A 2D Game Engine Using Bitmaps

Rahul Deore[1], Prathamesh Mujumdar[2], Rohan Gulik[3],
Prof. Savita Sangam [4]

1 Rahul.deore@hotmail.com, 2 prathamesh.22@gmail.com , 3 rohan.gulik@hotmail.com, 4 savita.sangam@gmail.com Associate Professor and H.O.D (IT),
Department of Information Technology, Shivajirao S. Jondhale College of Engineering, Dombivali (E).

ABSTRACT

Game engine is a software system designed to make the game development easier, quicker and more efficient. It consists of highly reusable components and pieces of codes that are used over and over again to reduce the cost and time of development. [1]

In this paper we present tools which collectively be called as game development tools or game engine that comprises Microsoft's revolutionary technology, DirectX. The main aim of the presented work is to make game development simpler, easier and powerful. The tools presented here allow developing a full fledged game in DirectX 2D graphics by doing some artwork and integrating and arranging the artwork as a game level. The main advantage of using this engine is that it allows the game developer to concentrate on functional aspect rather than internal implementation.

Keywords – **Sprite, Sprite Editor, Game Engine, 2D Game, DirectX Game, Direct Draw, Game development tool.**

1. INTRODUCTION

A sprite is a graphical character in any two – dimensional video game. Sprites are little images that can be used in a video game. We can draw characters with movement, intros, textures, patterns, backgrounds, logos, color palettes, etc. which can be a plane, man or any other entity that serves the purpose of going through the game on various levels, a character which can be moved on the video screen in games. This tool uses image editing and formatting concept that is more eye catching and attractive.

Sprite editor is a tool to draw sprites with drawing, flipping, coloring, and shifting functionalities for the sprite. The use of DirectX facilitates the game's look and feel more professional and powerful. This tool is bundled with a Tile editor which is a game environment drawing tool. A tile editor will allow creating tiles of different properties to be used at different places in a game. There's a third tool called level editor to design a level for the game. This uses tiles from the tile editor and facilitates functions to place and remove the tile in a level design. In all there are 1920 horizontal pixels and at a time the screen shows 640 pixels scrolling the screen horizontally one pixel at a time after the sprite have reached to the center.

Game engine developers attempt to "pre-invent the wheel" by developing robust software suites which include many elements a game developer may need to build a game. Most game engine suites provide facilities that ease development, such as graphics, sound, physics and AI functions. These game engines are sometimes called "middleware" because, as with the business sense of the term, they provide a flexible and reusable software platform which provides all the core functionality needed, right out of the box, to develop a game application while reducing costs, complexities, and time-to-market—all critical factors in the highly competitive video game industry. [2]

2. GAME DEVELOPMENT TECHNOLOGIES

2.1 Direct Draw

DirectDraw is part of Microsoft's DirectX API. DirectDraw is used to render graphics in applications where top performance is important. DirectDraw also allows applications to run full screen or embedded in a window such as most other MS Windows applications. DirectDraw uses hardware acceleration if it is available on the client's computer. DirectDraw allows direct access to video memory, hardware overlays, hardware blitters, and page flipping. Its video memory manager can

manipulate video memory with ease, taking full advantage of the blitting and color decompression capabilities of different types of display adapters.

DirectDraw is a 2D API. That is, it contains commands for 2D rendering and does not support 3D hardware acceleration. A programmer could use DirectDraw to draw 3D graphics, but the rendering would be slow compared to an API such as Direct3D which does support 3D hardware acceleration.

As of DirectX version 8.0, DirectDraw was merged into a new package called DirectX Graphics, which is really just Direct3D with a few DirectDraw API additions. DirectDraw can still be used by programmers, but they must use older DirectX interfaces (DirectX 7 and below). As of the release of the June 2010 DirectX SDK package, the DirectDraw header file and library are no longed included.

DirectDraw has been deprecated since version 7 but is still included with DirectX, although updates are no longer made. DirectDraw is now replaced by Direct2D.

The DirectDraw Surface file format, from Microsoft, is a standard for storing data compressed with the lossy S3 Texture Compression (S3TC) algorithm, which can be decompressed in hardware by GPUs and consoles like the Playstation Portable, PlayStation 3 and Xbox 360. This makes the format useful for storing graphical textures and cubic environment maps as a data file, both compressed and uncompressed. The Microsoft Windows file extension for this data format is '.dds'.

