**Отчет**

**По лабораторной работе №1**

**Теория игр**

Шаграев А.

Шамаль А.

А-13-05

**2009**

1. **Описание игры.**

Игровая вселенная состоит из следующих объектов:

* Игровое поле (уровень), моделирующее природное окружение: игровое поле, на котором будут располагаться все действующие лица, препятствия (стены), окружение (небо).
* Собственно, действующие лица – танки модели Abrahms.

Один из танков находится под управлением игрока (человека). Остальные танки управляются компьютером. Цель игры – уничтожить все компьютерные танки.

Пользовательский танк считается более прочным, поэтому при столкновениях с компьютерными танками компьютерные танки уничтожаются. При столкновении любого танка со стеной он уничтожается. У всякого танка изначально есть 100 hp; при попадании снаряда, выпущенного другим танком, в данный танк, его количество жизней уменьшается на некоторое количество единиц. Именно: всякий компьютерный танк может выпускать снаряды разной мощности. Если снаряд мощности n попадает в другой танк, тот теряет n hp. При этом стрелявший получит повреждения в размере n/10. Пользовательский танк стреляет всегда снарядами мощности 10.



Система принятия решений используется для управления миром:

* отслеживания статуса игры (есть ли еще компьютерные танки? не был ли уничтожен пользовательский танк?);
* управления поведением компьютерных танков;
* отслеживания игровых событий (танк врезался в стену, столкнулись два танка, снаряд попал в танк и т.д.

Управление в игре: управление танком осуществляется клавишами a,w,s,d (влево, вперед, назад, вправо). Стрельба – на кнопке m. Для вызова справки нужно нажать кнопку h. Для выключения справки также нужно нажать кнопку h.

1. **Описание таблиц.**
   1. **Главная таблица (таблица 1)**

Отслеживает состояние игры: остались ли компьютерные танки, не взорвался ли пользовательский танк. Перед переходом к таблице 1 система всегда переходит к следующему объекту.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | E |
| Количество комп. Танков > 0? | T | F |  |
| hp пользовательского танка > 0 ? |  | T |  |
|  | S | S | табл2 |

* 1. **Таблица работы с объектом (таблица 2)**

В зависимости от текущего объекта, с которым происходит работа, выбирается следующая таблица.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Компьютерный танк? | T | F |
| Пользовательский танк? |  | T |
|  | Табл 3 | Табл 6 |

* 1. **Управление компьютерным танком (таблица 3)**

Здесь происходит обработка событий, которые могут произойти с компьютерным танком.

Если он врезается в другой танк или стену, он уничтожается. Если он видит пользовательский танк, он стреляет. Если в него попадает снаряд, то нужно уменьшить количество его жизней, после продолжить его движение. Если ни одно из этих событий не сработало, то он просто продолжает движение.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | E |
| Врезался в танк? | T | F | F | F | F |  |
| Врезался в стену? |  | T | F | F | F |  |
| Попал снаряд? |  |  | T | F | F |  |
| Вижу пользовательский танк? |  |  |  | T | F |  |
| Hp < 0? |  |  |  |  | T |  |
| Действие | Уничтожить | Уничтожить | Hp-- |  | Уничтожить |  |
| Таблица | 1 | 1 | 7 | 5 | 1 | 4 |

