



Chapter 2

Designing for Different Display Modes

In this chapter, you'll learn about:

- ♦ **Video hardware standards**
- ♦ **Display modes**
- ♦ **Screen resolutions**
- ♦ **Aspect ratios**
- ♦ **Refresh rates**
- ♦ **Color capabilities**
- ♦ **Gamma levels**
- ♦ **Choosing display modes to design for**
- ♦ **Specific display mode recommendations**
- ♦ **Rules for display mode selection**

A Summary of Video Hardware Standards

The purpose of this section is to summarize the historical significance of the various video hardware standards that have emerged since the advent of the personal computer in the early 1980s. These standards include:

- Color graphics adapter (CGA)
- Enhanced graphics adapter (EGA)
- Video graphics adapter (VGA)
- Multicolor graphics array (MCGA)
- Super video graphics adapter (SVGA)

Color Graphics Adapter (CGA)

The CGA was one of the first video hardware standards to appear on the market. IBM introduced it in 1981 as a display option for their then fledgling line of personal computers. The CGA standard enabled graphics to be created in color (a major coup at the time) and offered three different levels of graphics resolution: low, medium, and high.

Although technically primitive by today's standards, the CGA was an important milestone in the annals of graphics development. It not only introduced color graphics to the PC-compatible world, but it also helped establish the 320x200-line video mode as an industry screen resolution which was adapted by scores of other systems like the Commodore 64, Amiga, Atari ST, Apple IIGS, and a number of home video game systems.

Virtually all PC graphics systems are backwardly compatible with the CGA standard. This means that they can use software specifically written for any of its video modes. However, it's important to point out that the CGA hasn't been supported by commercial arcade games since the late 1980s and is widely considered to be obsolete.



NOTE: Only DOS continues to support the CGA standard and even that support is limited to running old software.

Enhanced Graphics Adapter (EGA)

IBM introduced the EGA as a replacement for the CGA standard in 1984. The EGA's hardware could emulate the CGA video modes as well as use its own, improved graphics capabilities.

As with the CGA, the EGA's days as a graphics standard were relatively short-lived. The EGA was eclipsed by more powerful graphics standards by the late

1980s. The EGA's main contribution to the evolution of arcade games was the fact that it was very popular among PC game programmers due to its ability to display vibrant color (for that time) and support smooth animation via page flipping. In fact, such PC arcade game classics as *Duke Nukem* made their debut on EGA-compatible hardware.




NOTE: Only DOS and Windows 3.1 continue to support the EGA.

Video Graphics Adapter (VGA)

IBM introduced the VGA in 1987 for use in its higher-end PS/2 systems and as a replacement for the EGA standard. The VGA raised the bar of PC graphics capabilities—not only in terms of graphics resolution but also in its ability to display vivid color. The VGA was the first widely adopted video standard to provide support for analog color monitors, which allowed thousands of colors to be displayed instead of the 16 that were commonplace.

The VGA standard became very popular with game developers due to its ability to display many distinct shades of color on-screen at once. Virtually all arcade games written today are still designed with VGA compatibility in mind.

Although now somewhat dated in terms of features, the VGA standard represents the minimum display capabilities offered by any modern computer system.




NOTE: DOS, Windows 95/98/NT/2000, Linux, and Macintosh systems all support some version of the VGA standard.

Multicolor Graphics Array (MCGA)

The MCGA was a reduced-function version of IBM's VGA hardware standard. It first appeared in 1987 with IBM's introduction of its low-end Model 25 and Model 30 PS/2 computer line.

All MCGA display modes are 100% compatible with those supported by the VGA standard. As a result, MCGA-equipped systems can play almost all of the same games one can with a VGA-equipped system.



NOTE: Only DOS and Windows 3.1 continue to support the MCGA standard. Windows 95/98/NT/2000, Linux, and Macintosh systems do not.

Super Video Graphics Adapter (SVGA)

The SVGA evolved during the early 1990s as a means of taming the market chaos that occurred when different hardware manufacturers attempted to improve on the VGA's core technology without following a common hardware standard.

Video hardware that adheres to the SVGA standard supports additional graphics resolutions as well as the ability to display thousands and millions of colors. At the same time, it is fully backward compatible with all software written for the VGA, EGA, and CGA display standards.

Since its introduction, the SVGA has taken arcade games to a new level by allowing them to support higher resolutions and more colors that users demand, and it remains the current display standard across all computer platforms.



NOTE: DOS, Windows 95/98/NT/2000, Linux, and Macintosh systems support all elements of the SVGA standard.

Display Modes

Display modes, or *video modes*, define how graphic images appear on a given machine. Each video hardware standard supports different display mode capabilities. Meanwhile, every display mode has a number of unique characteristics associated with it. These characteristics determine everything from screen width to the number of colors that can be displayed at a time.

As a designer, it's important to get to know these characteristics and understand how they can influence the artwork you create. Specifically, you should concern yourself with these issues:

- Screen resolution
- Aspect ratio
- Refresh rate
- Color capability
- Gamma level

Screen Resolution

As you probably already know, all computer-generated images are composed of *pixels*, which are the smallest graphic elements that you can manipulate. Screen resolution measures the actual number, or the density of pixels per a horizontal and vertical screen line.

The basic law of screen resolution is this: the higher the screen resolution, the greater the amount of graphic detail and information that can be displayed, as

illustrated by Figure 2-1. In this example, the top figure represents an object rendered at low resolution while the bottom figure represents the same object when displayed at a high resolution.

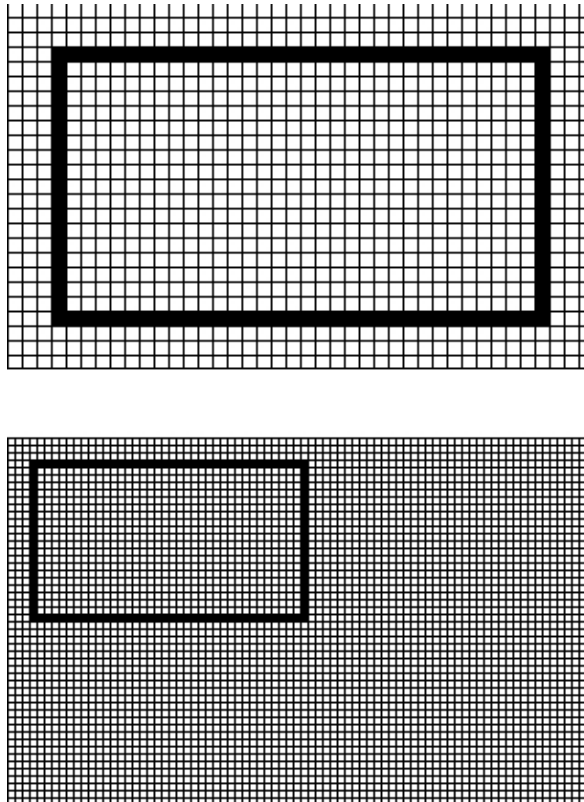


FIGURE 2-1: Screen Resolution Example

Multiplying the number of horizontal pixels (X) by the number of vertical pixels (Y) produces the total number of pixels that are supported at a given screen resolution. For example, a screen resolution that is 640 pixels wide by 480 pixels tall is said to have 307,200 pixels available. Screen resolutions are commonly referred to using this formula, i.e., 640x480.

Why It's Important

Screen resolution is important for two reasons. First, it defines how much detail we can see or create on a given screen. Second, it influences the quality of how we see this information.

The average, unaided human eye can resolve objects as small as .05 millimeters. However, even the most sophisticated computers can't come close to producing

images with that level of detail. Instead, most computer-generated images look blocky and coarse when compared to common print materials such as books or magazines.

When irregular graphic shapes such as diagonal lines or circles are displayed on-screen, their shape and appearance has to be approximated. This approximation produces an unfortunate byproduct called *aliasing*. Aliasing is the stair step-like pattern that gives computer-generated artwork its distinct digital look, as shown in Figure 2-2. Fortunately for us, as the screen resolution increases, so does the density of the pixels available per screen line. This effectively minimizes the impact of aliasing and makes graphic objects appear smoother than they really are.

