Segments in Computer Graphics

introduction:

Introduction segments are a fundamental concept in computer graphics, used to represent the basic building blocks of a graphical scene. They are commonly used in 2D graphics to represent lines or curves that connect two or more points.

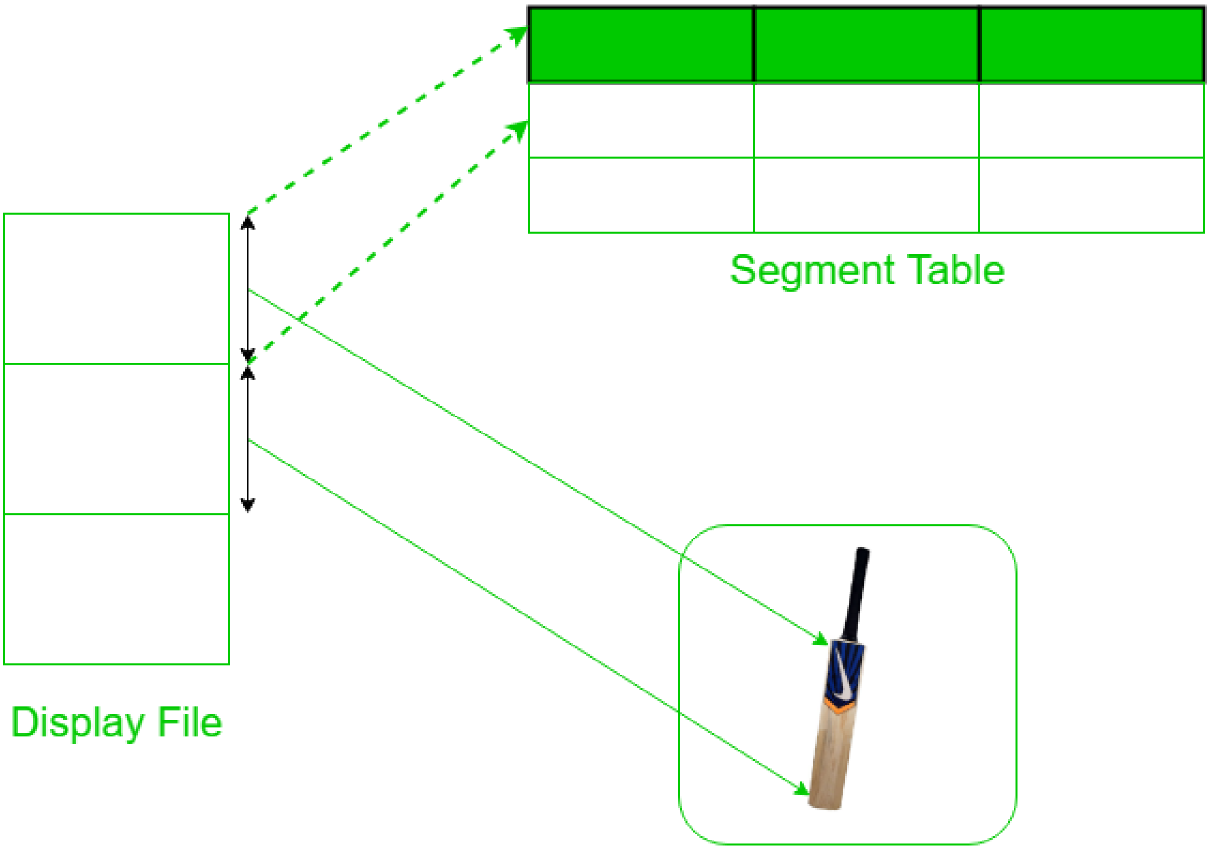
An introduction segment is defined by two endpoints, which are specified using their (x,y) coordinates in a 2D Cartesian coordinate system. Once defined, the introduction segment can be used to represent a variety of different shapes and objects, depending on its length, curvature, and orientation.

One common use of introduction segments is in the rendering of vector graphics, where they are used to represent the individual lines and curves that make up a complex image. In this context, introduction segments are often referred to as "paths", and can be used to represent everything from simple shapes like squares and circles, to complex curves like Bezier curves or splines.