2.2 Component Object Model

Component Object Model (COM) is a binary-interface standard for software component introduced by Microsoft. It is used to enable interprocess communication and dynamic object creation in a large range of programming languages. The term COM is often used in the Microsoft software development industry as an umbrella term that encompasses the OLE, OLE Automation, ActiveX, COM+ and DCOM technologies. COM is an interface technology defined and implemented as standard only on Microsoft Windows and Apple's Core Foundation 1.3 and later plug-in API that in any case implement only a subset of the whole COM interface. For some applications, COM has been replaced at least to some extent by the Microsoft .NET framework, and support for Web

Services through the Windows Communication Foundation (WCF). Different component types are identified by class IDs (CLSIDs), which are Globally Unique Identifiers (GUIDs). Each COM component exposes its functionality through one or more interfaces. The different interfaces supported by a component are distinguished from each other using interface IDs (IIDs), which are GUIDs too.

COM interfaces have bin\dings in several languages, such as C, C++, Visual Basic, Delphi, and several of the scripting languages implemented on the Windows platform. All access to components is done through the methods of the interfaces. This allows techniques such as inter-process, or even inter-computer programming (the latter using the support of DCOM).

2.3 Game Engine

A game engine is a system designed for the creation and development of video games. There are many game engines that are designed to work on video game consoles and personal computers. The core functionality typically provided by a game engine includes a rendering engine ("renderer") for 2D or 3D graphics, a physics engine or collision detection (and collision response), sound, scripting, animation, artificial intelligence, networking, streaming, memory management, threading, localization support, and a scene graph. The process of game development is often economized, in large part, by reusing/adapting the same game engine to create different games. A typical game engine is shown in Fig.1

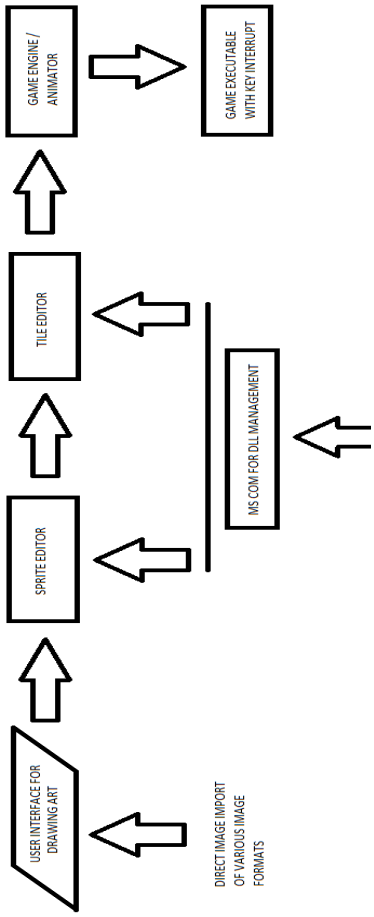


Fig. 1 Game Engine

This diagram shows component object model as a middleware that holds the game components such as sprite, level etc. together using the dynamic link libraries. There is a image and sound storage maintained separately. The output is game that is controlled using the specified keys.

Game engines provide a suite of visual development tools in addition to reusable software components. These tools are generally provided in an integrated development environment to enable simplified, rapid development of games in a data-driven manner. Game engine developers attempt to "pre-invent the wheel" by developing robust software suites which include many elements a game developer may need to build a game. Most game engine suites provide facilities that ease development, such as graphics, sound, physics and AI functions. These game engines are sometimes called "middleware" because, as with the business sense of the term, they provide a flexible and reusable software platform which provides all the core functionality needed, right out of the box, to develop a game application while reducing costs, complexities, and time-to-market—all critical factors in the highly competitive video game industry. Gamebryo and RenderWare are such widely used middleware programs.

The game engine described in this paper combines the output of tile editor, sprite editor, and level editor together. This whole gives a game as output. The game engine tool allows fast and efficient development of 2 Dimensional games using highly reusable components. The game engine developed includes sprite editor, tile editor, and level editor. The details of their components are explained in the following section.

2.4 Sprite Editor

A sprite editor tool is a part of game engine where a canvas is supplied to draw artwork corresponding to sprite for an intended game. This tool is capable of drawing sprites, (prime characters of game), drawing frames or movements for a sprite and other game characters. There are 12 frames for a sprite supported that will make the sprite movements smooth. This tool converts the drawn sprite into a bitmapped file and plays the bitmaps one after the other creating a feel of motion. This is done using the animation engine which is the highly reusable component in the sprite editor. Any bitmap input will be output as sprite. This is illustrated in Fig.2.