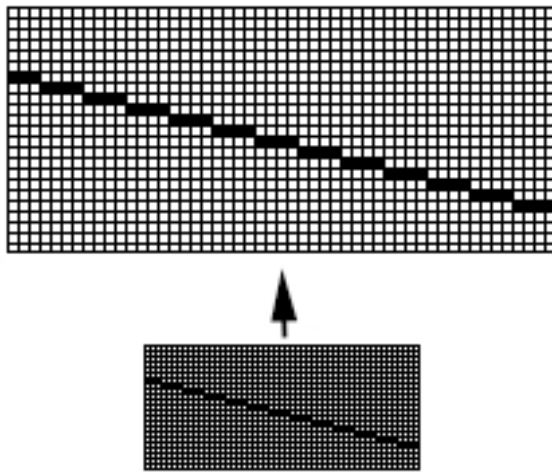


FIGURE 2-2: Aliasing

Screen resolution also has a direct influence on image quality. Using higher screen resolutions will always give your artwork higher fidelity over lower screen resolutions. Figure 2-3 shows some examples of this. Both objects A and B are identical but are rendered at different resolutions. If you look closely, you'll notice that object A looks much coarser than object B due to its lower resolution.



FIGURE 2-3: Screen Resolution and Image Quality

Given the sophistication of today's computers, people will demand and expect games that feature clear and crisp graphics. Don't disappoint them.

Today's computers are capable of displaying a variety of different screen resolutions. Table 2-1 lists the most common of these.

TABLE 2-1: Common Computer Screen Resolutions

Operating System	Resolution Name	Supported Screen Resolution	Total Pixels (X*Y)
Linux	Medium	640x480	307,200
Linux	High	800x600	480,000
DOS	Medium	320x200	64,000
DOS	Mode X	320x224	71,680
DOS	Mode X	320x240	76,800
DOS	Mode X	320x256	81,920
DOS	Mode X	320x480	153,600
DOS	Mode X	360x200	72,000
DOS	Mode X	360x224	80,640
DOS	Mode X	360x240	86,400
DOS	Mode X	360x256	92,160
DOS	Mode X	360x400	144,000
DOS	Mode X	360x480	172,800
DOS	Mode X	640x400	256,000
DOS	VGA	640x480	307,200
DOS	SVGA	800x600	480,000
Mac OS	14" Monitor	640x480	307,200
Mac OS	15" Monitor	800x600	480,000
Windows 3.1	Medium	640x480	307,200
Windows 3.1	High	800x600	480,000
Windows 95/98/NT/2000	Low	320x200	64,000
Windows 95/98/NT/2000	Medium	640x480	307,200
Windows 95/98/NT/2000	High	800x600	480,000

NOTE: The average 17-inch computer monitor can support screen resolutions of 1024x768 but very few arcade-style games actually run in that resolution. Therefore, we won't cover this resolution.

Mode X is a non-standard software modification of the standard VGA display modes. In addition to providing additional screen resolution, Mode X offers improved performance for most graphics operations, including complex arcade game animation. It's interesting to point out that all of the so-called DirectX display modes are also Mode X display modes in disguise. For the purposes of this discussion, only the most commonly used Mode X resolutions are mentioned here. Also, be aware that Mode X resolutions aren't typically available on non-PC

hardware (i.e., the Macintosh) because of differences in video hardware architecture.

Screen Resolution Issues

Failing to take screen resolution into account while designing your arcade game graphics can have disastrous results. Here are three important issues to consider when dealing with screen resolution:

- Graphics production time
- DPI differences
- Platform compatibility

Graphics Production Time

Even experienced designers find that creating high-quality game art takes lots of time, precision, and patience to do well. As such, it's not a process that takes kindly to being rushed. Once you get into a project, you'll often find that it can take you hours or even days to make a particular graphic object look just the way you want it.

This being said, it's important to understand how screen resolution can influence the time it takes you to create your game artwork. These influences include *screen real estate*, *subject matter*, and *comfort level*.

Screen Real Estate

The higher the screen resolution, the more on-screen “real estate” there is for you to contend with. For example, if you were designing in a relatively low screen resolution like 320x200, you would be working with 64,000 pixels. However, if you were designing in a 640x480 resolution, you would be working with 307,000 pixels, and so on. That's a huge increase in on-screen area, 4.8 times more to be exact. This extra screen area means that there's more screen space to fill in with your objects. This in itself can negatively impact how long it takes you to create your graphics.

Subject Matter

The subject matter of your artwork can also make a big difference on how long the graphics creation process takes. For example, very simple artwork, such as those composed of simple shapes, probably won't be affected by screen resolution very much. However, very complex artwork that features intricate shapes and patterns might be affected, as such artwork tends to be more susceptible to visual anomalies that higher screen resolutions can reveal.

Comfort Level

Finally, your comfort level with designing at different screen resolutions can be a big factor on how long it takes you to create graphics for a given project. For

example, an artist comfortable with designing at one resolution may not feel comfortable designing artwork at another. This is because most graphic artists tend to learn how to create game graphics at a specific resolution and, over time, adjust their drawing styles to match the particular quirks and eccentricities exhibited by that resolution. As a result, many artists aren't able to easily adapt to a different screen resolution without a significant amount of retraining and practice. For example, if you're used to drawing your artwork at a resolution of 320x200 and suddenly have to work at a resolution of 640x480 for a project, you may be in for a rude awakening. You'll quickly discover that the pixel size and orientation of this screen resolution has totally changed, potentially throwing you off and slowing you down.

In any case, be sure to consider these issues thoroughly before you tackle a project and make sure you set aside enough time to handle any such contingency. You have been warned!

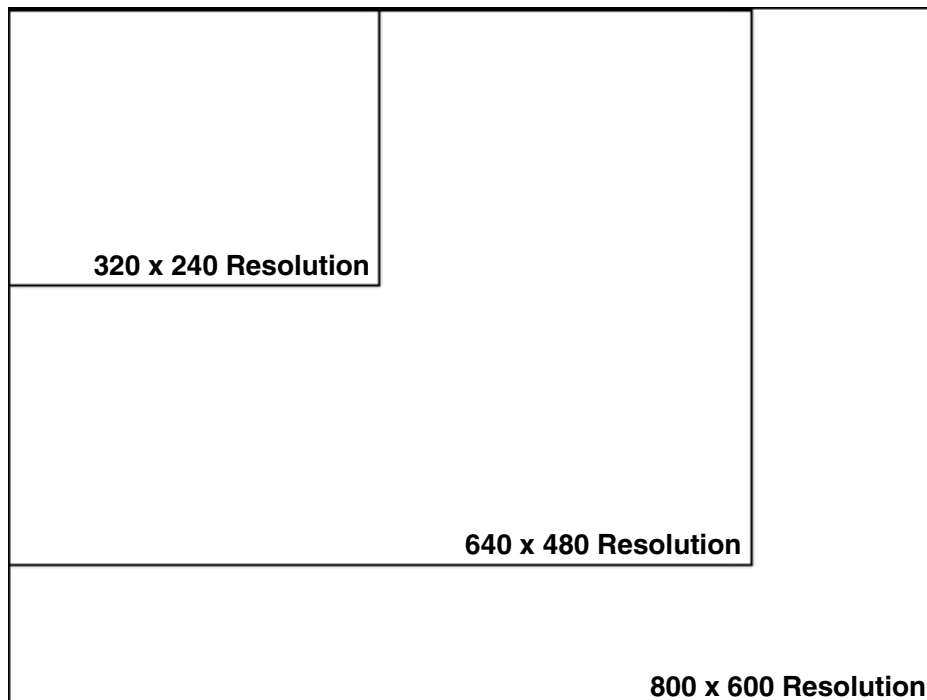


FIGURE 2-4: Comparison of Different Screen Resolution Sizes

DPI Differences

Screen resolution is also sometimes specified as DPI, or dots per inch. DPI is important to consider because it influences how different images display at various screen resolutions and monitor sizes. For example, an image created at a resolution of 640x480 on a 15-inch monitor will appear larger when shown on a

19-inch monitor at a resolution of 640x480. This is because the pixels on the larger monitor are less dense than on the smaller monitor. This has the effect of making your artwork appear blocky in addition to revealing potential design flaws, etc. Table 2-2 illustrates these DPI differences on various monitor sizes.

TABLE 2-2: DPI Differences at Common Screen Resolutions

Screen Resolution	Estimated DPI 14" Monitor	Estimated DPI 15" Monitor	Estimated DPI 17" Monitor	Estimated DPI 19" Monitor
320x240	32	30	25.6	22
360x200	37	34	29	25
640x480	66	60	51	44
800x600	82	75	64	56

NOTE: You can roughly estimate the DPI supported by a given monitor by using the formula X/S , where X is the horizontal screen resolution and S represents the width of the displayable picture of the monitor in inches. For example, a 15-inch monitor with a viewable picture area of 10.6 inches and a screen resolution of 640x480 would yield a DPI of 60.