* 1. **Управление поведением компьютерного танка (таблица 7)**

Поведение танка происходит следующим образом: он проезжает вперед некоторое расстояние, там полностью осматривается (поворачивает башню на 360 градусов), затем поворачивается на 180 градусов, проезжает то же самое расстояние, снова осматривается, разворачивается на 90 градусов и начинает все с начала. Это поведение реализуется конечным автоматом: в состоянии 0 будем двигаться вперед. Как только пройдено необходимое расстояние, движение останавливается и автомат переходит в состояние 1. В этом состоянии танк поворачивает башню на 360 градусов. После этого он переходит в состояние 2, в котором разворачивается на 180 градусов. После этого он переходит в состояние 3. В состоянии 3 он двигается вперед. Как только движение завершилось, он переходит в состояние 4. В нем он вращает башню на 360 градусов. После этого он переходит в состояние 5. В нем он разворачивается на 90 градусов и переходит в состояние 6. В состоянии 6, как только он закончил разворот, он начинает двигаться вперед и переходит в состояние 0.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Состояние 0? | T | T | F | F | F | F | F | F | F | F | F | F |
| Состояние 1? |  |  | T | T | F | F | F | F | F | F | F | F |
| Состояние 2? |  |  |  |  | T | T | F | F | F | F | F | F |
| Состояние 3? |  |  |  |  |  |  | T | T | F | F | F | F |
| Состояние 4? |  |  |  |  |  |  |  |  | T | T | F | F |
| Состояние 5? |  |  |  |  |  |  |  |  |  |  | T | T |
| Вращается? |  |  |  |  | T | F |  |  |  |  | T | F |
| Поворачивает башню? |  |  | T | F |  |  |  |  | T | F |  |  |
| Двигается? | T | F |  |  |  |  | T | F |  |  |  |  |
| Действие |  | Rot  Tur  С. 1 |  | Rot  C.2 |  | Move  С.3 |  | Rot  Tur  С.4 |  | Rot  C.5 |  | Move  C.0 |
| Таблица | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

* 1. **Уклонение от снаряда (таблица 4)**

При попадании в танк снаряда, он будет совершать маневр уклонения, для чего создается еще один конечный автомат. В состоянии 0 будем двигаться вперед. Как только пройдено необходимое расстояние, движение останавливается и автомат переходит в состояние 1. В этом состоянии танк поворачивается на 180 градусов. После этого он переходит в состояние 3, в котором двигается вперед. После этого он переходит в состояние 4. В состоянии 4 он поворачивается на 90 градусов, после чего переходит в состояние 0, и снимается флаг «в танк попал снаряд».

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Состояние 0? | T | T | F | F | F | F | F | F | F | F |
| Состояние 1? |  |  | T | T | F | F | F | F | F | F |
| Состояние 2? |  |  |  |  | T | T | F | F | F | F |
| Состояние 3? |  |  |  |  |  |  | T | T | F | F |
| Состояние 4? |  |  |  |  |  |  |  |  | T | T |
| Вращается? |  |  | T | F |  |  | T | F |  |  |
| Двигается? | T | F |  |  | T | F |  |  | T | F |
| Действие |  | Поворот,  С. 1 |  | Движение,С.2 |  | Поворот, С.3 |  | Движение, С.4 |  | Поворот,C.0 |
| Таблица | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

* 1. **Управление стрельбой компьютерного танка (таблица 5)**

Т.к. танк теряет hp в процессе стрельбы, логично корректировать мощность снаряда в зависимости от количества hp, имеющихся у танка. Действие заключается в выстреливании снаряда определенной мощности, которая и указывается в строке «действие».

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  | E |
| Hp > 90? | T | F | F | F | F | F | F | F | F | F | F | F | F |  |
| Hp > 80? |  | T | F | F | F | F | F | F | F | F | F | F | F |  |
| Hp > 70? |  |  | T | F | F | F | F | F | F | F | F | F | F |  |
| Hp > 60? |  |  |  | T | F | F | F | F | F | F | F | F | F |  |
| Hp > 50? |  |  |  |  | T | F | F | F | F | F | F | F | F |  |
| Hp > 45? |  |  |  |  |  | T | F | F | F | F | F | F | F |  |
| Hp > 40? |  |  |  |  |  |  | T | F | F | F | F | F | F |  |
| Hp > 35? |  |  |  |  |  |  |  | T | F | F | F | F | F |  |
| Hp > 30? |  |  |  |  |  |  |  |  | T | F | F | F | F |  |
| Hp > 25? |  |  |  |  |  |  |  |  |  | T | F | F | F |  |
| Hp > 20? |  |  |  |  |  |  |  |  |  |  | T | F | F |  |
| Hp > 15? |  |  |  |  |  |  |  |  |  |  |  | T | F |  |
| Hp > 10? |  |  |  |  |  |  |  |  |  |  |  |  | T |  |
| Действие | 100 | 90 | 80 | 70 | 60 | 50 | 45 | 40 | 35 | 30 | 20 | 15 | 10 | 9 |
| Таблица | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

