U.Rahamathunnisa, S. Pragadeeswaran / International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 www.ijera.com
Vol. 2, Issue 3, May-Jun 2012, pp.959-963

COLLISION DETECTION GAME USING COCOS2DX-A CROSS PLATFORM

U.Rahamathunnisa[1], S. Pragadeeswaran[2]
*Assistant Professor, SITE, VIT University, Vellore.
**MS(SE) Student, SITE, VIT University, Vellore.

Abstract
This paper discusses an interactive third person single user game which works on different platform like iPhone and android. Generally 2D games are developed in the win32 platform using cocos2D-x game engine and the objective is to port the game to different platforms like iPhone and android. The development environment is Visual Studio. There are several games existing in the real world like doodle jump, twee jump and this game will be unique in its own way.

Keywords:- *Cross platform, collision detection, sprite, Rendering*

1. INTRODUCTION
According to Mr.Sawyer,playing games is a problem solving activity and they solve social problems of entertainment[1].Developers prefer to use cross platforms for game development[2].The comparisons of mobile game development environment in swerve studio and xforge has been discussed[3].Collision detection is an essential part and it ensures that game physics are relatively realistic. The two main parts in collision detection are detecting whether or not a collision has happened, and if so, responding to the collision. Discovering if a collision has occurred is the basis of this problem.

While responding to the collision is computationally much easier than discovering a collision, it can still pose several problems in how objects are going to react to each other. Developers prefer to use cross platform tools to develop their software to ensure that their products run in as many hardware platforms are available[4].Interpreted languages provide more control over the user interface[5][6].

2. GAME DEVEOPMENT OVERVIEW
The collision detection game has been developed for the users not only for entertainment but it incorporates the problem solving techniques such as laws of physics.The game is a cross platform game developed with features in box 2d.The important part of the game is the collision detection and collision response.

2.1 BOX -2D FEATURES

- Continuous collision detection
- Contact callbacks: begin, end, pre-solve, post-solve
- Convex polyons and circles.
- Multiple shapes per body
- One-shot contact manifolds
- Dynamic tree broadphase
- Efficient pair management
- Fast broadphase AABB queries
- Collision groups and categories

Physics

- Continuous physics with time of impact solver
- Persistent body-joint-contact graph
- Island solution and sleep management
- Contact, friction, and restitution

- Stable stacking with a linear-time solver
- Revolute, prismatic, distance, pulley, gear, mouse joint, and other joint types
- Joint limits, motors, and friction
- Momentum decoupled position correction
- Fairly accurate reaction forces/impulses

System

- Small block and stack allocators
- Centralized tuning parameters
- Highly portable C++ with no use of STL containers

