УДК 004.421

РАЗРАБОТКА АЛГОРИТМА ПРИЛОЖЕНИЯ «АКВАРИУМ»

С. А. Хачатуров, А. О. Княгинина

Донской государственный технический университет, Ростов-на-Дону, Российская Федерация

hachaturovsergei@gmail.com
mambo.ann@yandex.ru

Рассмотрен алгоритм возникновения столкновений объектов на плоскости. Выявлены технически сложные и неправильно реализованные методы столкновения объектов и обоснована необходимость создания собственного нового алгоритма столкновений. На основе проведенного исследования авторы предлагают новый метод реализации алгоритма столкновений объектов.

Ключевые слова: java, приложение, алгоритм, разработка, столкновение объектов.

UDC 004.421

DEVELOPMENT OF "AQUARIUM" APPLICATION ALGORITHMS

S. A. Hachaturov, A. O. Knyaginina

Don State Technical University, Rostov-on-Don, Russian Federation

hachaturovsergei@gmail.com
mambo.ann@yandex.ru

This article considers the problem of objects collision. The paper identifies technically complex and incorrectly implemented methods of objects collision and the necessity for creation of a new collisions algorithm. Based on this study the authors propose their own method of implementation of the collision objects algorithm.

Keywords: java, application, algorithm, development, objects collision.

Введение. При разработке ряда компьютерных игр возникает необходимость обнаружения столкновений объектов. Данная тема на сегодняшний день хорошо изучена. Так в работе Собинова Д. И., Коробицына В. В. представлен обзор современных алгоритмов обнаружения столкновений объектов [1]. Рейган Бёрнс (Raigan Burns) и Мэйри Шепард (Mare Sheppard) — разработчики популярной flash-игры N — представили серии уроков, касающихся различных приёмов определения столкновений для маленьких и быстро движущихся объектов [2]. Томас Якобсен в своей работе «Моделирование движения персонажей компьютерных игр» описал различные методы, которые позволяют создают эффект правдоподобия при столкновении объектов в компьютерных играх [3]. Свои методы реализации столкновений объектов предлагают в различных обзорах и другие разработчики игр [4–6]. В результате изучения реализации в программном коде рассмотренных алгоритмов была выявлена проблема наличия множества технически сложных методов взаимодействия объектов. Неверные технологические решения приводят к значительному падению производительности таких приложений, а также осложняют их дальнейшую модификацию и развитие. В настоящей работе предложен метод расчета столкновения прямоугольных спрайтов, основанный на расчете положения объектов относительно друг друга. Реализация данного метода представлена на примере игры «Аквариум».

Основные элементы алгоритма и его реализация в игре «Аквариум». Приложение «Аквариум» — это мини-игра, главной целью которой является поедание главной управляемой

игроком рыбой (далее «пиранья») максимального количества других рыб аквариума (рис.1.). При этом рыбы, которых съедает «пиранья», должны быть меньше, чем она. Если рыба окажется больше, «пиранья» будет съедена и игра будет окончена. При поедании рыбок «пиранья» увеличивается в размере. Это происходит не сразу, но уже после первой съеденной рыбы вы заметите постепенное увеличение «пираньи».

Рис.1. Скриншот игры

В процессе работы над приложением появилась необходимость разработки и реализации алгоритма поиска столкновения 2D объектов. На основе анализа предложенных в литературе алгоритмов был разработан собственный алгоритм. Реализуемый в результате алгоритм является упрощенной версией метода проецирования [1, 3]. По сравнению с указанным методом, он проще и требует меньше аппаратных ресурсов.

Практически во всех играх приходится обрабатывать события, связанные со столкновением двух объектов или с препятствием. Как правило, при столкновении должны произойти какие-то действия, чаще всего связанные с изменением первоначального состояния объекта. Это может быть уничтожение объекта, его перемещение, уменьшение или увеличение. В основе изменения состояния объекта положен перебор имеющихся фреймов в анимационной последовательности или перерисовка изображения на новом месте [7, с. 218].

Столкновение прямоугольников. Чтобы определить прямоугольник, авторы используют координату его верхнего левого угла (рис.2). Для удобства сравнения положения всех прямоугольных спрайтов они помещены в двумерный массив. Первая строка массива хранит координату X, а вторая строка — координату Y. Ширина и высота хранятся в отдельных переменных [8, с. 417–418].

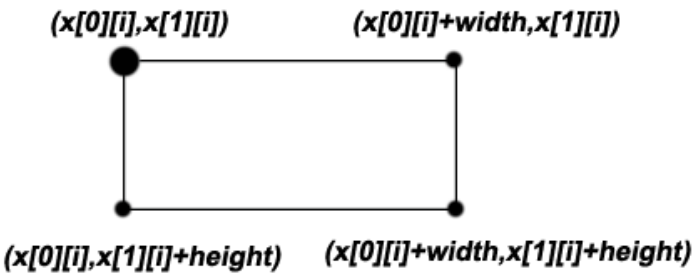

Рис.2. Размеры прямоугольного фрейма по координатам

Далее создается метод stolkn(), который вызывается каждые 20 мс, чтобы проверить столкнулись ли объекты ввиду движения спрайтов (рис.3.).

```
public void stolkn(){
    Random r = new Random();
for (int i=0; i<18; i++){
    for (int j=0;j<18;j++){
if ((((x[0][i]<x[0][j])&&(x[0][j]<x[0][i]+width))||
  ((x[0][i]<x[0][j]+width)&&(x[0][j]+width<x[0][i]+width)))&&
                (((x[1][i]<x[1][j])&&(x[1][j]<x[1][i]+height))||
                ((x[1][i]<x[1][j]+height)&&(x[1][j]+height<x[1][i]+height))))
```

Рис.3. Часть метода stolkn()

Метод заключается в следующем: двойной цикл проходит по всему массиву и проверяет условие соответствующее столкновению объектов для элементов массива. Первой точкой для проверки является верхний левый угол второго прямоугольного спрайта. Если она попадает в диапазон

(1)(x[0][i],x[0][i]+width) и (x[1][i],x[1]+height), значит можно сказать, что второй прямоугольник находится внутри первого (рис.4.). Отсюда следует, что объекты, соответствующие данным координатам, столкнулись. Столкновение объектов может происходить и в других случаях. Например, верхний правый угол находится в диапазоне (1), нижний левый или правый угол также удовлетворяют вышеуказанному условию. В сумме получается 4 ситуации столкновения объектов, которые возможно проверить предложенной функцией.