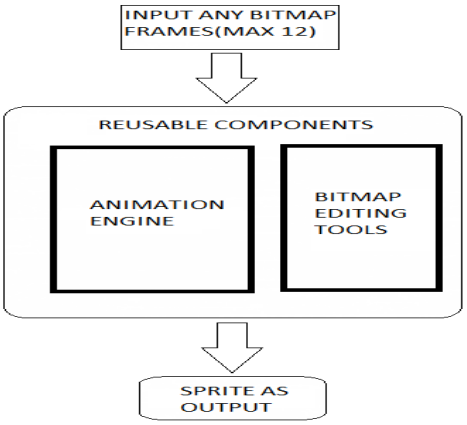


Fig. 2 Sprite editor overview

The above figure explains the working of sprite editor. It consists of an animation engine and bitmap editing tools that are highly reusable components of sprite editor. The input to them is bitmap images that are animated and output as sprite.

The figure 3 shows the Sprite drawn on a canvas of the sprite editor. The image here is showing staircase effect and checkerboard effect since the canvas elements are plotted larger for ease of drawing the artwork. The actual versions of the sprites will be smaller than the drawn one and hence it will resemble smoother sprite. The one that is used in the game is shown in figure 3.1 and figure 3.2

Fig.3 Sprite Drawing On Canvas

Fig.3.1 Game Sprite

Fig.3.2 Downsized Sprite

2.5 Tile Editor

Tiles form another huge part of the software. Sprite is a moving character whereas tile form the stationary part on which movement can be performed. There are screen movements that constitutes the sprite is advancing in the current level towards the goal. While the sprite is moving on a tile, the tile should considerably redraw the commencing part. This makes the tile steady with the trailing part gradually hiding inside the screen (actually getting rubbed) and the leading part to be drawn is drawing function giving a feel that the further part is getting revealed.

Tile editor tool will allow creating tiles of different properties to be used at different places in a game like solid on bottom, solid on top etc. Tile editor basically used for creating the full background environment in the game.

When a user has created an image it can be tiled across the screen of your window. It means that the particular image will be one after the other and one on top of the other across the screen. This will give us an impression of tiles across the screen.
It is particularly used for providing a background to the game screen and giving a visual delight to the player in the current gaming spree.

2.6 LEVEL EDITOR

A level editor will allow designing game levels. Level constitutes environment where the sprite will interact, the goal, other interacting sprites and bonuses to the sprites. All these thing can be designed in a level editor. The interface of level editor is simple and allows selection of tiles that we want to place in the level.

Level editor tool is created to design a level for the game. This uses tiles from the tile editor and

facilitates functions to place and remove the tile in a level design. We have designed a horizontal scrolling for level that will make horizontal movements for discovering whole level.

In all there are 1920 horizontal pixels and at a time the screen shows 640 pixels scrolling the screen horizontally one pixel at a time after the sprite have reached to the center**.**

3. RESULTS

The layout shown in Fig.4 common to sprite editor as well as tile editor. The absence of animate button and presence of tile property is the only difference in the tile editor. Moreover the tile is of smaller size 32 bit by 32 bit.

