

# Summary of Comments on 2006 IGCAEA Conference (Cal Poly)

---

## Page: 1

---

Sequence number: 1

Author: jking

Date: 6/27/2006 7:30:42 AM

Type: Note

Thanks for looking at my presentation. You might want to print these comments out and then read them as you look at the display of the slides. Maybe you are one of those lucky folks that have two displays or one large enough to display both side by side. Or you can leave the notes where they are and double click on them to read them.

Jim King - 6/27/2006

## Page: 2

---

Sequence number: 1

Author: jking

Date: 6/27/2006 7:31:26 AM

Type: Note

I probably should have titled the talk "Resolution Independent Specification and Rendering" because the objects to be rendered must be supplied in a resolution independent fashion. Then we must render that material to any display of any resolution.

I work for Adobe Systems Incorporated (I am just completing my 15th year working at Adobe) and the company was founded in 1982 to design and supply software for PostScript®, the standard device independent language for describing high quality professional documents for printing and display. Also note that the same "imaging model" that was defined for PostScript, forms the basis of our very popular standard Portable Document Format or PDF.

## Page: 3

---

Sequence number: 1

Author: jking

Date: 6/27/2006 7:31:37 AM

Type: Note

PostScript supports three main "inking" types of information: sampled images, vector graphics and typographic text. Note that I put important adjectives in front of each of those (sampled, vector, and typographic). I will be talking about the importance of each of those adjectives during this presentation.

I think we all know what an image is. It is simply a 2-D array of color values usually arranged left

to right top to bottom that describe what color is found in a picture with the grid of the array laid over the top of it. We will get back to the significance of this particular description in a minute. Next is what is traditionally called "vector graphics." In PostScript and PDF, this might better be called "path graphics" because it is based upon describing paths in a 2-D coordinate system and then filling those paths with a color or "stroking" the path to provide an outline. And of course, no imaging model is complete without the ability to display text, and in our case very high quality text using nearly any font that has been designed and used with printing presses over the last 400 years. We are going to go into detail about resolution independent representations of each of these.

## Page: 5

---

Sequence number: 1

Author: jking

Date: 6/27/2006 7:31:51 AM

Type: Note

Today's PCs and most other computers drive the image to be displayed off of a RAM buffer that contains the color values for each unique color position possible on the display. These are typically called "frame buffers". Then we have images that I will call bit mapped because the pixels, or picture elements, are mapped to positions and colors in the frame buffer and hence on the display. If one wants something to be displayed, one simply stored the correct color values, or pixels into the frame buffer RAM storage area in the correct spot and using the correct color representation.

This is a very general, fast and efficient way to interface to a display. Speed was such a concern for getting information into the display's bitmap that specially designed circuits or tightly coded software was created to do the bit moving. These went under the name "bit blt" pronounced "bit blit". I believe that the "blt" came from an early DEC computer instruction that stood for "block transfer."

Of course, as the resolution of displays increases the number of bits increase as the square, so although our circuit speeds have increased an amazing amount in the last 10-20 years, getting the bits moved around is still an interesting activity and leads us to the whole business of video cards for PCs.

## Page: 6

---

Sequence number: 1

Author: jking

Date: 6/27/2006 7:32:19 AM

Type: Note

First some basics. I am sure you all know the next few things I am going to talk about but it never hurts to make sure we are all viewing things the same way. If I have an image represented as an array of color values, which we typically call "pixels" for picture elements, then I could copy those pixels into an area from which a display is being generated and I can get that image displayed. Suppose that I am using a display that shows 100 pixels per inch in both dimensions. Then if I have an image that contains 100 x 100 pixels in its descriptive array, it will display as a 1 inch by 1 inch image on the display.

## Page: 7

---

Sequence number: 1

Author: jking

Date: 6/27/2006 7:32:40 AM

Type: Note

If we copy that same image, unchanged, into the frame buffer for a display with twice the resolution (say 200 pixels per inch) then I get a image 1/2 the size.