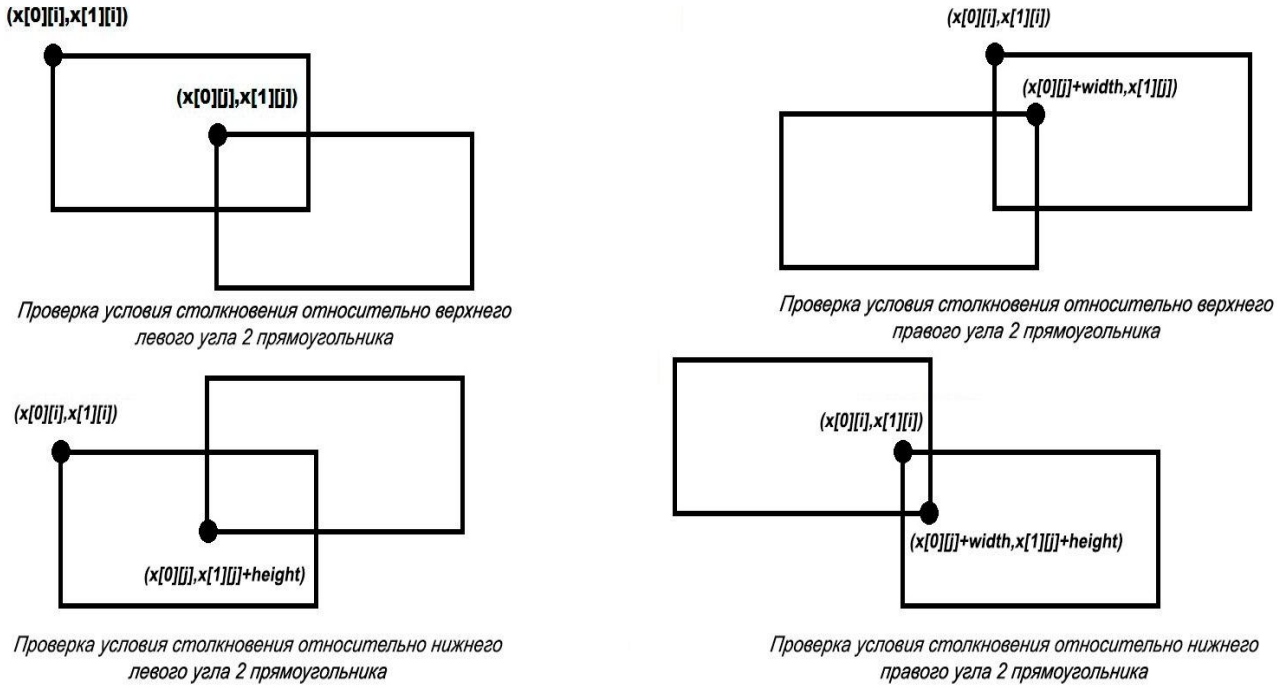

Рис.4. Положения прямоугольников

Заключение. Рассмотрен новый метод столкновения объектов, при построении которого были учтены проблемы уже имеющихся алгоритмов. Разработанный алгоритм был протестирован в реальном приложении и показал высокую скорость проверки данных. Таким образом, применение предложенного алгоритма позволяет не только снизить нагрузку аппаратного обеспечения, но и сократить время и трудоёмкость разработки.

Библиографический список

1. Собинов, Д. И. Алгоритмы обнаружения столкновений / Д. И. Собинов, В. В. Коробицын // Математические структуры и моделирование. — 2010. — вып. 21. — С.82–95.

2. N: уроки по программированию игр во флэше [Электронный ресурс]. / Собрание сочинений о флэше. — Режим доступа : http://noregret.org/tutor/n/ (дата обращения : 01.05.2016).

3. Thomas Jakobsen. Advanced Character Physics, Gamasutra, January 21, 2003. — Режим доступа : http://www.gamasutra.com/resource_guide/20030121/ /jacobson_01.shtml (дата обращения : 01.05.2016)

4. Базовая теория столкновения объектов, спрайтов на Javascript /. [Электронный ресурс] / Хабрахабр — Режим доступа : http://habrahabr.ru/post/128438/ (дата обращения : 01.05.2016).

5. Взаимодействие 2D объектов (столкновения) [Электронный ресурс] / Сайт сообщества русскоязычных XNA разработчиков. — Режим доступа : http://xnadev.ru/articles.php?article_id=72 (дата обращения : 01.05.2016).

6. Обработка столкновения объектов [Электронный ресурс]/ Тостер — вопросы и ответы для IT-специалистов. — Режим доступа : http://toster.ru/q/19683 (дата обращения : 01.05.2016).

7. Горнаков, С. Г. Программирование мобильных телефонов на Java 2 MicroEdition/ С. Г. Горнаков. — Москва : ДМК-Пресс, 2004. — 336 с.

8. Цехнер, М. Программирование игр под Android/ М. Цехнер. — Санкт-Петербург : Питер, 2013. — 688 с.