NOTE: While Windows, the Macintosh, and Linux all have different system DPI specifications of 96, 72, and 75 DPI respectively, these differences only influence text fonts and not bitmapped graphics as used in arcade games. Figure 2-5 shows some examples of how DPI can influence the appearance of graphic images.

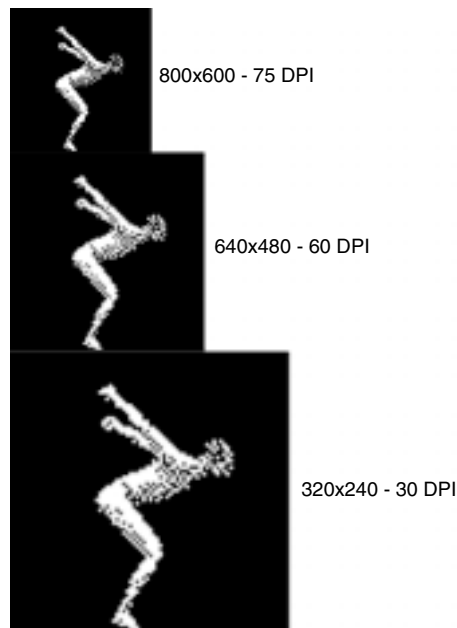


FIGURE 2-5: DPI and the Appearance of Graphic Shapes

Platform Compatibility

Despite the fact that computers are capable of supporting many different screen resolutions, only a few of these resolutions are actually common across multiple platforms. For example, screen resolutions such as 320x200, 640x480, and 800x600 are commonly found on several different systems, while screen resolutions such as 360x200 are platform specific and proprietary.

To play it safe, avoid designing in proprietary screen resolutions unless you have a very specific need to do so. Proprietary screen resolutions tend to complicate matters, especially when trying to port your artwork from one platform to another. Instead, plan on designing your game artwork at those screen resolutions that you know to be compatible with and common across different arcade game platforms.

NOTE: Most of the so-called Mode X screen resolutions are considered to be proprietary and aren't recommended unless you and your development team have made a conscious decision to do so.

A specific screen resolution is considered compatible when it is supported by all major computer platforms as identified in Table 2-3.

TABLE 2-3: Compatible Screen Resolutions across Platforms

Screen Resolution	DOS Compatible	Windows 3.1 Compatible	Windows 95/98/NT/2000 Compatible	Macintosh Compatible	Linux Compatible
320x200	✓	+	✓		
320x240	✓	+	✓		
640x480	✓	✓	✓	✓	✓
800x600	✓*	✓	✓	✓	✓

* Above 640x480 in 16 colors, DOS needs a third-party video driver to support these resolutions.

+ These screen resolutions are not native. DOS programs that use them should run, but Windows 3.1 does not support these resolutions.

What You Can Do

- You can ensure the image quality of your game artwork by always choosing a higher screen resolution over a lower screen resolution. However, don't make the mistake of assuming that using a high screen resolution will always make your game artwork better, because it won't. Using high screen resolutions can be a double-edged sword. It can just as easily make your graphics look worse by revealing subtle (and not-so-subtle) design flaws in your artwork, i.e., sloppiness, poor style, and the like. Also, for performance and compatibility reasons, using a high screen resolution won't always be an option that's available

to you. Fortunately, it happens that many games can work well at lower screen resolutions as well. Refer to Table 2-11 to see some examples of this.

- You can actually speed up the graphics creation time for most game projects by cheating. For example, if your game doesn't require very high-resolution objects, consider creating your graphics using a lower screen resolution instead, i.e., use 320x240 instead of 640x480 or 640x480 instead of 800x600, etc. If the lack of image quality concerns you, don't let it worry you too much. You can often make up for the lack of overall resolution by strategically using color. Color, as we'll see later in Chapters 7 and 8, has the unique ability of being able to simulate resolution through clever shading. What's more, there are many instances where fast animation or certain game styles (i.e., "retro") can minimize the impact of not having very highly detailed graphic objects present. This is because most users psychologically won't perceive the lack of detail once they become engrossed in the game's action.
- Unfortunately, you can't control how different monitors handle DPI at different screen sizes, as there is simply too much variation possible. However, you might be able to anticipate how some systems will render your artwork by testing your artwork at different screen sizes prior to incorporating it into a game. This way, you might be able to make some changes that will minimize the flaws and defects that might be exposed in your artwork on different sized monitor screens.
- To avoid graphics compatibility and porting problems between major arcade game platforms such as the PC or Macintosh, always try to design your artwork using a commonly supported screen resolution. For example, 640x480 is compatible with Windows, Linux, and Macintosh machines, while 320x200 is largely compatible with DOS, Windows, and many dedicated video game consoles (i.e., Super Nintendo, etc.).

Aspect Ratio

An important, but little known fact is that most popular screen resolutions are horizontally oriented and actually appear rectangular when displayed, and not square. Therefore, they are said to have an *aspect ratio*, or the ratio of horizontal pixels to vertical pixels, of 4:3. In other words, the images on these screens are 1.33 times wider than they are tall. We refer to this measurement as the *horizontal aspect ratio* of an image.

Why It's Important

Aspect ratio is important because it helps to ensure that geometric shapes such as circles, curves, and diagonal lines are displayed on the screen correctly. In order to achieve this, however, some screen resolutions, such as those with 4:3 aspect ratios, actually change the shape of their pixels from rectangles to squares. Doing

this keeps whatever appears on the screen looking proportional and consistent. So, for example, a circle drawn on such screens will appear as a circle rather than an ellipse.

Figure 2-6 illustrates how aspect ratio affects screen shapes while Figure 2-7 compares the pixel shapes of different screen resolutions. The figure on the left represents an image rendered at a perfect 4:3 aspect ratio while the figure on the right shows the same image when displayed on a screen that doesn't support a 4:3 aspect ratio.

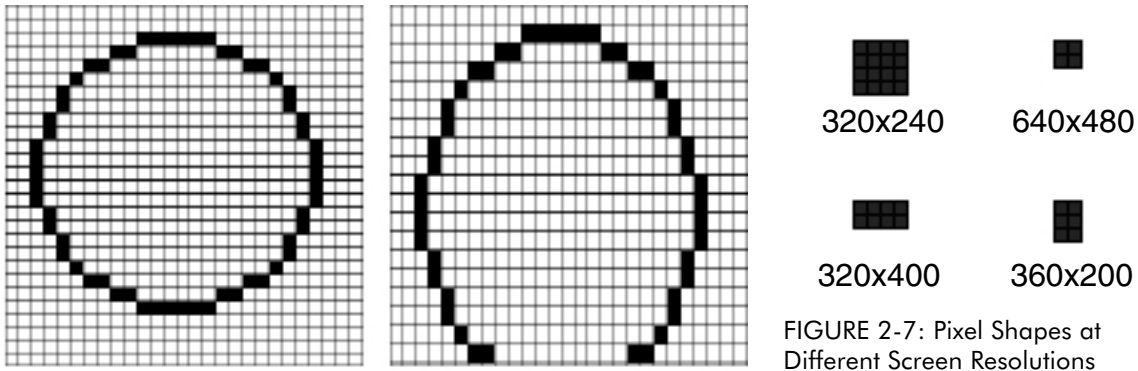


FIGURE 2-6 Aspect Ratio Examples

A display mode that has a 4:3 aspect ratio is considered to have a perfect aspect ratio. However, due to individual system hardware differences and how certain display modes are implemented on some machines, not all screen resolutions can produce perfect 4:3 aspect ratios. Therefore, the shapes drawn in these screen resolutions will appear somewhat distorted.

The significance of pixel shape should not be ignored! As pixels are essentially the building blocks for all graphics, you will find that many types of objects are more difficult to draw and accurately render with non-square pixels than with square pixels. In addition, pixel shape can contribute to the horizontal and vertical distortions some images exhibit when displayed in screen modes with imperfect 4:3 aspect ratios. Keep this in mind should you opt to use a screen resolution that supports non-square pixels.



NOTE: Several of the so-called Mode X, or non-standard, PC display modes use square pixels and have perfect aspect ratios.

Incidentally, you can determine the horizontal aspect ratio for a given display mode by dividing X by 4 ($X/4$) and Y by 4 ($Y/4$), where X is the horizontal screen resolution and Y is the vertical screen resolution. Then, simply divide the results.