* 1. **Управление событиями пользовательского танка (таблица 6).**

Поведением танка управляет пользователь, но события отслеживает система.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  | E |
| Врезался в стену? | T | F | F |  |
| Попал снаряд? |  |  | T |  |
| Hp < 0? |  | T | F |  |
| Действие | Уничтожить | Уничтожить | Hp-- |  |
| Таблица | 1 | 1 | 1 | 1 |

Общее количество условий: 42.

Общее количество действий: 29.

Общее количество правил: 48.

1. **Результаты трансляции СИМПР**

Оптимизационная трансляция 1 таблицы: C1 P1 C2 P2 E

Оптимизационная трансляция 2 таблицы: C1 P1 C2 P2

Оптимизационная трансляция 3 таблицы: C1 P1 C2 P2 C3 P3 C4 P4 C5 P5 E

Оптимизационная трансляция 4 таблицы: C1 C7 P1 P2 C2 C6 P3 P4 C3 P5 P6 C4 P7 P8 C5 P9 P10

Оптимизационная трансляция 5 таблицы: C1 P1 C2 P2 C3 P3 C4 P4 C5 P5 C6 P6 C7 P7 C8 P8 C9 P9 C10 P10 C11 P11 C12 P12 C13 P13 E

Оптимизационная трансляция 6 таблицы: C1 P1 С3 З2 С2 P3

1. **Программная реализация.**

Программа реализована в среде Microsoft Visual Studio 2005, использовался язык C++. Прорисовка окружения осуществляется средствами MS DirectX 9.0.

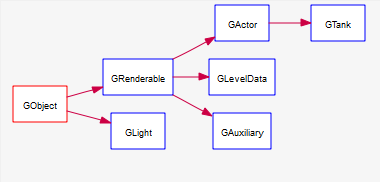
*Минимальные системные требования:*

* Процессор Intel или AMD с тактовой частотой не менее 1 Ггц
* 512 Мб ОЗУ
* Видеокарта, совместимая с DirectX 9.0 (nVidia GeForce FX и выше, AMD Radeon 9500 и выше)

*Используемые 3D-эффекты:*

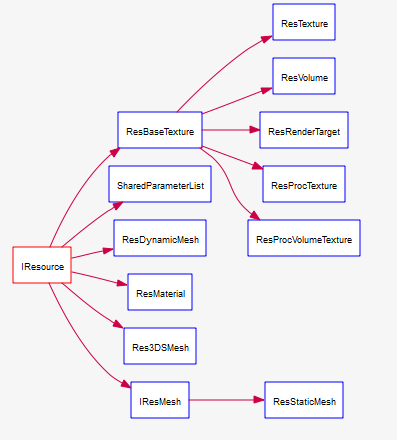
* Динамическое попиксельное освещение.
* Рельефное текстурирование (Bump mapping).
* Полупрозрачные объекты.
* Система частиц (дым, взрывы, след от снаряда).
* Мягкие частицы.
* Модели в формате (.3DS).
* Трехмерные текстуры.
* Кэширование эффектов.

*Иерархия основных классов.*

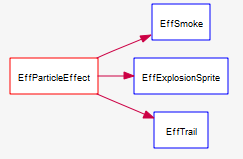
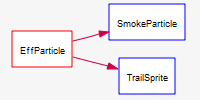
GObject – базовый класс иерархии игровых объектов.

Инстанциировать можно только терминальные классы.

