the added benefit of being able to store data of different types in each individual cell. This makes it ideal for storing data for use in computer graphics applications.
One example of where a cell array would be useful is when storing images for use in an image processing application. Each cell could contain an image, and the application could then operate on all the images in the cell array simultaneously. This would be much more efficient than having to process each image individually.
Another example where a cell array can be used is when storing 3D models for use in a computer game or other 3D applications. Each cell can contain a different model, and the application can then render all the models in the cell array at once. This can save a lot of time, as opposed to having to load and render each model individually.

Examples:

In computer graphics, a cell array is an array of cells, each of which stores data associated with a particular graphical object. For example, a cell might store the data for a line or polygon, or it might store the data for an image.
The cell array is a primitive that allows users to display an arbiter shape defined as a two-dimensional grid pattern. A predefined matrix of color values is mapped by this function onto a specified rectangular coordinate region.
Cell arrays are used in many different ways in computer graphics. One common use is to store the data for a set of objects that are all to be displayed on the screen at once. This can be useful, for example, when you want to display a group of objects that all have the same color or style.
Another common use for cell arrays is to store the data for a set of objects that are to be displayed one at a time. This can be useful, for example, when you want to display a sequence of images or animations.
Finally, cell arrays can also be used to store the data for a set of objects that are not necessarily intended to be displayed on the

screen at all. This can be useful, for example, when you want to process the data in some way before it is displayed.

Approach:

There are many ways to approach cell arrays in computer graphics. One way is to create test cases that can be used to explore the behavior of cell arrays. Another way is to look at how cell arrays are used in other areas of computer science.

Cell arrays can be used to store data in a structured way. They can be used to store information about objects in a scene, or they can be used to store results from computations. Cell arrays are often used when there is a need to store data in a two-dimensional structure.

One use for cell arrays is storing the vertices of polygons. Each row in the cell array can represent a different polygon, and each column can represent a different vertex within that polygon. The data in the cells can then be used to draw the polygons on the screen.

Another use for cell arrays is storing image data. The cells can represent pixels, and the data in the cells can be used to determine the color of each pixel. This type of storage is very efficient because it doesn’t require much memory.

Cell arrays can also be used to store results from computations. For example, if you were doing a simulation of some physical system, you could use a cell array to store the position of each particle at each time step. This would allow you to easily access the data later if you wanted to analyze it or visualize it in some way.

Advantages:

A cell array is a data structure that is used to store data in an ordered manner. It is similar to a matrix, but it can store data of different types and sizes. A cell array is used in computer graphics to store information about the colors of pixels in an image.

The advantage of using a cell array is that it can store data of different types and sizes, which makes it more flexible than a matrix. The disadvantage is that it takes up more memory than a matrix.

Disadvantages:

There are several disadvantages to using cell arrays in computer graphics. One is that they can be difficult to work with and understand. Another is that they can take up a lot of memory, which can be a problem when working with large images or files. Finally, they can be slow to render, which can impact the overall performance of your system.

Applications:

A cell array is a data structure that stores data in an ordered, rectangular grid. The most common application of cell arrays is to store images, but they can also be used to store other types of data such as 3D models, textures, and animation frames.
Cell arrays are often used in computer graphics because they offer a efficient way to store and manipulate large amounts of data. For example, when rendering a 3D scene, each frame may be represented by a cell array. By storing the scene in a cell array, the renderer can easily access and modify any element of the scene without having to recalculate the entire image.
Another common use for cell arrays is to store animations. Animations are typically composed of a sequence of images, each of which is represented by a cell array. By storing the animation in a cell array, the animator can easily access and modify any frame of the animation without having to recalculate the entire sequence.

Character Generation Types :

- We can display letters and numbers in variety of size and style.
- The overall design style for the set of character is called typeface.
- Today large numbers of typefaces are available for computer application for example Helvetica, New York Platino etc.
- Originally, the term font referred to a set of cast metal character forms in a particular size and format, such as 10-point Courier Italic or 12-point Palatino Bold. Now, the terms font and typeface are often used interchangeably, since printing is no longer done with cast metal forms.
- Two different representations are used for storing computer fonts.