For example, to determine the aspect ratio for a screen resolution 800x600 you would apply this formula as follows: $800/4 = 200$ and $600/4 = 150$. Dividing 200 by 150 yields 1.33 or a perfect 4:3 aspect ratio.



NOTE: Programmers sometimes equate the perfect 4:3 aspect ratio to a value of 1.0. According to this scale, those display modes that have aspect ratios that are more than 4:3 or less than 4:3 are considered to be greater than 1.0 and less than 1.0, respectively.

Table 2-4 shows some of the more common aspect ratios supported by different display modes (both Mode X and the standard, compatible modes).

TABLE 2-4: Common Display Mode Aspect Ratios

Screen Resolution	Horizontal Aspect Ratio	Square Pixels	Display Mode Type
320x200	1.6		Mode X
320x224	1.42		Mode X
320x240	1.33	✓	Mode X, Compatible
320x256	1.25		Mode X
320x400	.80		Mode X
320x480	.66		Mode X
360x200	1.8		Mode X
360x224	1.6		Mode X
360x240	1.5		Mode X
360x256	1.4	✓	Mode X
360x400	.90		Mode X
360x480	.75		Mode X
640x400	1.6		Mode X
640x480	1.33	✓	Compatible
800x600	1.33	✓	Compatible

Aspect Ratio Issues

If you've ever wondered why the shape and orientation of identical graphic images sometimes appear differently across systems, you can pretty much bet that aspect ratio is the reason. When a screen's aspect ratio is out of whack, so too will be everything else on the screen.

As such, you're likely to encounter these two problems related to aspect ratio when designing arcade game artwork for different display modes:

- Vertical compression
- Horizontal distortion

Vertical Compression

Graphics that were originally created on a screen that has a larger aspect ratio (i.e., 1.6) and then displayed on a screen with perfect 4:3 (1.33 horizontal) aspect ratio will appear vertically compressed or squashed. For example, this typically occurs when you draw your graphics at a resolution of 320x200 and then display them at a resolution of 320x240.

Horizontal Distortion

Graphics that were originally created on a screen that has a smaller horizontal aspect ratio (i.e., .80) and then displayed on a screen with perfect 4:3 (1.33) aspect ratio will appear horizontally distorted or stretched. For example, this occurs when you draw your graphics at a resolution of 320x400 and then display them at a resolution of 320x240.

In either case, you'll have graphic images that won't look right on-screen. Figure 2-8 illustrates these issues. Object A was created in a display mode with a perfect aspect ratio and square pixels. Object B shows the same image when vertically compressed as it's rendered in a display mode with an imperfect aspect ratio and pixels that are slightly wider than they are tall. Object C shows the same image when horizontally distorted as it's rendered in a display mode with an imperfect aspect ratio and pixels that are much wider than they are tall.

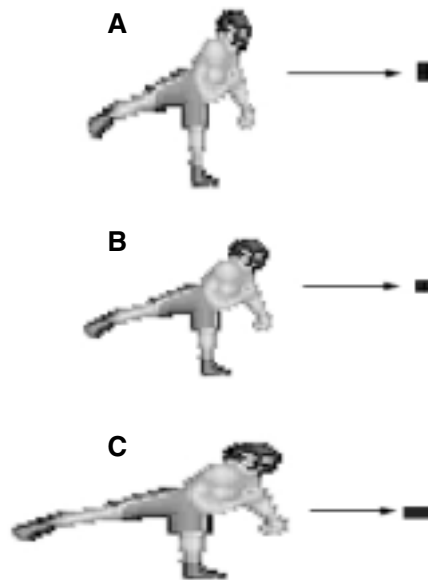


FIGURE 2-8: Example of Vertical Compression and Horizontal Distortion

What You Can Do

- You can minimize the impact of screen aspect ratio on your artwork by designing your graphics as closely as possible to the *target resolution* of your game. For our purposes, the target resolution refers to the resolution that the game will actually run in. For example, if you already know ahead of time (and you usually will) that the artwork for the DOS game you're working on will run in a resolution of 320x240, design your graphics at a screen resolution of 320x240 and not at 320x224 or 320x200, etc. This is just common sense.
- You can reduce aspect ratio distortion by avoiding designing your artwork in display modes that have horizontal aspect ratios that are less than 1.33. Such display modes have problems correctly displaying and maintaining the proportions of rectangular objects such as sprites and background tiles. If, for some reason, you must design for these display modes, try to use them for title and menu screens or for situations where proportion doesn't matter.
- Never try to save time by trying to adjust the aspect ratio of your artwork *after the fact*. By this I mean, don't draw your graphics in a resolution of 320x224 and then try to resize them down to match a screen resolution with a different aspect ratio, such as 320x200. The end results just won't look very good.

Refresh Rate

The *refresh rate*, also known as the frame rate, is a number that measures how fast the electron gun in a computer monitor's CRT (cathode-ray tube) paints an image from the top of the screen to the bottom. Refresh rates are measured in *hertz* (Hz), or cycles per second. For example, a display mode with a refresh rate of 60Hz repaints the display 60 times per second.

Why It's Important

The refresh rate affects how comfortable (or uncomfortable) it is to look at an image on a given screen. Obviously, you are more likely to be productive when you are comfortable than when you are not.

Refresh Rate Issues

- User Fatigue

User Fatigue

Regarding game graphics design, the refresh rate is important to mention for one reason only: user fatigue.

If the refresh rate of the display is too slow, the phosphors that compose the image will fade before the CRT can repaint them. This premature fading creates a phenomenon called *flicker*. Among other things, flicker can induce eyestrain and even headaches if you're exposed to it over a prolonged period of time.

Since creating game artwork and animation requires you to spend a significant amount of time in front of the computer screen, this isn't an ideal situation. Fortunately, you can minimize the effects of flicker by simply upping the screen's refresh rate. That's because higher refresh rates produce much less flicker.

VESA (<http://www.vesa.org>), an association of computer manufacturers that sets standards in the computer industry, recommends that users set their refresh rates to at least 85Hz because eyestrain and related discomfort can be drastically reduced at this level. Incidentally, this is a change from their old recommendation that had previously established 72Hz as the minimum satisfactory refresh rate.

Refresh rates are dependent on two factors: the capabilities of the computer's video hardware and the capabilities of the monitor. Therefore, not all systems or platforms will support the same refresh rate at the same resolution. In general, however, most new monitors come equipped to handle refresh rates of at least 72Hz and more than a few are capable of supporting refresh rates of 85Hz or more as well.

Table 2-5 shows some refresh rates that are typically supported at different display modes.

TABLE 2-5: Common Refresh Rates

Display Mode	Common Refresh Rate(s)
320x200	60Hz, 70Hz, 72Hz, 75Hz, 85Hz
320x224	60Hz
320x240	60Hz
320x400	60Hz, 70Hz
360x200	60Hz, 72Hz
360x240	60Hz
360x400	60Hz, 72Hz
360x480	60Hz
640x400	60Hz, 70Hz, 72Hz
640x480	60Hz, 72Hz, 75Hz, 85Hz, 120Hz
800x600	56Hz, 60Hz, 72Hz, 75Hz, 85Hz, 95Hz, 100Hz, 110Hz, 120Hz



NOTE: Refresh rates are sometimes influenced by external factors like electrical current. For example, it was common for home computers during the 1980s to have their screens refresh their contents at 60Hz in the United States but at 50Hz in Europe due to electrical differences between the two regions.

Table 2-6 shows some examples of different refresh rates and their impact on eye-strain after about three hours of continuous use.


TABLE 2-6: Refresh Rates and Impact on Eyestrain

Refresh Rate	Likelihood of Eyestrain	Comments
< 60Hz	Very High	Usually seen only on older computers.
60Hz-67Hz	High	Common refresh rate supported by many low-end computer monitors and televisions. The default Windows, DOS, Linux, and Macintosh refresh rates.
72Hz-75Hz	Moderate	Supported by most low-end computer monitors and video cards. Very common.
85Hz	Low	Supported by most midrange computer monitors and video cards. Increasingly common.
> 95Hz	Very Low	Supported only by high-end computer monitors and video cards. Still somewhat uncommon.