*IResource – иерархия классов, инкапсулирующих аппаратные ресурсы видеокарты.*

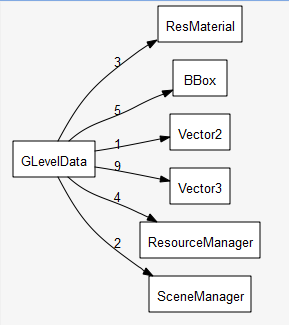
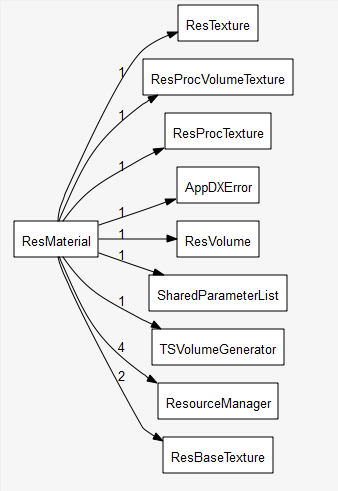


*Иерерхия классов, отвечающих за спецэффекты (дым, взрывы и т.д.)*





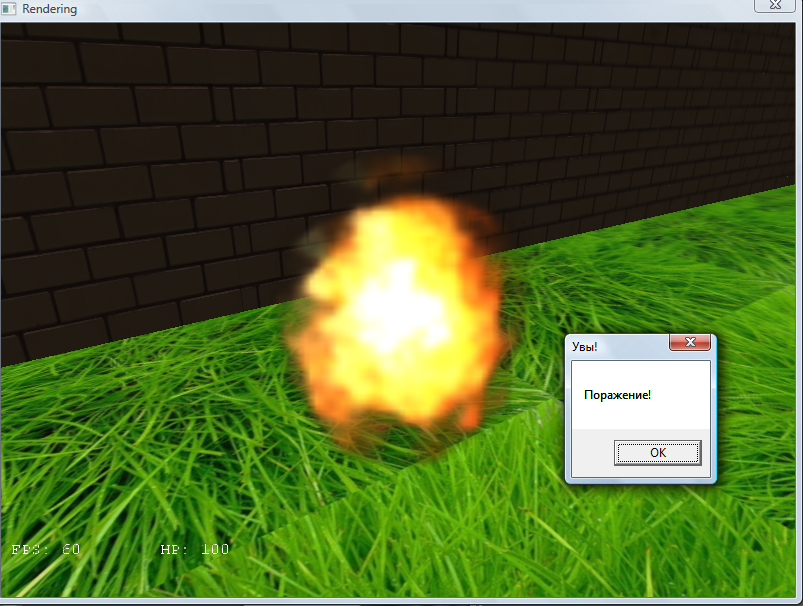
*Взаимодействие основных подсистем (примеры).*



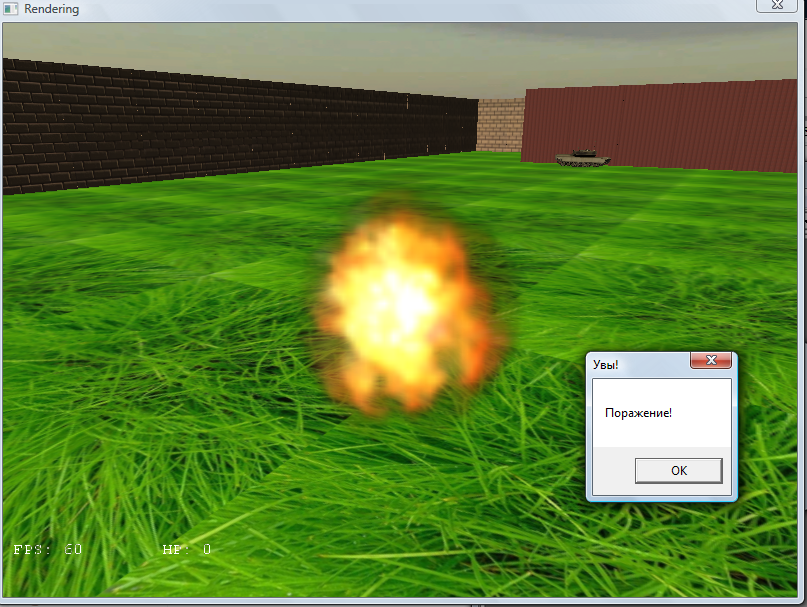
*Метрические характеристики проекта (только 3D-подсистема)*