Bitmap font / Bitmapped Font

- A simple method for representing the character shapes in a particular typeface is to use rectangular grid patterns.
- Figure below shows pattern for particular letter.

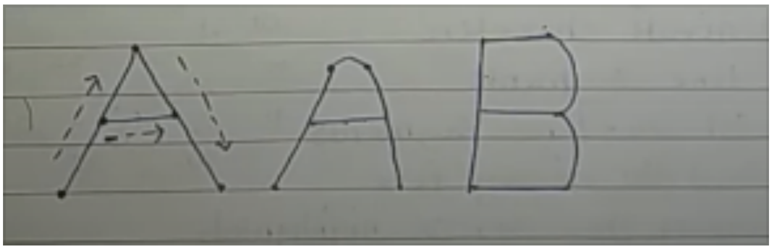
1	1	1	1	1	1	0
0	1	1	0	0	1	1
0	1	1	0	0	1	1
0	1	1	1	1	1	0
0	1	1	0	0	1	1
0	1	1	0	0	1	1
1	1	1	1	1	1	0

Fig. : - Grid pattern for letter B.

- When the pattern in figure is copied to the area of frame buffer, the 1 bits designate which pixel positions are to be displayed on the monitor.
- Bitmap fonts are the simplest to define and display as character grid only need to be mapped to a frame-buffer position.
- Bitmap fonts require more space because each variation (size and format) must be stored in a font cache.
- It is possible to generate different size and other variation from one set but this usually does not produce good result.

Stroke Method

Here we create character’s lines and arc using bresenhems line drawing and circle drawing methods.
So here the character will be a collection of lines and arcs.



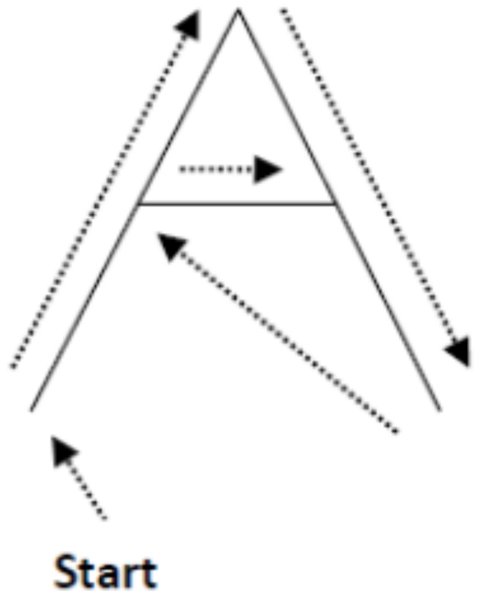
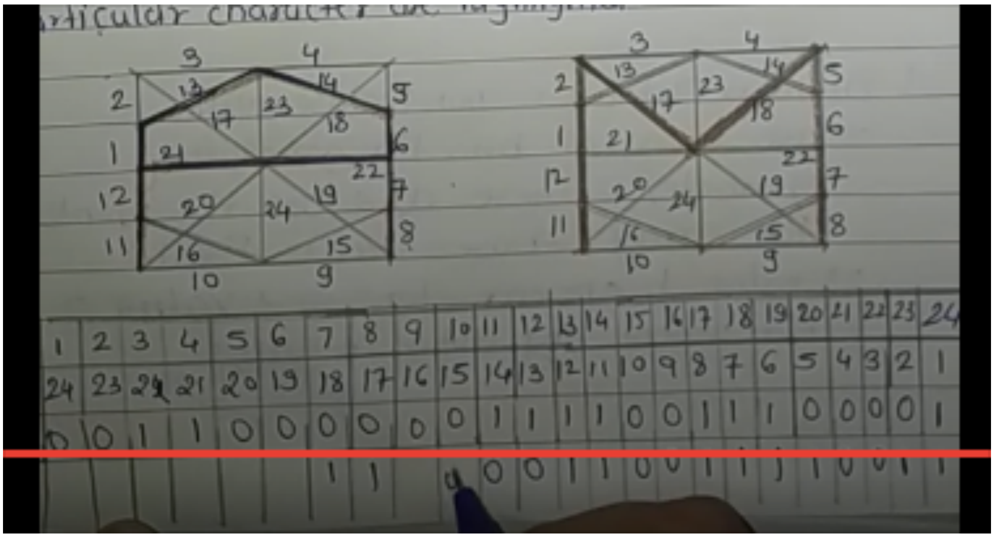