Another common use of introduction segments is in computer-aided design (CAD) applications, where they are used to represent the basic building blocks of 2D and 3D models. In this context, introduction segments can be used to represent everything from straight edges and curves to more complex shapes like spirals and helixes.

To view an entire image or a part of image with various attributes, we need to organize image information in a particular manner since existing structure of display file does not satisfy our requirements of viewing an image. To achieve this display, file is divided into **Segments**. Each segment corresponds to a component and is associated with a set of attributes and image transformation parameters like scaling, rotation. Presence of Segment allows :

- Subdivision of picture.
- Visualization of particular part of picture.
- Scaling, rotation and translation of picture.



Types of Segments :

- **Posted Segment :** When visible attribute of segment is set to 1, it is called Posted segment. This is included in active segment list.
- **Unposted Segment :** When visible attribute of segment is set to 0, it is called Unposted segment. This is not included in active segment list.

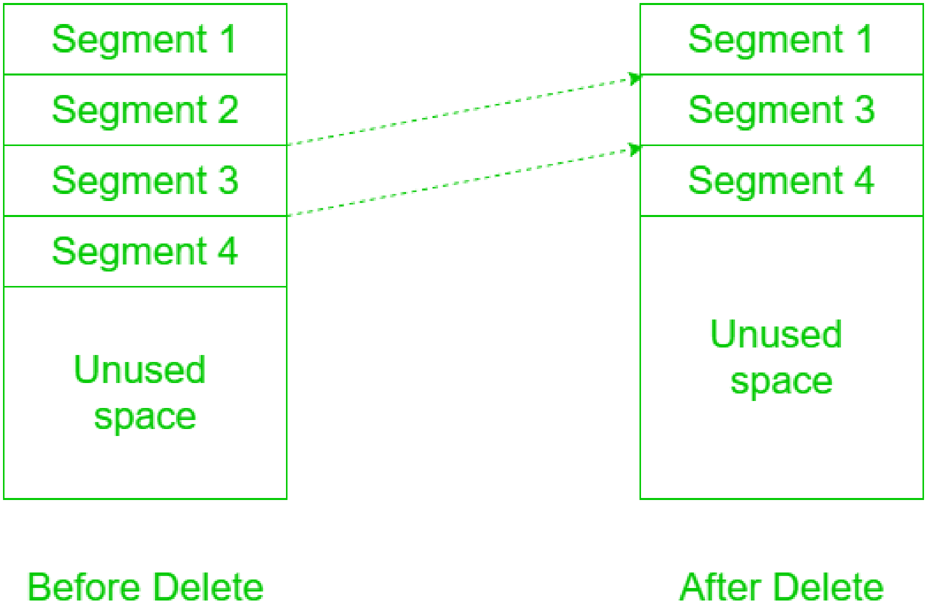
Functions for Segmenting the display :

1. **Segment Creation :** Segment must be created or opened when no other segment is open, since two segments can't be opened at the same time because it's difficult to assign drawing instruction to particular segment. The segment created must be given a name to identify it which must be a **valid** one and there should be no segment with the same name. After this, we initialize items in segment table under our segment name and the first instruction of this segment is allocated at next free storage in display file and attributes of segments are initialized to default. **Algorithm :**
    1. If any segment is open, give error message : ***"Segment is still open"*** and go to step 8.

    2. Read the name of the new segment.
    3. If the segment name is not valid, give error message : ***"Segment name not a valid name"*** and go to step 8.
    4. If given segment name already exists, give error message : ***"Segment name already exists in name list"*** and go to step 8.
    5. Make next free storage area in display file as start of new segment.
    6. Initialize size of new segment to 0 and all its attributes to their default values.
    7. Inform that the new segment is now open.
    8. Stop.