## Page: 8

---

Sequence number: 1

Author: jking

Date: 6/27/2006 7:32:54 AM

Type: Note

In order to get the same size image displayed on a the display with twice the resolution, then I need to double the number of pixels in the image array in each dimension to 200 x 200. Of course, there is a seesaw between the number of pixels in the image array and the size that image will appear on a display of some given resolution.

## Page: 9

---

Sequence number: 1

Author: jking

Date: 6/27/2006 7:33:05 AM

Type: Note

What if I had an image that I wanted to display at a give size on the display but the resolution for the image does not match the resolution for the display in a way that would result in the image coming out the correct size. For example, when we distribute a PDF file containing an image to 1,000 people we certainly cannot insist that each of those people have a display of the same common resolution. This brings us to what Adobe calls a "sampled image."

## Page: 10

---

Sequence number: 1

Author: jking

Date: 6/27/2006 7:33:14 AM

Type: Note

The word "sampled" is used to emphasize the notion of laying a grid of some resolution on top of the image and then describing the color found in the picture at each grid point. What determines

the resolution of the grid I use to “sample” the colors of the image? The answer should not be “Well what is the resolution on which you want to display that image”. It should be “Well what resolution would capture the most amount of the detail in that image that you want.” In any case, the point is that one cannot sample images at some device resolution because of the requirement for displaying the image on a wide variety of devices with widely varying resolutions. So we use the term “sampled image” to emphasize the fact that the image has its own resolution or number of pixels used to represent it and it is not determined by any particular device.

## Page: 1 1

---

Sequence number: 1

Author: jking

Date: 6/27/2006 7:33:24 AM

Type: Note

We then render the pixels of the sampled image onto the display possibly transformed by scaling, rotating or skewing. The resolution at which the image is sampled is completely independent from the resolution of the display.

## Page: 1 2

---

Sequence number: 1

Author: jking

Date: 6/27/2006 7:33:32 AM

Type: Note

Of course, if I do not insist that all my images are prepared at a resolution that will make them appear correctly on a given device then I must do some rather complicated adjustments as I move the image from, say a PDF file, to the display buffer. One might think of this as a sub-sampling or super-sampling operation (changing the number of pixels used to represent the image). But it might interest you to know that Adobe’s method for doing this is integrated into the rendering operation. Scaling, rotating, obliqueing and even halftoning are all done using a conceptually simple scheme. For each pixel wanted for the particular device, we compute which pixel in the original image is “over” that device pixel. Then we color it accordingly. The back computation takes into account any rotation or scaling required. If halftoning is also being done, then a threshold array is introduced as a filter between the device pixels and the sampled image pixels to create the appropriate halftone effect.

## Page: 1 4

---

Sequence number: 1

Author: jking

Date: 6/27/2006 7:33:43 AM

Type: Note

Here we have two images of a blossom. The one on the left is described using vector graphics and the one on the right is a sampled image. You may be able to detect a slight difference but the

image has been sampled at a reasonably high resolution so it looks quite smooth and nice.

## Page: 15

---

Sequence number: 1

Author: jking

Date: 6/27/2006 7:35:22 AM

Type: Note

This is the same slide as the previous one, but I have cropped it to magnify a portion of both blossoms. If you compare these two representations you can see a vivid difference. On the far right you can see the individual pixels of the image samples. On the left you see none. I hope you are all already familiar with what is happening on the right. We have magnified the sampled image so much that the effective resolution available from its array of samples is only 10-20 pixels per inch and these are readily visible.

But what is happening on the left side to make the blossom look so good even when magnified so much? The answer is “vector graphics”. This blossom was actually drawn by hand, meticulously by a talented artist using one of Adobe’s products, Adobe Illustrator®. This product allows one to draw shapes and to fill them with solid colors or varying shades of colors. This blossom is one of the samples provided with Illustrator to show what beautiful things can be done with it. We will explore next in much more detail exactly how this kind of vector graphics works.

If you are looking at these slides using Acrobat®, you might want to return to slide 14 and zoom it to a very high level and then pan around. It is very interesting to see the difference in the two representations.

