Chapter 5

Implementing Windowing using Display File for Bioinformation

5.1 Introduction

A bioinformation user have a graphics programme to draw an entire DNA, RNA structure but be interested only the one strand of DNA. Often, the computer is used in design application because it can easily and accurately create, store, and modify very complex drawings. When drawings are too complex, however, they become difficult to read. In such situation it is useful to display only those portions of the drawing that are of immediate interest. This gives the effect of looking at the image through a window. It is desirable to enlarge these portions to take full advantage of the available display surface. The method for selecting and enlarging portion of a DNA is called windowing. The technique for not showing that part of the drawing
which is not interested in is called clipping. The term windowing has been used to mean several different things, all related to partitioning of the display.

In this chapter we shall consider the ideas of windowing, clipping and viewing transformation. We will add to our graphics package the routines for setting windows and viewports and for removing lines which lies outside the DNA structure display. The transformation theory explained in previous chapter will guide us in setting up a viewing transformation which will be applied to each display file instruction as the instruction is created.

5.2 Viewing Transformation for DNA Image

Sometimes two models of the DNA we are displaying. There is the object model and there is the image of the object which appears on the display. When the image is considered we are actually referring to a model of the DNA stored in the computer. The object model is said to reside the object space. This DNA model represents the object using the physical units of the length. In the object space, length of the object may be measured in any units from light-years to angstroms. The length of the image on the screen, however must be measured in screen coordinates.
There is a way of converting from the object space units of measure to those of the image space. This can be done by the scaling transformation. By scaling, we can uniformly reduce the size of the object until its dimensions lie between 0 and 1. Very small DNA structure can be enlarged until their overall dimensions almost 1 unit. The physical dimensions of the Bioinformation image are scaled until they are suitable for display.

It may be however that the object is too complex to show in its entirely or that we are particularly interested in just a portion of it. We would like to imagine a box about a portion of the object. We would only display what is enclosed in the box. Such a box is called a window.

It might also happen that we do not wish to use the entire screen for the display. We would like to imagine a box on the screen and have the image confirmed to that box. Such a box in that screen space is called a viewport.

When the window is changed, we see a different part of the object shown at the same position on the display. If we change the viewport,
we see the same part of the object drawn at a different place on the display.

In specifying both window and viewport it have enough information to determine the translation and scaling transformations necessary to map from the object space to the image space. This can be done with the following three steps.

First the object together with its window is translated until the lower left corner of the window is at the origin. Second, the object and window are scaled until the window has the dimensions of the viewport. In effect, this converts object and window into image and viewport. The final transformation step is another translation to move the viewport to its correct position on the screen.

We are really trying to do two things. We are changing the window size to become the size of the viewport and we are positioning it at the desired location on the screen. The positioning is just moving the lower left corner of the window to the viewports lower left corner location, but we do this in two steps. We first move the corner to the origin and second move it to the viewport corner location. We take two steps because while it is at the origin, we can perform necessary scaling without disturbing the corner’s position.
The overall transformation which performs these three steps we shall call the viewing transformation of Bioinformation image. It creates a particular view of the object. The height and width of the windows do not have the same proportion as the height and width of the viewport, then the viewing transformation will cause some distortion of the image in order to squeeze the shape selected by the window into the shape presented by the viewport.

5.3 Viewing Transformation Implementation

The first step of our viewing transformation is specifying the size of the DNA image window. Then confine for window to a rectangular shape parallel with the x and y axes. By doing this need only specify the smallest and largest possible x values and the smallest and largest possible y values. Our routine for specifying the size of the window will store this boundary values in global variable so that they will be available when it comes time to perform the transformation. Likewise, we must specify the boundaries of the viewport. The length of the viewport or window in either the x or the y dimension is determined by subtracting the lower boundaries from the upper boundary. Note that we cannot have the lower boundary equal to the upper boundary.
for a window because this would cause us to divide by zero when we try to determine the scale scaling transformation.

In this system it not changes viewing parameters in the middle of a segment. Each segment is treated as a snapshot of the object. The viewing transformation describes how the camera is positioned. In this model it is reasonable to prohibit movement of the camera while taking the picture. We shall follow this rule by keeping two sets of viewing parameters, one set for the user to change and second set to actually be used in the windowing and clipping routines. Changing the window in our system becomes two step process. First the user changes his set of window and viewport boundaries. Second, the user’s values are copies into the variables actually used by the windowing and clipping routines. By performing this copying as part of the segment creation process, we ensure that changes in the viewing parameters being used cannot occur in the middle of a segment.

We wish any given window setting to apply to an entire display file segment of Bioinformation image. We can enforce this restriction by only allowing the above copying of parameters to occur when a segment is created. We shall therefore modify our segment creation
routine to reset the viewing transformation to match the latest user request.