3. GAME DEVELOPMENT ARCHITECTURE

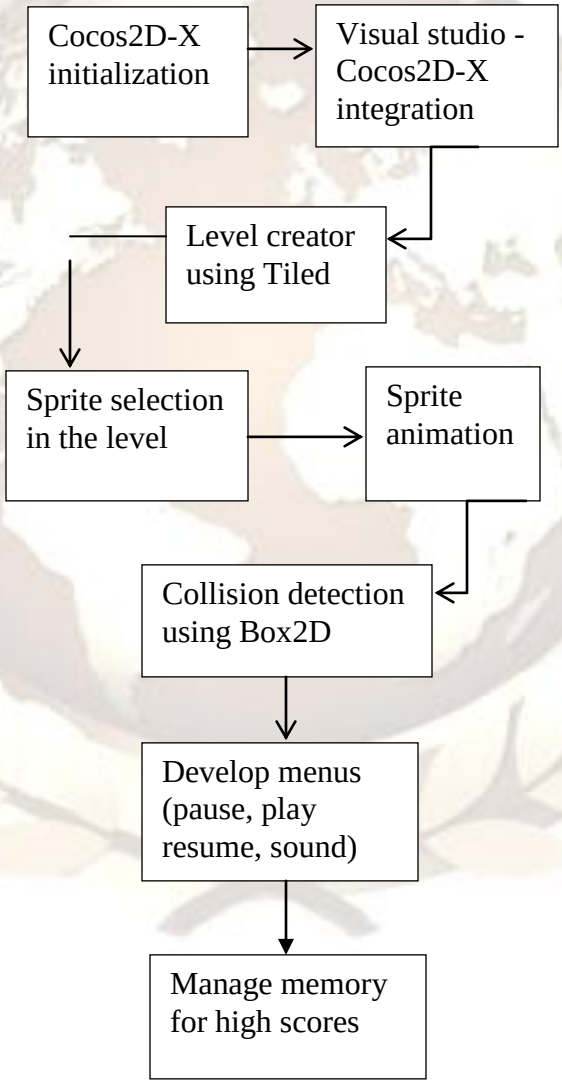

Fig 1. Overall System Architecture

Fig 1 describes the overall system architecture of the game development. It contains the various stages are as follows:

.

a) Level Generation

b) Character(Sprite) Animation

c) Collision Detection

d) Score & Memory Management

e) Menu Development

f) Porting to iPhone/android

a) Level Generation

This module deals with the level generation by creating the maps, selecting maps, arranging them in the order which we like either random or user defined. We use Tiled software for creating tmx files (map files). It contains many tile layers and object layers.

1. Tile layer- for the background tiles

2. Object layers- for the objects (polygon/ordinary objects)

Here we arrange the platforms where the player should jump, enemies which the player should not collide and collectables for collecting and providing extra scores.

b) Sprite Animation

This module deals with the sprite selection for the game. Sprites can be used as Hero for the game, enemies, collectables and background . The sprite in Cocos2d-x is selected using CCSprite, CCSpriteFrameCache and CCSpriteBatchNode. The frame cache is for plist files. Plist is nothing but the sprite sheet. Thus the sprite is selected and for animating it we use CCAnimation. The animation should be carried at regular intervals.

c) Collision Detection

This is the important part of the game development where sprites need to collide and their collision is to be detected or not. Here we are using the concept of box2d. We create polygons or other shapes around the bodies in the box2d world. Now we check if the polygons collide or not by checking the point of contact or contact listener. Contact listener contains

1.Begin Contact-what is to be done at the beginning of contact

2.End Contact-what is to be done at the end of contact

The collision is between player-enemy, player-collectables.

d) Score & Memory Management

This module deals with the score calculation and maintaining the high scores. The score is calculated based on the number of pixels the character has travelled and the ratio is considered. If he collects the collectables extra points will be added to the net score. For maintaining the high scores we need to create a memory allocation for the scores and display the top five or ten scores. The memory can be created by using the CCUserdefaults or new operator. The memory management technique should be used to optimize the performance of the game.

e) Menu Development

This module deals with the menu development such as the pause/play toggling, sound on/off toggling, close and to check the high scores options. The menu is created using CCMenu, CCLabel and CCLabelBMFont. The menu provides ease of gaming. There will be a main menu for starting the game and exiting the game the other menus will be inside the game screen.

f) **Porting**

Porting to different platforms is based on the release version that is developed using the windows. The exe file is used for running in the windows platform. The release version contains compatible game for other versions. The platform specific code is coded again and deployed. It is then release for iPhone and android platforms. A DLL is taken from the windows platform and it is used for other platforms.

4. RESULT ANALYSIS

Fig 2 describes the sample output for the game scenario in windows platform. This scenario includes the sprite Animation and based on the collision detected, the scores are displayed on the score board. Fig 3 shows debugging of the game. Fig 4 describes the sample output for the game scenario in Iphone.