What You Can Do

To fight user fatigue, follow these suggestions:

- Don't design your graphics on a screen that supports less than a 72Hz refresh rate, especially on larger monitors (i.e., those above 17") as they're more prone to flicker problems than smaller monitors. Although 72Hz isn't as good as 85Hz, it's still a lot better than 60Hz. Of course, if your computer supports refresh rates of 85Hz or above, then take advantage of it. Your eyes will thank you.
- As it turns out, many operating systems default to lower refresh rates (i.e., 60Hz) even though they're capable of supporting much higher ones. For example, Windows always defaults to 60Hz even if your system can support more. However, this situation can usually be corrected by simply adjusting your system's display configuration, i.e., the Display Properties applet under the Windows Control Panel or the Monitors & Sound control panel on the Macintosh.
- It's been scientifically proven that taking frequent breaks can reduce user fatigue. So, for every hour spent staring at a flickering CRT screen, try to spend at least 5-10 minutes looking away from the screen. Close and rest your eyes or simply focus on something else such as what's out your window.



NOTE: DOS systems can't usually adjust their refresh rates unless a third-party utility is used, such as SciTech software's *Display Doctor* or the free *UniRefresh* utility that is included on the book's accompanying CD-ROM. See Appendix B for more information.

Table 2-7 shows the default refresh rates of the different computer platforms.

TABLE 2-7: Default System Refresh Rates

Platform	Default Refresh Rate
DOS	60Hz
Windows	60Hz
Macintosh	67Hz
Linux	60Hz or 67Hz*

* Depends on whether Linux is running on Intel or Macintosh hardware.

- Only use a high-quality computer monitor when designing your artwork since it's more likely to be able to handle the higher refresh rates you'll require than a cheaper, less capable monitor.

Color Capability

Everything around us is composed of color, including what we see on the computer screen. Arcade games, whether they're running on PCs or dedicated video game consoles, are capable of using and displaying an astounding range of colors. In fact, many people consider color use to be one of the most appealing aspects of arcade style games.

All computers understand color as a combination of three primary colors: red (R), green (G), and blue (B), or RGB. These combinations can be manipulated to create any of the millions of colors available in the color spectrum. The average computer display is capable of generating an almost unlimited number of colors. However, under most circumstances, it usually can't show them all at once. As it happens, the actual number of colors that can be displayed at any given time is dependent on the specific hardware capabilities of the system. For example, some machines can display hundreds, thousands, or even millions of simultaneous colors while others can only display a few. In order to compensate for this, the computer scientists developed the concept of the *color palette*. Color palettes are collections of colors from which one or more colors can be selected. Two types of color palettes exist: *physical palettes* and *logical palettes*.

Physical color palettes (also referred to as hardware palettes) contain all of the possible color values that can be generated by a computer's graphics hardware. In many cases, this can be as many as 16,777,216 colors, if not more. Logical palettes

(also commonly referred to as color tables or just palettes), on the other hand, contain only a small portion of the colors that are present in a system's physical palette. They can be best thought of as “windows” into the actual hardware palette. Logical palettes are useful because they make it easy to organize and specify colors when designing graphic images.

The number of colors present in the logical palette is determined by the available *color depth*, or the number of data bits that make up each colored pixel in a given display mode.

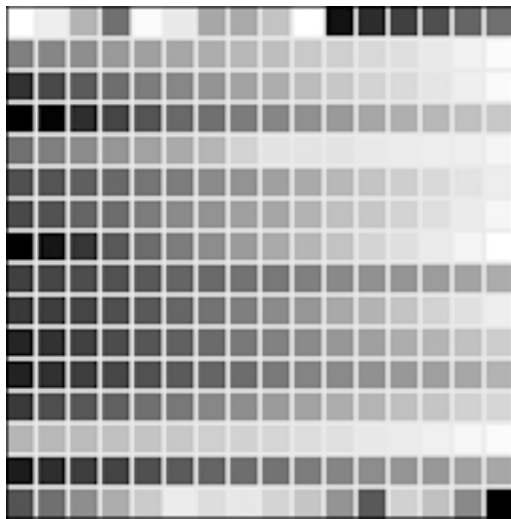


FIGURE 2-9: Logical Palette Example



FIGURE 2-10: Physical Palette Example

Why It's Important

A system's color capability is important because it determines the extent to which you can add color to the images you create. Platforms with limited color capability present more design difficulties and challenges than those that support more extensive color capabilities, etc. These challenges can influence the aesthetics as well as the compatibility of your game artwork.

Refer to Table 2-8 for a list of possible color depth values and the number of colors each can display.

TABLE 2-8: Common Color Depths

<i>Color Depth</i>	<i>Number of Simultaneous Colors in Logical Palette</i>	<i>Common Name(s)</i>	<i>Comments</i>
1 bit/pixel	2	Monochrome, black & white	Previously used by older PCs and Macintoshes for arcade games. Obsolete.
2 bits/pixel	4	N/A	Previously used by older PCs for arcade games. Obsolete.
3 bits/pixel	8	N/A	Used by some older computers. Obsolete.
4 bits/pixel	16	Low color	Used by older computers and dedicated video console systems for arcade game graphics. Largely obsolete.
5 bits/pixel	32	N/A	Used by older computers and dedicated video console systems for arcade game graphics.
6 bits/pixel	64	N/A	Used by older computers and dedicated video console systems for arcade game graphics.
7 bits/pixel	128	N/A	Not used by any computer or video game system.
8 bits/pixel	256	Pseudocolor, Palette mode, indexed color, 8-bit color	Commonly used for arcade-style game graphics. Still very popular with many game developers.
15 bits/pixel	32,768	15-bit high color, thousands of colors	Some systems do not support 15-bit high color. Officially, neither Windows nor the Macintosh does. However, some third-party graphics cards support these modes and provide drivers, etc., on these platforms.
16 bits/pixel	65,536	16-bit high color, thousands of colors	Very common on modern Windows, Linux, and Macintosh systems. Used for a growing number of arcade-style games. Finally starting to replace 8-bit color for modern arcade games due to its higher color fidelity and performance advantages over the 24-bit modes.
24 bits/pixel	16,777,216	True color, millions of colors	Common on modern Windows, Linux, and Macintosh systems.

Logical palettes do not exist in display modes capable of supporting thousands or millions of colors (i.e., 16-bit or 24-bit color depths) as it becomes too difficult and unwieldy to specify colors in these modes.



NOTE: Technically, there is also such a thing as 32-bit true color but it's really just 24-bit true color with a special, 8-bit overlay mode. It's mainly useful for video compositing and for special 3D color effects and has only limited use for most 2D arcade game graphics.

Also, be sure to look at Table 2-9 for more information on the color capabilities supported by various computer platforms.

TABLE 2-9: Common Cross-platform Display Mode Color Capabilities

<i>Platform Screen Resolution</i>	<i>Supported Screen Resolution</i>	<i>Supported On-Screen Colors</i>
DOS Medium	320x200	16 or 256 (thousands or millions possible with VESA support)
DOS Mode X	320x240	256
DOS VGA	640x480	16 or 256 (thousands or millions possible with VESA support)
DOS SVGA	800x600	16 or 256 (thousands or millions possible with VESA support)
Linux Low	640x480	16, 256, thousands, millions
Linux Medium	800x600	16, 256, thousands, millions
Macintosh 14" Monitor	640x480	256, thousands, millions
Macintosh 15" Monitor	800x600	256, thousands, millions
Windows Low	320x200	256, thousands, millions
Windows Medium	640x480	16, 256, thousands, millions
Windows High	800x600	16, 256, thousands, millions



NOTE: The actual number of available colors supported by a particular display mode will depend on the physical capabilities and features of your graphics hardware.

Color Capability Issues

Given the fact that different systems have different color capabilities, it doesn't take a rocket scientist to realize how this might influence how you design your game artwork. Two issues that should be immediately considered are:

- Color vibrancy
- System capabilities

Color Vibrancy

In general, you'll find that images and games that display their graphics using thousands or millions of colors will look noticeably more vibrant and realistic than those that only use a few dozen or a few hundred colors. This is because these extra colors can make objects seem smoother and more defined due to the larger number of shades that are available.

Meanwhile, objects that are rendered using fewer colors will seem coarser and washed out in comparison. On such systems, you'll also need to spend a lot more of your time planning out the colors you'll want to use. In addition, you'll have to learn to make compromises between those objects that you want to render faithfully and those you don't. For example, you'll probably want to apply more color for the game's main character than for background images, etc.

For these reasons, it's advantageous to have more colors at your disposal to design with.