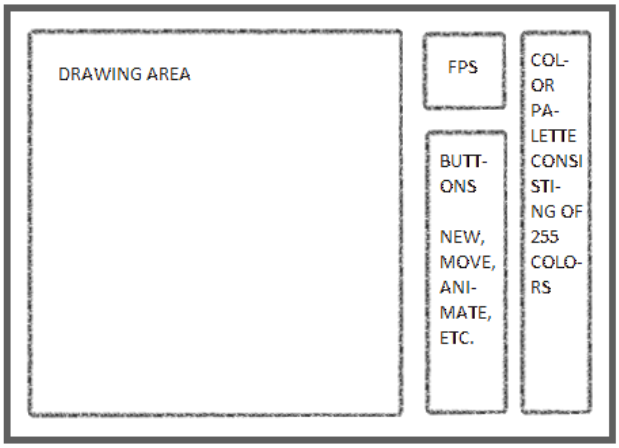


Fig. 4 Sprite/Tile editor User Interface

The Sprite Editor we have made looks like this

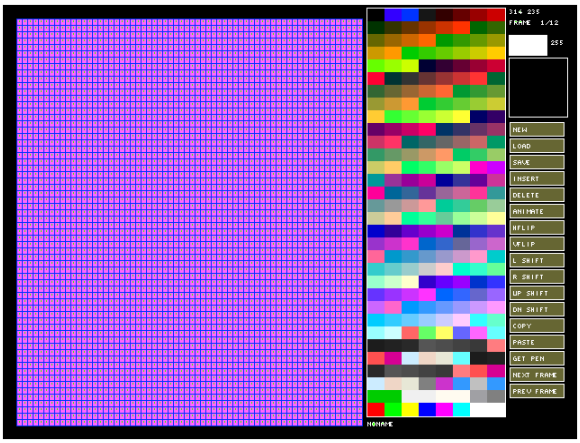


Fig. 5.1 Sprite Editor Interface

The left part in the above figure shows the drawing canvas, to its right is the color palette, and at the extreme right are the frame indicator, color chosen, sprite demo and buttons. These buttons are special tools to make changes to the drawn sprite. The Shift button will allow shifting the sprite left, right, up, down. The flip will allow flipping the object vertically or horizontally. Copy and past are for the whole canvas. These buttons are common to Tile editor also.

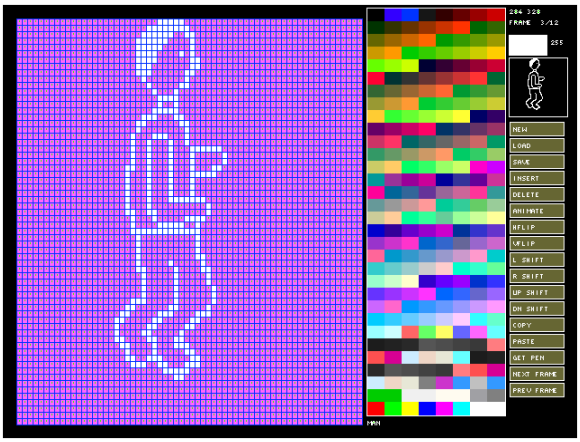
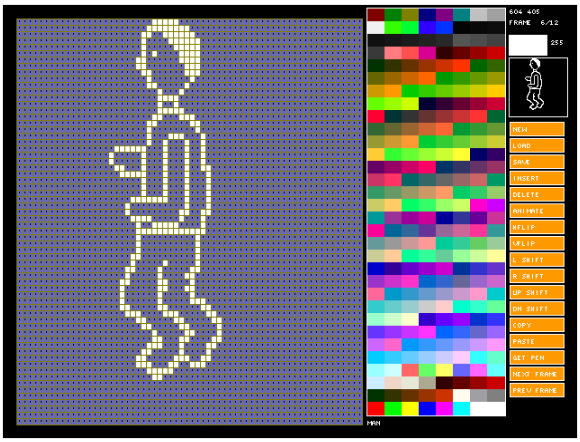


Fig 5.2 Sprite Editor Demo

The above figure demonstrates a sample sprite.



This figure shows the feature vertical flip. The vertical flip allows the sprite to change the direction. This way you do not need to create a new image on the canvas, just vertical flip the image you have created.

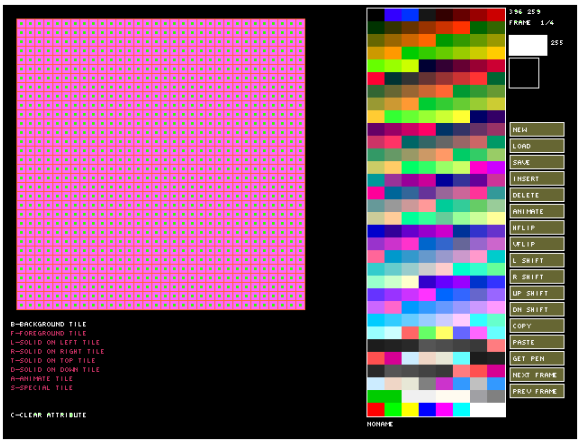


Fig 5.3 Tile Editor with tile properties

The tile editor consists of 32 bits for the drawing bitmap. It has same features and functionality as the sprite editor with some added tile properties. If a sprite has to be shown standing on a tile, it must be a solid tile. The solid tile properties are solid on top, if the sprite is standing on it, solid on bottom, solid on left, solid on right. All these properties can be selected if the tile is expected to have all of them. There are some more properties also including background tile, foreground tile, animated tile, and special effect tile. Each of them can be selected if they do not conflict logically.