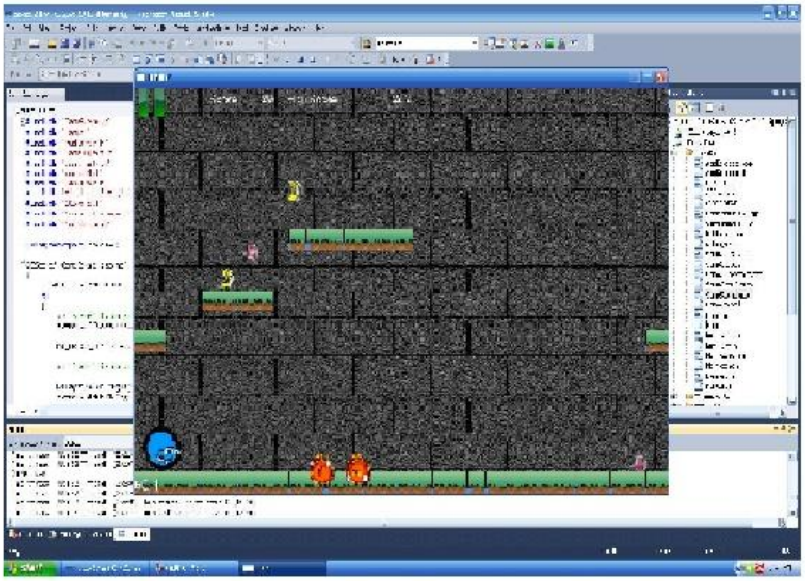

Fig 2.Sample Game scenario[Windows]

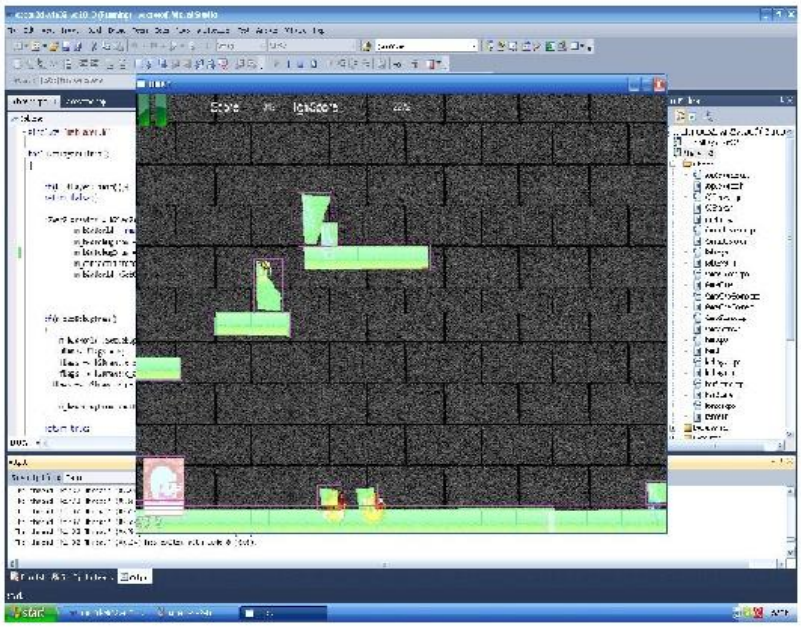

Fig 3.Game debugging

Fig 4.Sample Game scenario [IPhone]

CONCLUSION

This paper discusses the collision detection game development. The game includes several features incorporating game physics in it.It works for various platform like iphone and android.Most of the game development is not only for entertainment but they involve the problem solving method in it and this collision detection game development scenario paves way to learn the laws of physics.

REFERENCES

[1] Sawyer, B. (2004). Presentation in GDC 2004

[2] Wang, G. G. (2004). Semi-structured Questionnaire. Mr. Guo Guang Wang is the Java Engineer in Wireless Technology Department of NetEase, China. His responsibilities involve: mobile phone game based on J2ME development and Java programmes maintenance and debug.

[3] Chen Xin(2009),Cross-Platform Mobile Phone Game Development Environment International Conference on Industrial and Information Systems

[4] Zhang, S. J. (2004). Telephone Interview. Mr. Sheng Jing Zhang is product manager in Tianxingyuanjing Company (www.81088.com) in Beijing China. He is responsible for the development of a game simulator for Symbian 60 Series mobile phone.

[5] Introduction to Mobile Game Development. Nokia Corporation. Released in January 2003.

[6] Costikyan, G. (2002). J2ME & BREW Game Design: Designing games for J2ME and BREW technology. Game Developers Conference 2002.