## Page: 16

---

Sequence number: 1

Author: jking

Date: 6/27/2006 7:35:54 AM

Type: Note

This is the basics of vector graphics, at least the Adobe versions of it. A path is described on a 2-D surface and then it can be “stroked,” “filled” or both. The fill can be a gradient fill or shading going from one color on one side to a different color on the other. Much more complicated shading descriptions are also supported.

## Page: 17

---

Sequence number: 1

Author: jking

Date: 6/27/2006 7:36:02 AM

Type: Note

This slide takes us to the ultimate depth of exactly how such a star is represented or described within PDF (the description used to show this star in this presentation). The star is laid out on a 2 dimensional coordinate system just like we learned about in high school math classes. In this

case we have the zero, zero point being in the middle of our star. The description in the box on the lower right are the exact PDF instructions used to draw the star. Here the “m” stands for “move to” and its parameters precede it in the notation. (The correct mathematical term for this is “postfix” notation.) So the first instruction is to move to the coordinate point (0, 64). Next it has the instruction to draw a line to the point (-17, 14) the “l” being a shorthand for “line to”. This does not actually call for any drawing to be done for the final figure but we are giving instructions for constructing an imaginary path. We continue to give line to instructions tracing the outline of the star in the two dimensional coordinate system. When we finally arrive back at the starting point, we then set an RGB color for the stroke portion of the path (the RG instruction) and then set the color for the fill portion of the path (the rg instruction) and issue the final “b” instruction, which says to “both” fill and stroke the imaginary path with real colors. Note that the move to (“m”) operation did not cause any stroking to occur. That is the difference between a move to and a line to. The move simply positions us in the 2-D space without leaving any trace information, whereas the line to contributes to the current path being constructed. There are coordinate transformation instructions also available in PostScript and PDF to be able to move, scale, rotate and otherwise warp the description of the star to fit within an overall display surface.

## Page: 18

---

Sequence number: 1  
Author: jking  
Date: 6/27/2006 7:36:12 AM  
Type: Note

One might wonder how such smooth shapes are created for the blossom without resorting to something like splines curves which would introduce some resolution dependences. The answer for Adobe’s imaging model is Bezier curves, named after the inventor, Pierre Bezier (1970). This is a very powerful way to parameterize descriptions of cubic curves. Some fascinating mathematical properties are fully explored in widely available literature.

## Page: 19

---

Sequence number: 1  
Author: jking  
Date: 6/27/2006 7:36:19 AM  
Type: Note

This is another deep dive into the exact representation of the heart in PDF notation. The Bezier curve is a cubic defined by four points. The two end points between which the curve is to be drawn and two other points that determine tangent lines to the curve at the endpoints. The further the associated tangent point is from the end point, the stronger the tangent pull toward the line between the auxiliary point and the end point becomes. And where the auxiliary point is placed determines the tangent line.

The PDF notation introduces a “c” operator standing for “curve to” and it takes 6 numbers as its preceding postfix notation. Those six numbers determine three of the points in the 2-D space and the fourth is supplied implicitly as the “current point”. After the completion of the curve to the current point is advanced to the new end point of that curve, ready for the next segment to be

added to the current path.

## Page: 20

---

Sequence number: 1

Author: jking

Date: 6/27/2006 7:36:26 AM

Type: Note

But wait a minute. The simple filling of the star or the heart with a solid color seems like a far cry from the flower that was shown earlier. Here is one of the pedals of the flower and it looks like it would take thousands of shapes of different colors to make something like this.

## Page: 21

---

Sequence number: 1

Author: jking

Date: 6/27/2006 7:36:32 AM

Type: Note

We add one more feature to the filling of shapes with color and that is shading. Instead of filling the shape with a single color different colors are used as controlled by a set of colors supplied and associated with different spots within the shape.

## Page: 23

---

Sequence number: 1

Author: jking

Date: 6/27/2006 7:36:40 AM

Type: Note

Text is represented as vector graphics in the Adobe imaging model as path outlines. Those paths form the font descriptive “programs” that define the look and feel for a complete set of characters that define that font. Since these descriptions are represented as vector graphics or path graphics, then can be scaled, rotated, positioned and otherwise transformed by the same coordinate transformations noted for regular vector graphics. Since the font characters often individually occur many times on one page or in one document (how many uses of the letter “e” are there on this page) all the same, the work of converting a outline to a properly scaled and rotated blit-able image is done once and the results are cached temporarily for the rendering of the page or document. This speeds things up considerably.