System Capabilities

Unfortunately, basic realities in the form of platform limitations prevent us from always being able to take advantage of such color-rich display modes. This is because a system's color capability is entirely dependent on the hardware it's running on. If the video hardware you're designing on or for doesn't support high or true color display modes, you're out of luck. Just as you can't get blood from a stone, you can't get high color from a system that can't physically support it.

What You Can Do

If you're not able to use a high color display mode in your game, you'll have to make the most of the situation. Here are a few ideas on how:

- You can simulate the presence of more color in an image through clever color selection and special shading tricks. As it turns out, certain color combinations, when used properly, can take advantage of how the average user perceives color so that less color can actually seem like more. We'll explore the issues of user color perception and color selection in more detail in Chapters 7 and 8.
- You can use a technique called *dithering* to simulate the presence of color. Dithering works by blending patterns of two colors to produce a third color much like you would when mixing paint. Through the careful blending of different colors and patterns, you can generate hundreds of additional shades. For example, a display mode that supports only 16 colors can usually generate up to 256 shades through dithering. Meanwhile, a screen with 256 colors can typically generate 65,536 shades and so on. Dithering is commonly used by systems to display images that contain more colors than they normally could display. For more information on dithering, consult Chapter 8.

- You can use high screen resolutions to trick the user into focusing more on the detail and composition of the image rather than its actual color. Images that are exquisitely detailed tend to minimize the need for color. Granted, this technique won't work for every situation, but it does work from time to time.
- You can take advantage of the fact that once a user starts playing a game, he or she tends to focus more on the game's action than on the specific details of the game's objects. This is especially true for most arcade games where fast action forces the user to keep his eyes trained on only small areas of the screen at any one time. This happens because most game objects usually move too fast for the player's eyes to be able to discern more than just passing details of object forms and shapes, etc.

Gamma Level

Gamma level is a measurement that describes the relationship between the color input from the computer's video hardware and how bright it appears on the monitor. The gamma level takes into account several elements of a system, including the tolerances of the CRT, that generate the picture in a computer monitor or TV and the overall color capabilities of the system.

A system's gamma level is usually indicated as a value on a scale between 1.0 and 3.0. As you move the scale towards a gamma level of 1.0, the colors on the screen appear lighter. Conversely, as you move the scale towards a gamma level of 3.0, the colors on the screen become darker.

Why It's Important

Gamma level is important because of how it can influence the appearance of your graphics images in different situation. For example, a properly set gamma level will produce true, realistic looking colors with good reproduction of the light, middle, and dark tones. An improperly adjusted gamma level won't.

Some systems, such as the Apple Macintosh and Silicon Graphics workstations, come with a properly tuned gamma level right out of the box. However, most DOS- and Windows-based PCs do not and will often need to adjust their gamma levels in order to correctly display color on the screen.


Gamma Level Issues

When designing your game graphics, you need to consider issues such as:

- Color accuracy
- Platform-specific gamma differences

Color Accuracy

Gamma level can directly affect an image's *color accuracy*, its overall ability to accurately reproduce color information on the screen. Consider, for example, an identical image of a soldier shown on two systems with different gamma level settings. On one system, the soldier's uniform may appear correctly as khaki while on the other it may show up as brown. As you can appreciate, this issue presents a serious challenge to game graphics designers because they're never really able to determine how the colors they choose will look on different platforms unless they actually test them.



NOTE: Windows 3.1 is much worse at handling gamma levels than any of the other versions of Windows. Whenever possible, avoid creating your graphics on Windows 3.1. DOS also has no inherent ability to adjust its gamma level. Therefore, graphics designed in the DOS environment will definitely appear different (usually darker) than they will on another platform.

Platform-Specific Gamma Differences

Due to system gamma level differences, images will always appear much darker when viewed on a PC than they will on a Macintosh. This is because most PCs have gamma levels between 2.2 and 2.5 while all Macintoshes have a built-in gamma level setting of 1.8. In comparison, gamma levels on most video game consoles will usually be around 2.5, as they tend to use televisions rather than monitors for their displays. This is because virtually all televisions have a fixed gamma level of around 2.5.

The issue becomes even more complicated when you consider that gamma levels can vary even among machines on the same platform! For example, two Windows PCs might have their gamma levels set differently even if they use identical hardware and software.

Table 2-10 illustrates these system-specific gamma differences.

TABLE 2-10: Common Platform Screen Gamma Levels

Platform	Approximate Gamma Level	Built-in Gamma Correction
DOS	2.2-2.5	
Linux	1.8 or 2.2*	
Macintosh	1.8	✓
Windows	2.2-2.5	

* Depends on whether Linux runs on Intel or Macintosh hardware.



NOTE: The gamma levels mentioned in Table 2-10 are approximated because every system varies to some extent based on hardware quality and individual user adjustments.

What You Can Do

Unfortunately, there isn't too much you can do about rectifying this situation until the various computer hardware manufacturers standardize on a gamma level (there are actually proposals on the table that recommend the standard gamma level to be 2.2). In the interim, you can try:

- Using an image-processing program to manually adjust the gamma level for each platform on which you want to display your graphics. For example, if you created your artwork on a system with a gamma of 2.2 and you wanted to maintain the same gamma level on a machine with a gamma level of 1.8, you could use such a program to make this change. Fortunately, this is easy to do and a number of popular graphics programs provide this capability.
- Adding a gamma correction feature to your game. Such a mechanism will allow the user to manually adjust the gamma levels on their machines to suit their particular tastes and preferences. For example, if the game appears too dark, this option can be used to adjust the colors to something lighter and vice versa. Such options are now becoming standard features on today's modern computer games in order to compensate for platform specific gamma differences.
- Calibrating your computer's monitor using color-calibration software. Such software is usually bundled with your system. While this won't guarantee that the color in your artwork will display consistently across different platforms, it can help to ensure the colors you use in your images are at least created accurately in the first place.

It is not possible to compensate for gamma levels if you don't know your current gamma level setting. However, you can determine your system's current gamma level by displaying the image available on the book's accompanying CD-ROM in the GAMMA directory.

Choosing Display Modes to Design For

Every display mode has its own particular advantages and disadvantages. Deciding which display mode to design for isn't always an easy choice to make and it shouldn't be taken lightly. You need to weigh several important issues, including:

- Screen performance
- Image clarity
- Color capability

- System compatibility
- Audience hardware capabilities
- Programming support
- Graphics production time

Screen Performance

When it comes to fast-action arcade games, the issue of screen performance is still far and away the most important consideration when evaluating a display mode. While display modes with higher resolution and color capabilities will definitely enhance the quality of your artwork, they will almost always do so at the expense of drawing speed. This is due to the fact that higher screen resolutions and color modes will consume more of your system's resources (i.e., CPU time, RAM, and disk space) than lower resolutions and color modes will. There's simply more information for the computer to process, store, and manipulate, and hence, more raw computing power is needed to generate the display. As a result, higher resolution and higher color display modes will always display graphics somewhat slower than their lower resolution and color counterparts. So, while your images may look better when rendered in high resolutions and color display modes, other crucial items like playability may ultimately suffer in return for the improvement in aesthetic quality.

Therefore, when considering designing for any display mode you must factor in its performance. Consult with your developers to find the answer to the question: "Will using this particular display mode slow down the game?" If the answer is "yes," you might want to consider finding an alternative display mode.

There are several factors that can influence a particular display mode's screen performance. They include:

- Memory architecture
- Large screen objects
- High color displays
- Screen scrolling
- Programming

Certain display modes, namely the non-standard Mode X ones, offer programmers, and hence games, better performance than more traditional display modes due to their unique memory architecture. Therefore, using such screen modes can often give your games a needed speed boost, especially when performing complex animation.

Games that feature particularly large screen objects exacerbate the problem. This is due to the fact that larger screen objects take longer to draw and animate than

smaller ones. Display modes that feature high resolutions and high color depths are particularly prone to this issue.

Games that display their graphics using lots of color (16-bit or above) can slow the screen display considerably. This is because such display modes require anywhere from two to four times more memory to be manipulated by the computer than those that use 16 or 256 colors. Avoid using such display modes unless they're absolutely necessary and make sure the target platform for your game is fast enough to handle the extra work required.



NOTE: This situation is especially true when it comes to DOS games since many DOS programming environments do not provide optimum video performance without resorting to custom display drivers and/or low-level access to the video hardware. It was also an important consideration with Windows 3.1 and 95 games until recently. Nowadays, most systems ship with very fast video hardware and the drivers for these devices are often pre-optimized for 16-bit color display modes rather than the 8-bit modes. Nevertheless, this is still an issue to think carefully about.