Fig. .': - Stroke Method for Letter A.

- It uses small line segments to generate a character.
- The small series of line segments are drawn like a stroke of a pen to form a character as shown in figure.
- We can generate our own stroke method by calling line drawing algorithm.
- Here it is necessary to decide which line segments are needs for each character and then draw that line to display character.
- It support scaling by changing length of line segment.

Star Burst Method

Here we use fixed lines and highly some specific lines to draw a character.



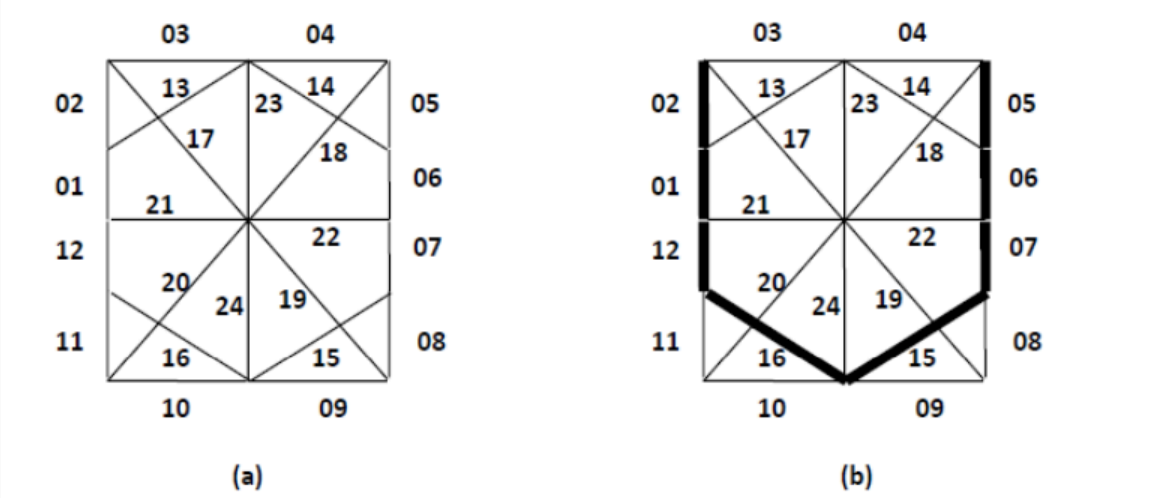


Fig. : - (a) Starbust Method. (b) Letter V using starbust method

- In this method a fix pattern of line segments are used to generate characters.
- As shown in figure there are 24 line segments.
- We highlight those lines which are necessary to draw a particular character.
- Pattern for particular character is stored in the form of 24 bit code. In which each bit represents corresponding line having that number.
- That code contains 0 or 1 based on line segment need to highlight. We put bit value 1 for highlighted line and 0 for other line.
- Code for letter V is 110011100001001100000000
- This technique is not used now a days because:

1. It requires more memory to store 24 bit code for single character.

2. It requires conversion from code to character.
3. It doesn't provide curve shapes.

Anti-Aliasing - Computer Graphics

Antialiasing is a computer graphics method that removes the aliasing effect. The aliasing effect occurs when rasterised images have jagged edges, sometimes called "jaggies" (an image rendered using pixels). Technically, jagged edges are a problem that arises when scan conversion is done with low-frequency sampling, also known as **under-sampling**, this under-sampling causes distortion of the image. Moreover, when real-world objects made of continuous, smooth curves are rasterised using pixels, aliasing occurs. Under-sampling is an important factor in anti-aliasing. The information in the image is lost when the sample size is too small. When sampling is done at a frequency lower than the Nyquist sampling frequency, under-sampling takes place. **We must have a sampling frequency that is at least two times higher than the highest frequency appearing in the image in order to prevent this loss.**