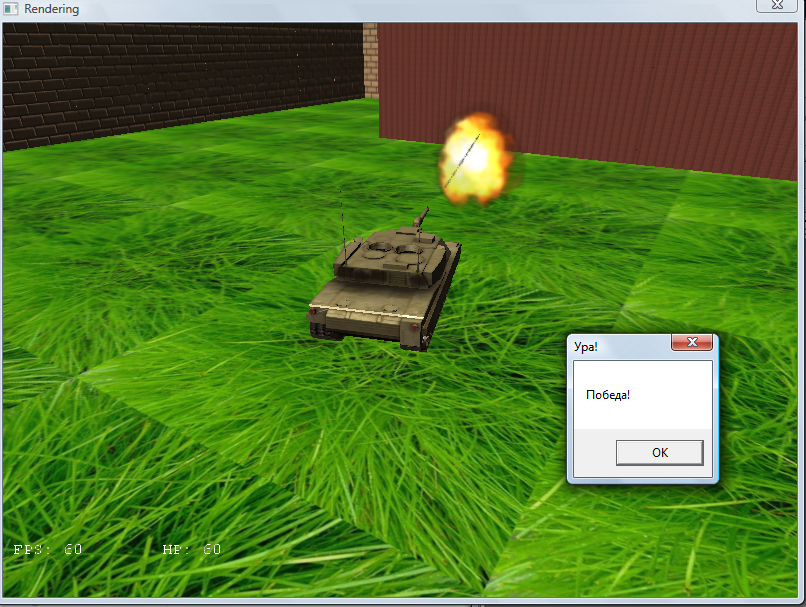
|  |  |
| --- | --- |
| Название характеристики | Значение |
| Файлов (ни считая файлов эффектов) | 28 |
| Строк кода | 6899 |
| Из них условных переходов | 15.3% |
| Комментариев | 6.1% |
| Описаний классов | 64 |
| Среднее кол-во методов на класс | 5,42 |
| Среднее кол-во стейтментов на метод | 6,1 |
| Максимальный показатель сложности | 32 |
| Максимальная глубина | 9 |
| Средний показатель сложности | 2,33 |
| Число глобальных функций | 39 |
| Стейтментов | 3687 |
| Файлов эффектов | 12 |

1. **Примеры работы программы**



1. **Приложение.**





1. **Реализация интерфейса с СИМПР.**

if (Msg == 49821)

{

WORD WParamHi = HIWORD(wParam);

WORD WParamLo = LOWORD(wParam);

if (WParamHi == 0)

return Universe->Conditions(WParamLo, lParam, AbsTime);

return Universe->Actions(WParamLo, lParam);

}

int Conditions(int TableNumber, int CondNumber, float cur\_time)

{

bool res;

switch (TableNumber)

{

case 1:

{

switch (CondNumber)

{

case 1:

{

GetNextObject();

int comp\_number = 0;

for (int i = 0; i < CompTanks.size(); ++i)

if (ActiveCompTanks[i])

++comp\_number;

res = comp\_number == 0;

}

break;

case 2:

{

int comp\_number = 0;

for (int i = 0; i < CompTanks.size(); ++i)

if (ActiveCompTanks[i])

++comp\_number;

res = (!(comp\_number == 0) && (!IsActiveUserTank));

}

break;

}

}

break;

case 2:

{

switch (CondNumber)

{

case 1: res = ObjNumber > 0; break;

case 2: res = ObjNumber == 0; break;

}

}

break;

case 3:

{

switch (CondNumber)

{

case 1: res = CompTanks[ObjNumber - 1]->TankCollide; break;

case 2: res = CompTanks[ObjNumber - 1]->WallCollide; break;

case 3: res = CompTanks[ObjNumber - 1]->MissileCollide; break;

case 4: res = CompTanks[ObjNumber - 1]->SeeUser; break;

case 5: res = CompTanks[ObjNumber - 1]->GameValue <= 0; break;