Screen scrolling, especially horizontal scrolling, can cause arcade games to experience major performance problems. This is because the computer must buffer and then move several copies of the current screen in order to produce these effects. While most video hardware is already set up to perform vertically oriented screen scrolling, horizontal scrolling is a different matter altogether. It typically requires special programming which consumes even more of the system's resources. Screen resolution and color depth can also impact screen-scrolling performance. Higher color, high resolution screens need more memory and CPU time to store and scroll the screen than lower color, lower resolution screens do.

Finally, how a game is programmed can have a significant impact on screen performance. While this is usually out of your control, as the game designer/artist, it's important for you to understand how a game is being written. You should know about any restrictions that the game's code imposes way ahead of time in order to avoid any problems later on.

For example, say the game's sprite engine requires that all sprites be 32 pixels wide. If you didn't find this out before you started drawing, you can easily wind up with hundreds of graphic objects that can't be used. So, if you're not sure about the details, ask someone! Close communication with the game's programmer(s) is key to preventing this and other mishaps during the course of the project. This way, you can do whatever you can in order to ensure that the graphics you create are properly optimized prior to their inclusion in the game.

Image Clarity

Image clarity is a subjective measurement that looks at the display mode's ability to show images with sufficient detail and definition. Screen resolution influences image clarity by allowing more information to be placed on the screen.

As a designer, you need to weigh the benefits that the different display modes offer you in terms of image clarity with that of the other factors mentioned here, particularly graphics production time and screen performance.

Color Capability

As discussed, an arcade game's color capabilities play an important role in the user's positive perception of the game. This being said, it's advantageous for us to use as much color as possible in our game artwork.

When possible, you should design your artwork in high color (16-bit and above) display modes, as they will give you the most freedom with regard to your color selections and the type of color effects you can create. However, low color (8-bit) display modes tend to be the most practical choice for the widest range of arcade style games. They offer distinct performance (for the most part) and compatibility advantages over their higher color brethren. Therefore, you should carefully weigh your potential gains with your potential losses when choosing a display mode strictly on its color capability.

System Compatibility

Despite Windows' dominance in the market, designers should not underestimate compatibility with other systems and computer platforms when designing their artwork. This is particularly true with regard to the Macintosh and Linux platforms as they support similar if not identical graphics capabilities and represent a sizeable audience to boot.

By designing your artwork using cross-platform compatible display modes (i.e., non-proprietary), you have the opportunity to port your artwork to other platforms with a minimum of fuss and hassle.

Audience Hardware Capabilities

The actual hardware capabilities of your intended target audience can also be a determining factor regarding which display modes you choose. Many users have video hardware that supports certain screen resolution and color depths. Not all video hardware will support every display mode or display mode feature. This is usually due to limited available video memory. Video memory is used to hold the screen image. The amount of memory required to generate the image depends

primarily on the current screen resolution and color depth being used. This formula calculates how much video memory each display mode requires:

$$\text{Video Memory} = \frac{((X - \text{Resolution}) \times (Y - \text{Resolution}) \times \text{Color Depth})}{(8 * 1,048,576)}$$

Table 2-11 provides examples of the memory requirements needed by different display modes. This will often dictate the capabilities of your audience and ultimately the display mode you choose.

TABLE 2-11: Video Memory Requirements for Common Display Modes

Display Mode	320x200	640x480	800x600
4-bit (16 colors)	256 KB	256 KB	512 KB
8-bit (256 colors)	256 KB	512 KB	1 MB
16-bit (65,536 colors)	256 KB	1 MB	1 MB
24-bit (16.7 million colors)	256 KB	1 MB	2 MB

NOTE: It's important to point out that most video cards in modern systems ship with at least 4 MB of video memory. However, older cards, particularly lower-end cards manufactured in the last four to five years, usually only offer 1 or 2 MB of video memory.

NOTE: Of your potential audience, those users who are still running DOS, Windows 3.1, or pre-Power Macintosh systems are most likely to encounter issues with video memory.

Programming Support

Choosing a display mode isn't always a matter of aesthetics. Sometimes, you may find yourself designing for a given display mode due to its programming support.

Programming support, whether it's through a third-party graphics library or some operating system API, might make using a certain display mode attractive because it's simply easier to develop for when compared to another. Years ago, most game programming was done using custom, hand-coded assembly language routines. Today, however, such practices are often too difficult and time-consuming to implement. Some display modes are more widely documented and supported by programming tools than others. This means that programming considerations may ultimately make the choice of display modes for you.

Graphics Production Time

Unless you're working at your own pace, you'll eventually discover that game development is a very timely business. The market is very competitive and developers are always seeking to be the first out with the latest and greatest game. For example, at ZapSpot we release a new game every two weeks, no matter what. Creating a game's artwork and animation is no easy feat and next to programming the game itself, it's typically one of the longest parts of the game development process.

As such, you need to weigh how each display mode can affect the time you've been allotted to create your artwork for a given project. You'll find that some display modes, especially those with relatively low screen resolution and color capabilities, won't require as much of your time as display modes that support higher resolutions and more colors. So, in effect, your time requirements can make the choice for you.

Display Mode Selection Matrix

Table 2-12 summarizes the overall pros and cons to the various display modes discussed here. The information may help you gain more insight into which different display modes to choose.

TABLE 2-12: Display Mode Selection Matrix

Display Mode	Image Clarity	Performance	Color Capability	System Compatibility	Programming Support	Graphics Production Time
320x200	Low	Fast	16 or 256 (thousands or millions possible with VESA support)	DOS, Windows	Excellent at 16 or 256 colors but tends to be more limited at higher color depths.	Short
320x224	Low	Very Fast	256	DOS	Good	Short
320x240	Low	Very Fast	256	DOS	Excellent	Short
320x256	Low	Very Fast	256	DOS	Good	Short
320x400	Low	Very Fast	256	DOS	Good	Short
320x480	Low	Very Fast	256	DOS	Good	Short
360x200	Low	Very Fast	256	DOS	Good	Medium
360x224	Low	Very Fast	256	DOS	Good	Medium
360x240	Low	Very Fast	256	DOS	Good	Medium
360x256	Medium	Very Fast	256	DOS	Good	Medium

Display Mode	Image Clarity	Performance	Color Capability	System Compatibility	Programming Support	Graphics Production Time
360x400	Medium	Very Fast	256	DOS	Good	Medium
360x480	Medium	Very Fast	256	DOS	Good	Medium
640x400	Medium	Medium	256	DOS	Good	Long
640x480	High	Fast	16, 256, 32,768, 65,536, or 16,777,216	DOS, Windows, Linux, Macintosh	Excellent	Long
800x600	High	Medium	16, 256, 32,768, 65,536, or 16,777,216	DOS, Windows, Linux, Macintosh	Excellent	Long



NOTE: The number of simultaneous colors ultimately influences screen performance more than any other factor, including screen resolution.

Comprehensive Comparison of Display Mode Attributes

Table 2-13 provides a comprehensive comparison of the different display mode attributes offered by the most commonly used display modes on different computer platforms.

TABLE 2-13: Comprehensive Comparison of Display Mode Attributes

Platform Screen Resolution	Supported Screen Resolution	Horizontal Aspect Ratio	Supported On-Screen Colors	Approximate Gamma Level	Common Refresh Rates
DOS Medium	320x200	1.6	16 or 256 (thousands or millions possible with VESA support)	2.2-2.5	60Hz, 70Hz, 72Hz, 75Hz, 85Hz
DOS Mode X	320x224	1.4	256	2.2-2.5	51Hz, 60Hz
DOS Mode X	320x240	1.33	256	2.2-2.5	60Hz, 70Hz
DOS Mode X	320x256	1.25	256	2.2-2.5	58Hz
DOS Mode X	320x400	.80	256	2.2-2.5	60Hz, 70Hz
DOS Mode X	320x480	.66	256	2.2-2.5	60Hz, 70Hz
DOS Mode X	360x200	1.8	256	2.2-2.5	72Hz
DOS Mode X	360x224	1.6	256	2.2-2.5	51Hz
DOS Mode X	360x240	1.5	256	2.2-2.5	61Hz
DOS Mode X	360x256	1.4	256	2.2-2.5	57Hz