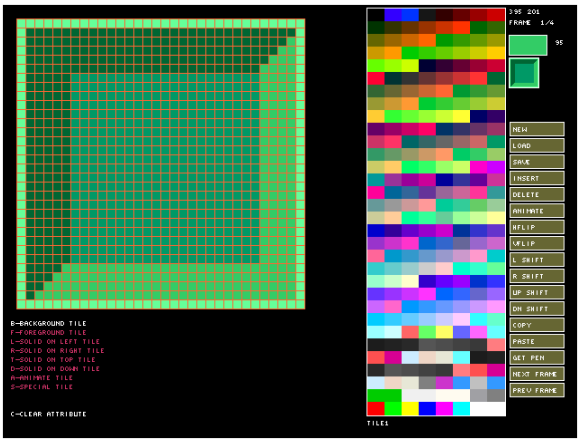


Fig. 5.4 Sample tile graphics drawn.

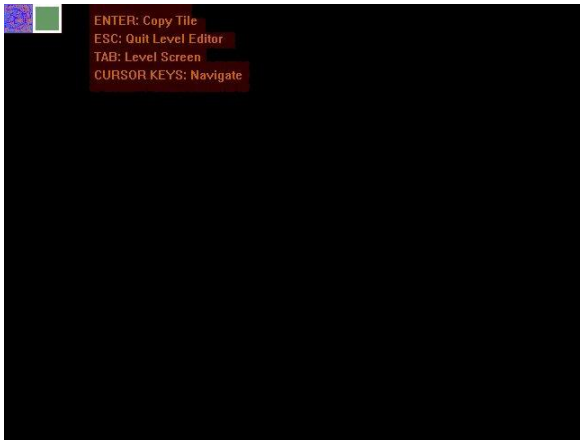


Fig. 5.5 Level Editor Interface

The Level editor screen here shows the two tiles that have been made in the tile editor. You can select a tile by pressing the enter key and tab to switch between level screen and level editor. By doing so you get in to the level screen, where tile can be pasted using enter key and deleted using the del key on the keyboard. F2 key is used to save the level created here.

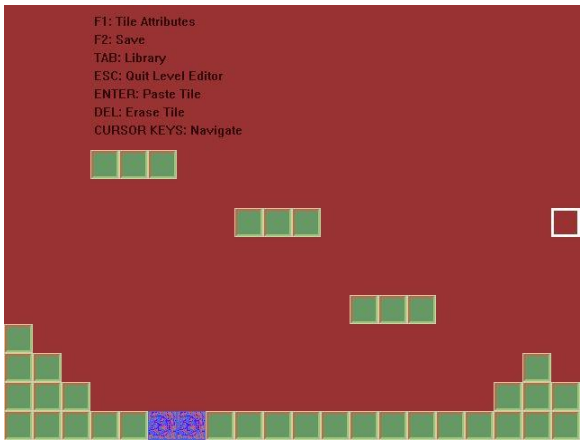


Fig. 5.6 Sample Level Drawn.

Fig. 5.7

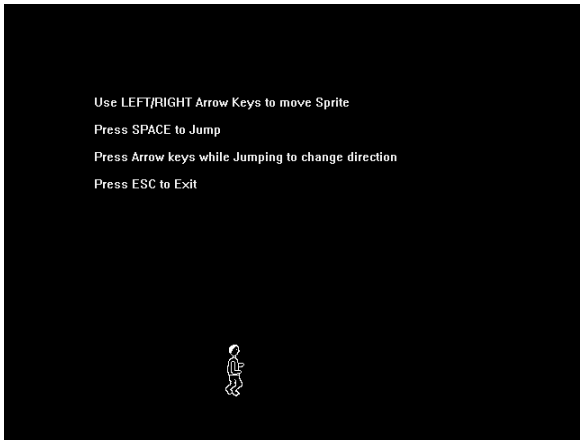


Fig. 5.8

Fig. 5.9

Fig 5.7-5.9 shows a sample sprite animated in blank environment. Here the sprite is moving in left and right direction, and showing jump effect by holding the space bar and direction key simultaneously.

5. Conclusion

Learning game programming and animation has never been so easy. The commercially available software's such as Autodesk MAYA are extremely high in class and learning them is very expensive in terms of time and money.

The basic concept behind development of this tool was to create a simple tool using which the beginner in the gaming and animation area can easily learn the gaming concepts and game development. This tool allows user to efficiently develop 2D games.

Our game engine is simple, easy to use and allows faster and efficient game development using the reusable components provided in it.

REFERENCES

[1] Radha Shankarmani, Saurabh Jain, Gaurang Sinha, *Game architecture and programming wiley, India.*

[2] Wikipedia, http://www.en.wikipedia.org/wiki/game_engine

[3] *Game Architecture and Design*, Andrew Rollings & Dave Morris

[4] *Professional Game Programming*, Mike McShaffry, Dreamtech Press

[5] *Game Programming*, Andy Harris, Wiley India