We wish to perform the following transformations. First, we wish to translate by the lower x and y boundaries of the window. This moves the lower left corner of the window to the origin. Second, we wish to scale by the size of the viewport divided by the size of the window. This changes the dimensions of the window to those of the viewport. Finally, we wish to translate by the lower x and y boundary values of the viewport. This moves the lower left corner from the origin to the correct viewport position. We can form each of these transformation matrices as we did. We can multiply the matrices together to form a single transformation. And then apply it to a general point.

Clipping for Bioinformation Image

If user needs not to display particular part he can clip it. After scaled and positioned, we shall consider how to cut off the lines which are outside the windows so that only the lines within the windows are displayed. This process is called clipping. In clipping each line of the display to determine whether or not it is completely inside the windows, lies completely outside the windows or cross a window boundary. If it is inside, the line is displayed; if it is outside, nothing
is drawn. If it crosses the boundary, we must determine the point of intersection and draw only the portion which lies inside.

5.4 Suggested Algorithm for Setting Viewport for Bioinformation Image

Algorithm SET-VIEWPORT(XL,XH,YL,YH) User routine for specifying the viewport.

Arguments XL, XH the left and right viewport boundaries

YL, YH the bottom and top viewport boundaries.

Variable VXL-HOLD, VXH-HOLD, VYL-HOLD, VYH-HOLD storage for the viewport boundaries.

BEGIN

VXL-HOLD <- XL;

VXH-HOLD <- XH;

VYL-HOLD <- YL;

VYH-HOLD <- YH;

END;

5.5 Suggested Algorithm for Window for Bioinformation

Algorithm SET-WINDOW(XL,XH,YL,YH) User routine for specifying the window.

Arguments XL, XH the left and right window boundaries
YL, YH the bottom and top window boundaries.

Variable WXL-HOLD, WXH-HOLD, WYL-HOLD, WYH-HOLD storage for the window boundaries.

BEGIN

WXL-HOLD <- XL;
WXH-HOLD <- XH;
WYL-HOLD <- YL;
WYH-HOLD <- YH;

END;

5.6 Suggested Algorithm for viewing parameter

Algorithm to set the clipping and viewing parameters from window and viewport specifications.

Variable VXL-HOLD, VXH-HOLD, VYL-HOLD, VYH-HOLD viewport parameters

WXL-HOLD, WXH-HOLD, WYL-HOLD, WYH-HOLD window parameters

WXL, WXH, WYL, WYH, VXL, VXH, VYL, VYH
the current clipping parameters

WSX, WSY the window-to-viewport scale factors

BEGIN

WXL <- WXL-HOLD;

WXH <- WXH-HOLD;
WYL <- WYL-HOLD;
WYH <- WYH-HOLD;
VXL <- VXL-HOLD;
VXH <- VXH-HOLD;
VYL <- VYL-HOLD;
VYH <- VYH-HOLD;
WSX <- (VXH - VXL) / (WXH - WXL);
WSY <- (VYH - VYL) / (WYH - WYL);
END;

5.7 Suggested Algorithm for Viewing Transformation of a any point of Bioinformation Image

VIEWING-TRANSFORM (OP,X,Y) Viewing transformation of a point.

Arguments  OP, X, Y the instruction to be transformed.

Variable  WXL, WYL, WSX, WSY, VXL, VYL window and viewport parameters

X1, Y1 the transformed point

BEGIN

X1 <- (X - WXL) * WSX + VXL;
Y1 <- (Y - WYL) * WSY + VYL;
PUT-POINT(OP, X1, Y1);
END;

Algorithm DISPLAY-FILE-ENTER(OP) (Modification of Algorithm 2.23) Combine
operation and position to form an instruction and save it in the display file

Argument OP the operation to be entered
Global  DF-PEN-X, DF-PEN-Y the current pen position
BEGIN
IF OP < 1 AND OP > -32 THEN PUT-POINT(OP, 0, 0)
ELSE CLIP(OP, DF-PEN-X, DF-PEN-Y);
RETURN;
END;

5.8 Suggested Algorithm for Display File Entry
Algorithm DISPLAY-FILE-ENTER(OP) Combine operation and position to form an instruction and save it in the display file

Argument OP the operation to be entered
Variable  DF-PEN-X, DF-PEN-Y the current pen position
BEGIN
IF OP < 1 AND OP > -32 THEN PUT-POINT(OP, 0, 0)
ELSE CLIP(OP, DF-PEN-X, DF-PEN-Y);
END;

5.9 Multiple Windowing for Bioinformation Image

Sometimes there is need for multiple windowing; that is, a first image is created by one or more window transformations on the object. Then, windows are applied to this first image to create the second image. Further windowing transformations may be done until the desired picture is created. Every application of a window transformation allows the user to slice up a portion of the picture and reposition it on the screen. Thus multiple windowing gives the user freedom to rearrange components of the picture. The same effect may be achieved however by applying a number of single window transformations to the object.