<i>Platform Screen Resolution</i>	<i>Supported Screen Resolution</i>	<i>Horizontal Aspect Ratio</i>	<i>Supported On-Screen Colors</i>	<i>Approximate Gamma Level</i>	<i>Common Refresh Rates</i>
DOS Mode X	360x400	.90	256	2.2-2.5	72Hz
DOS Mode X	360x480	.75	256	2.2-2.5	61Hz
DOS Mode X	640x400	1.6	256	2.2-2.5	70Hz, 72Hz
DOS VGA	640x480	1.33	16 or 256 (thousands or millions possible with VESA support)	2.2-2.5	60Hz, 70Hz, 72Hz, 75Hz, 85Hz
DOS SVGA	800x600	1.33	16 or 256 (thousands or millions possible with VESA support)	2.2-2.5	56Hz, 60Hz, 70Hz, 72Hz, 75Hz, 85Hz
Linux Low	640x480	1.33	16, 256, thousands, millions	2.2	60Hz, 70Hz, 72Hz, 75Hz, 85Hz
Linux Medium	800x600	1.33	256, thousands, millions	2.2	60Hz, 70Hz, 72Hz, 75Hz, 85Hz
Macintosh 14" Monitor	640x480	1.33	256, thousands, millions	1.8	67Hz, 72Hz, 75Hz, 85Hz, 95Hz, 100Hz, 120Hz
Macintosh 15" Monitor	800x600	1.33	256, thousands, millions	1.8	67Hz, 72Hz, 75Hz, 85Hz, 95Hz, 100Hz, 120Hz
Windows Low	320x240	1.33	256, thousands, millions	2.2-2.5	72Hz
Windows Medium	640x480	1.33	16, 256, thousands, millions	2.2-2.5	60Hz, 70Hz, 72Hz, 75Hz, 85Hz, 95Hz, 110Hz, 120Hz
Windows SVGA	800x600	1.33	16, 256, thousands, millions	2.2-2.5	60Hz, 70Hz, 72Hz, 75Hz, 85Hz, 95Hz, 100Hz, 120Hz



NOTE: Performance rating is a fairly subjective measurement as different systems and hardware configurations tend to drastically influence these results. Both the Windows and Linux results assume optimized graphics drivers are being used. The examples given in Table 2-13 are taken from my own experiences. Your own conclusions and mileage may vary from mine.



NOTE: Apple Macintosh systems only recently standardized on 800x600 display modes. Prior to the mid-1990s, the Macintosh series actually used a screen resolution of 832x624. However, though a sizeable number of Macintosh systems still support this display mode, it's so close to 800x600 that any differences from the designer's standpoint are negligible.

Specific Display Mode Recommendations

Table 2-14 summarizes my display mode recommendations.

TABLE 2-14: Recommendations for Compatible Display Modes

Category	320x200/Mode X Display Modes	640x480 Display Modes	800x600 Display Modes
Image clarity and performance	Use when performance requirements supersede image clarity.	Use when you want to emphasize a game's image clarity over its performance.	Use when your game requires the best possible image clarity and when performance is important but a secondary concern.
Platform compatibility	Generally recommended for DOS games or when you need to maintain compatibility with video game consoles, etc.	Generally recommended only for designing graphics for Windows, Linux, or Macintosh games.	Use only when designing graphics for Windows, Linux, or Macintosh games.
Graphics development time	Use when designing and graphics production time is limited.	Use when you have more time available to do graphics design and production.	Use when you have more time available to do graphics design and production.



NOTE: Although I recommend designing your artwork in a 640x480 display mode, there's really no significant penalty if you choose to do so at 800x600. Besides offering more screen area to work with, 800x600 display modes share the same screen aspect ratios as the 640x480 display modes. So, if designing in a 800x600 display mode is more comfortable for you, please do so. I use 800x600 extensively for all of the game artwork I design.

Display Mode and Arcade Game Sub-Genre Recommendations

The final assessment of a display mode is its usefulness in rendering the graphics for the different types of arcade games. Table 2-15 summarizes which display modes work best with which games. Please use this table as a rough, general guide. Due to technical limitations, you'll often find that some display modes work better for certain types of games than others do.

TABLE 2-15: Display Modes and Arcade Game Recommendation Matrix

<i>Platform</i>	<i>Display Mode</i>	<i>Color Support</i>	<i>Recommended Game Type(s)</i>
DOS	320x200	16, 256	Puzzlers
DOS	320x240	256	Pong games, shooters, maze/chase games, puzzlers, and platform scrollers
DOS	640x480	256	Pong games, shooters, maze/chase games, puzzlers, and platform scrollers
Windows*	320x240	256	Shooters, maze/chase games, puzzlers, and platform scrollers
Windows	640x480	256, thousands	Pong games, shooters, maze/chase games, puzzlers, and platform scrollers
Windows	800x600	256, thousands	Pong games, puzzlers, and platform scrollers
Linux	640x480	256, thousands	Pong games, shooters, maze/chase games, puzzlers, and platform scrollers
Linux	800x600	256, thousands	Pong games, puzzlers, and platform scrollers
Macintosh	640x480	256, thousands	Pong games, shooters, maze/chase games, puzzlers, and platform scrollers
Macintosh	800x600	256, thousands	Pong games, puzzlers, and platform scrollers

* Denotes not compatible with Windows 3.1.

Arcade Game Type Recommendation Explanations

- **Pong games**—Pong games work well across all display modes due to the fact that they don't usually have more than a dozen or so objects on-screen at any one time. In addition, these games do not usually scroll the screen. As previously mentioned, screen scrolling can place a major drain on arcade game performance.
- **Puzzlers**—Puzzlers work great across all display modes and platforms because they don't require a significant amount of the computer's resources. Such games feature minimal animation and often no scrolling.
- **Maze/chase games**—Maze/chase games work well on most display modes. Because they can display many simultaneous on-screen objects and can employ scrolling, they aren't usually suited for high resolutions such as

800x600 and above. However, they are pretty safe for most other resolutions with few, if any, performance related problems.

- **Shooters**—Like maze/chase games, shooters work well on most display modes. However, as they tend to make extensive use of screen scrolling and can display dozens, if not hundreds, of on-screen objects, they shouldn't be used on resolutions above 640x480 except on fast machines.
- **Platform scrollers**—Platform scrollers are the most sophisticated and resource-intensive of all arcade-style games. They make extensive use of horizontal (and often vertical) scrolling and often display large, complex objects and backgrounds. As a result, they are largely unsuitable for display modes with resolutions above 640x480 except on relatively fast machines.

Rules for Display Mode Selection

As you have seen, choosing a display mode for which to design your artwork and animation can be a difficult and often complex process. There are many issues to consider and weigh. To help make the selection process somewhat easier, try following these rules:

- **Check with your programmer**—Always check with your programmer or programming team. When it comes to technical matters, whether you're talking about the development environment, platform capabilities, or programming tools, the programmer usually knows best. Make sure that you discuss your thoughts on the matter with them and get their input before you start drawing anything. As they're the ones who will have to make your graphics actually work in a game, they're in the best position to tell you what can and can't be done in a given display mode as well as disclose any other related technical issues you may face. When in doubt about some technical issue, never guess or make assumptions. Play it safe and consult a programmer.
- **Consider the target platform(s)**—Always consider the capabilities of the target platform for your game. Different systems have different capabilities. You need to look at these and determine if the target platform can handle what you have in mind. When in doubt, go for the lowest common denominator. For example, although 16- and 24-bit color is supported by a large number of computers, all machines support 256 colors. If you're not sure about what percentage of your audience has hardware capable of supporting more than 256 colors, then stick with 256 colors.
- **Factor in your color requirements**—Take a good look at the color requirements of your game and the particular color capabilities of each display mode that is available to you. Be sure to evaluate issues such as whether or not a given display mode can effectively support the number of colors your game is likely to need. When in doubt, go for the next highest display mode. For

example, if you're designing an arcade game and think that you'll need to use more than 256 colors, go for a display mode that supports 65,536.

- **Factor in your image quality requirements**—Each arcade game has different resolution requirements. Some require the maximum detail possible while others can get away with a relatively low screen resolution. Therefore, you need to evaluate the specific resolution needs of your game and determine the most suitable display mode from that. You should also carefully examine the aspect ratio of each display mode. Will designing your graphics in a display mode with unusual aspect ratios create any problems for you? If so, consider using a different display mode. When in doubt, choose a screen resolution that is common across several platforms (as described in Table 2-3) as one or more of these will provide adequate image fidelity, cross-platform compatibility, and reasonable screen performance.