Anti-Aliasing Methods:

A high-resolution display, post-filtering (super-sampling), pre-filtering (area sampling), and pixel phasing are the techniques used to remove aliasing. The explanations of these are given below:

1. **Using High-Resolution Display** - Displaying objects at a greater resolution is one technique to decrease aliasing impact and boost the sampling rate. When using high resolution, the jaggies are reduced to a size that renders them invisible to the human eye. As a result, sharp edges get blurred and appear smooth.
   Real-Life Applications:
   For example, OLED displays and retina displays in Apple products both have high pixel densities, which results in jaggies that are so microscopic that they are blurry and invisible to the human eye.
2. **Post-Filtering or Super-Sampling** - With this technique, we reduce the adequate pixel size while improving the sampling resolution by treating the screen as though it were formed of

a much finer grid. The screen resolution, however, does not change. Now, the average pixel intensity is determined from the average of the intensities of the subpixels after each subpixel's intensity has been calculated. In order to display the image at a lesser resolution or screen resolution, we do sampling at a higher resolution, a process known as supersampling. Due to the fact that this process is carried out after creating the rasterised image, this technique is also known as post filtration.

Real-Life Applications:

The finest image quality in gaming is produced with SSAA (Super-sample Antialiasing) or FSAA (full-scene Antialiasing). It is frequently referred to as the "**pure AA,"** which is extremely slow and expensive to compute. When no better AA techniques were available, this technique was frequently utilised in the beginning. Other SSAA modes are available, including 2X, 4X, 8X, and others that indicate sampling that is done x times (greater than) the present resolution.

MSAA (multisampling Antialiasing), a quicker and more accurate version of super-sampling AA, is a better AA type. Its computational cost is lower. Companies that produce graphics cards, such as CSAA by NVIDIA and CFAA by AMD, are working to improve and advance super-sampling techniques.

3. **Pre-Filtering or Area-Sampling** - The areas of each pixel's overlap with the objects displayed are taken into account while calculating pixel intensities in area sampling. In this case, the computation of pixel colour is centred on the overlap of scene objects with a pixel region.

   Example: Let's say a line crosses two pixels. A pixel that covers a larger amount of a line (90%) displays 90% intensity, whereas a pixel that covers a smaller piece (10%) displays 10-15% intensity. If a pixel region overlaps with multiple colour areas, the final pixel colour is calculated as the average of those colours. Pre-filtering is another name for this technique because it is used before rasterising the image. Some basic graphics algorithms are used to complete it.
4. **Pixel Phasing** - It is a method to eliminate aliasing. In this case, pixel coordinates are altered to virtually exact positions close to object geometry. For dispersing intensities and aiding

with pixel phasing, some systems let you change the size of individual pixels.

Application of Anti-Aliasing:

1. **Compensating for Line Intensity Differences** - Despite the diagonal line being 1.414 times larger than the horizontal line when a horizontal line and a diagonal line are plotted on a raster display, the amount of pixels needed to depict both lines is the same. The extended line's intensity decreases as a result. Anti-aliasing techniques are used to allocate the intensity of pixels in accordance with the length of the line to make up for this loss of intensity.
2. **Anti-Aliasing Area Boundaries** - Jaggies along area boundaries can be eliminated using anti-aliasing principles. These techniques can be used to smooth out area borders in scanline algorithms. If moving pixels is an option, they are moved to positions nearer the edges of the area. Other techniques modify the amount of pixel area inside the boundary by adjusting the pixel intensity at the boundary position. Area borders are effectively rounded off using these techniques.