2. **Closing a Segment :** After completing entry of all display file instructions, the segment needs to be closed for which it has to be renamed, which is done by changing the name of currently open segment as 0. Now the segment with name 0 is open i.e. unnamed segment is open and if two unnamed segments are present in display file one needs to be deleted. **Algorithm :**
    1. If any segment is not open, give error message : ***"No segment is open now"*** and go to step 6.
    2. Change the name of currently opened segment to any unnamed segment, lets say 0.
    3. Delete any other unnamed segment instruction which may have been saved and initialize above unnamed segment with no instructions.
    4. Make the next free storage area available in display file as start of the unnamed segment.
    5. Initialize size of unnamed segment to 0.
    6. Stop.
3. **Deleting a Segment :** To delete a particular segment from display file, we must just delete that one segment without destroying or reforming the entire display and recover space

occupied by this segment. Use this space for some other segment. The method to achieve this depends upon the data structure used to represent display file. In case of arrays, the gap left by deleted segment is filled by shifting up all the segments following it.



Display file contents before and after deleting Segment 2

Algorithm :

1. Read the name of the segment to be deleted.
2. If segment name is not valid, give error message : ***"Segment name is not a valid name"*** and go to step 8.
3. If the segment is open, give error message : ***"Can't delete an open segment"*** and go to step 8.
4. If size of segment is less than 0, no processing is required and go to step 8.
5. The segments which follow the deleted segment are shifted by its size.
6. Recover deleted space by resetting index of next free instruction.
7. The starting position of shifted segments is adjusted by subtracting the size of deleted segment from it.

    8. Stop.

4. **Renaming a Segment :** This is done to achieve **Double Buffering** i.e. the idea of storing two images, one to show and other to create, alter and for animation. **Algorithm :**
    1. If both old and new segment names are not valid, give error message : ***"Segment names are not valid names"*** and go to step 6.
    2. If any of two segments is open, give error message : ***"Segments are still open"*** and go to step 6.
    3. If new segment name given already exists in the display list, give error message : ***"Segment name already exists"*** and go to step 6.
    4. The old segment table entry are copied into new position.
    5. Delete the old segment.
    6. Stop.

Advantages of using segmented display :

- Segmentation allows to organize display files in sub-picture structure.
- It allows to apply different set of attributes to different portions of image.
- It makes it easier to the picture by changing/replacing segments.
- It allows application of transformation on selective portions of image.

Uses of Segment :

1. Drawing lines and curves: Segments are commonly used to represent lines and curves in 2D graphics. By connecting two or more points with a series of line segments, you can create a variety of shapes and objects, including geometric shapes, symbols, and illustrations.
2. Vector graphics: In vector graphics, segments are used to define the paths of shapes and objects. By defining a series of

interconnected segments, you can create complex curves and shapes that can be resized and transformed without losing their clarity or resolution.

3. Computer-aided design (CAD): Segments are widely used in CAD applications to define the edges and curves of 2D and 3D models. By connecting a series of segments together, you can create complex shapes and objects that can be manipulated and transformed in a virtual environment.
4. Animation: Segments are used extensively in animation to create smooth, fluid motion. By defining a series of keyframes and connecting them with segments, you can create animations that mimic the movement of real objects and characters.
5. Computer vision: Segments are often used in computer vision applications to identify and track objects in real-time video. By defining a set of reference points on an object and tracking their movement using segments, you can create a model of the object's motion and use it to track it in a video stream.

Issues in Segment :

Here are a few of the main issues to consider:

1. Aliasing: When using segments to represent lines or curves, aliasing can occur if the segments are not defined with sufficient detail. This can result in jagged edges or rough curves that do not accurately represent the intended shape.
2. Performance: If segments are used extensively in a graphics application, it can impact performance and slow down rendering times. This can be especially problematic in real-time applications, such as video games, where fast frame rates are essential for a smooth user experience.
3. Memory usage: Depending on the complexity of the scene being rendered, segments can require a significant amount of memory. This can be a concern in applications with limited memory resources, such as mobile devices or embedded systems.
4. Accuracy: While segments can be used to represent complex shapes and objects, their accuracy is limited by the number of

segments used to define them. To achieve a high level of accuracy, a large number of segments may be required, which can impact performance and memory usage.

5. Rendering artifacts: When using segments to create complex shapes or objects, rendering artifacts can occur if the segments are not connected properly. This can result in visible gaps or overlaps between segments, which can be distracting or visually unappealing.