}

}

break;

case 4:

{

switch (CondNumber)

{

case 1: res = CompTanks[ObjNumber - 1]->Status == 0; break;

case 2: res = CompTanks[ObjNumber - 1]->Status == 1; break;

case 3: res = CompTanks[ObjNumber - 1]->Status == 2; break;

case 4: res = CompTanks[ObjNumber - 1]->Status == 3; break;

case 5: res = CompTanks[ObjNumber - 1]->Status == 4; break;

case 6: res = CompTanks[ObjNumber - 1]->IsRotating(); break;

case 7: res = CompTanks[ObjNumber - 1]->IsMoving(); break;

}

}

break;

case 5:

{

switch (CondNumber)

{

case 1: res = CompTanks[ObjNumber - 1]->GameValue > 90; break;

case 2: res = CompTanks[ObjNumber - 1]->GameValue > 80; break;

case 3: res = CompTanks[ObjNumber - 1]->GameValue > 70; break;

case 4: res = CompTanks[ObjNumber - 1]->GameValue > 60; break;

case 5: res = CompTanks[ObjNumber - 1]->GameValue > 50; break;

case 6: res = CompTanks[ObjNumber - 1]->GameValue > 45; break;

case 7: res = CompTanks[ObjNumber - 1]->GameValue > 40; break;

case 8: res = CompTanks[ObjNumber - 1]->GameValue > 35; break;

case 9: res = CompTanks[ObjNumber - 1]->GameValue > 30; break;

case 10: res = CompTanks[ObjNumber - 1]->GameValue > 25; break;

case 11: res = CompTanks[ObjNumber - 1]->GameValue > 20; break;

case 12: res = CompTanks[ObjNumber - 1]->GameValue > 15; break;

case 13: res = CompTanks[ObjNumber - 1]->GameValue > 10; break;

}

}

break;

case 6:

{

switch (CondNumber)

{

case 1: res = UserTank->WallCollide; break;

case 2: res = UserTank->MissileCollide; break;

case 3: res = UserTank->GameValue <= 0; break;

}

}

break;

}

if (res)

return 1;

else

return 0;

}

int Actions(int TableNumber, int CondNumber)

{

switch (TableNumber)

{

case 3:

{

switch (CondNumber)

{

case 1: DestroyCompTank(ObjNumber - 1); break;

case 2:

CompTanks[ObjNumber - 1]->GameValue -= 10;

CompTanks[ObjNumber - 1]->MissileCollide = false;

break;

case 3: DestroyCompTank(ObjNumber - 1); break;

}

}

case 4: CompTanks[ObjNumber - 1]->Core(); break;

case 5:

{

switch (CondNumber)

{

case 1: CompTankFire(ObjNumber - 1, 100); break;

case 2: CompTankFire(ObjNumber - 1, 90); break;

case 3: CompTankFire(ObjNumber - 1, 80); break;

case 4: CompTankFire(ObjNumber - 1, 70); break;

case 5: CompTankFire(ObjNumber - 1, 60); break;

case 6: CompTankFire(ObjNumber - 1, 50); break;

case 7: CompTankFire(ObjNumber - 1, 45); break;

case 8: CompTankFire(ObjNumber - 1, 40); break;

case 9: CompTankFire(ObjNumber - 1, 35); break;

case 10: CompTankFire(ObjNumber - 1, 30); break;

case 11: CompTankFire(ObjNumber - 1, 20); break;

case 12: CompTankFire(ObjNumber - 1, 15); break;

case 13: CompTankFire(ObjNumber - 1, 10); break;

case 14: CompTankFire(ObjNumber - 1, 9); break;

}

}

break;

case 6:

{

switch (CondNumber)

{

case 1: DestroyUserTank(); break;

case 2: UserTank->GameValue -= 20; UserTank->MissileCollide = false; break;

}

}

break;

}

return 1;

}