## Page: 24

---

Sequence number: 1

Author: jking

Date: 6/27/2006 7:36:49 AM

Type: Note

There are a couple of other problems when rendering text from graphic outlines. On relatively low resolution displays or printers the exact choice of which device pixel to turn on or off is critical to the appearance of the text.

For this reason, and particularly because much of this technology was first developed 20 years ago when displays were very low resolution, special techniques have been applied to the problem of reducing outlines to high quality bit map images. One of the least known technologies is what has come to be known as "hinting". Consider the rendering of a lower case "m". It has three "legs" and for many font designs it looks best if all three legs look to be exactly the same width. At low resolutions with small characters there may be a choice of using either one pixel to represent the stroke of a leg or two. That choice should be made the same for all three legs. If one just rasterizes the outline using standard rasterization techniques, it seldom happens that all three legs come out the same number of pixels wide. So "hints" are added to the font outlines to suggest that, if at all possible, these three artifacts need the same treatment.

## Page: 26

---

Sequence number: 1

Author: jking

Date: 6/27/2006 7:37:00 AM

Type: Note

Objects are defined to be rendered within a given coordinate system and our Adobe Imaging Model supports changing the coordinate system before playing out the description of an object so that it can be drawn at any place on the page. The coordinate transformation mechanism is based on a 3x3 matrix multiply and provides for the x, y placement, any scaling in either dimension and rotation and/or skewing to any degree. The matrix values are supplied as indicated on the slide by providing 6 of the 9 values for the matrix multiply (the other three values are always 1, 1, 0 so do not need to be supplied).

The 6 numbers shown with each object are the ones that would be used to position and warp the figures as shown.

## Page: 27

---

Sequence number: 1

Author: jking

Date: 6/27/2006 7:37:07 AM

Type: Note

Now I finally want to get around to relating all of what I have been talking about to high resolution displays.

## Page: 28

---



Sequence number: 1  
Author: jking  
Date: 6/27/2006 7:37:16 AM  
Type: Note

It turns out that the same problems that we have had with font characters the operating system and application developers have had with windows, controls, menus and palettes. On lower resolution displays for good quality appearance one must hand tune specific bit mapped graphics. It is so important as to exactly which pixels are turned to which color. This leads to resolution dependent graphics which when displayed unchanged on high resolution displays come out very small and hard to read.

In order to solve this problem, the developers of the operating systems and applications either have to have a two stage strategy, bitmaps for low resolution and vector graphics for high resolution, or develop hinting and gray scale methods similar to what has been done for text characters.

## Page: 29

---

Sequence number: 1  
Author: jking  
Date: 6/27/2006 7:37:24 AM  
Type: Note

On high resolution displays we need to use resolution independent methods as we have described in this talk. On high resolution displays we do not have the same problems as we had with low resolution displays where each pixels matters. On a high resolution display there might be a trade-off between having 4 or 5 pixels strokes, Whereas on a low resolution display the difference between using a 1 pixel stroke or a 2 pixel stroke is critical. The difference between a 4 pixel stroke or a 5 pixel stroke is not so great and the difference between a 20 pixel stroke or a 21 pixel stroke is hard to detect.

Support for high resolution displays is needed in the Operating Systems and applications but we have had almost 20 years of using bit blitting and resolution dependent methods. It will take time and new releases of both OSs and applications to make this happen. There is also the lingering fear that the higher resolution of the displays still requires too much processing power to do resolution independent processing. I believe this is not true. Both of the major OS vendors (Apple and Microsoft) are aware of the need to make these changes and they will be available in some future releases.

## Page: 30

---

Sequence number: 1  
Author: jking  
Date: 6/27/2006 7:37:56 AM  
Type: Note

Thanks for sticking with me through this presentation. As you can tell, I am still fascinated by this technology.

A copy of the most recent version of these slides is available from the hyperlink shown above.

You can click on it!

Jim King - 6/27/2006

