УДК 004.946

Н. А. БОЧКАРЕВ, Р. С. МОЛОТОВ

ПОДХОДЫ К ТРАНСФОРМАЦИИ ОБЪЕКТОВ ВИРТУАЛЬНЫХ ПРОСТРАНСТВ В СРЕДЕ UNITY

Рассматриваются особенности применения API-функций и воспроизведения анимаций для трансформации объектов в виртуальной среде симуляторов. Определяется ареал применения каждого из подходов, их преимущества и недостатки.

Ключевые слова: unity, моделирование, симуляторы, анимация.

Введение

В настоящее время в образовательном процессе активно внедряется и применяется программное обеспечение для обучения и контроля знаний, а также всё чаще для обучения используются симуляторы и различные тренажёры.

Практически в каждом тренажёре или симуляторе есть объекты, с помощью которых необходимо реализовать (визуализировать) взаимодействие между пользователем и виртуальным пространством. Это достигается изменением положения, цвета, размера или вращением этих объектов.

Для разработки симулятора – реализации имитационных моделей транспортных средств (ТС) – был выбран движок Unity.

При взаимодействии пользователя с виртуальным пространством симулятора объекты, наполняющие это пространство, могут менять свои свойства, взаимодействовать друг с другом, перемещаться, поворачиваться и т. д. Эти изменения могут быть определены заранее заданным сценарием, а могут и являться реакцией на действия (ввод) пользователя.

Наиболее частым вариантом изменений будет трансформация отдельного объекта. В среде Unity отдельный объект (GameObject) имеет неотключаемый компонент Transform [3], который отвечает за положение, поворот и размеры объекта; содержит инструменты и соответствующие им API-функцииTranslate, Rotate, Scale [4].

С помощью трансформаций объектов можно легко визуализировать динамические имитационные модели: подвижные механизмы, транспортные средства, персонажи и т. д.

Рассмотрим трансформации объектов виртуального пространства в среде Unity на примере построения динамической имитационной модели транспортного средства (ЗСУ 2С6 ЗПРК Тунгуска-М). Для этого разобьём все моделируемые трансформации на два класса:

1) производимые с параметрами, зависящими от действий пользователя (поворот башни, наведение орудий);

2) воспроизводимые с постоянными, заранее заданными параметрами (открытие и закрытие люков, вращение антенны локатора).

Для подобного моделирования будут использованы различные подходы: вызов API-функций и воспроизведение записанных анимаций соответственно. Рассмотрение реализации трансформаций разными способами позволит сравнить их и определить ареал (случаи) их применения.

Использование API-функций

Для осуществления трансформаций объектов, в частности вращения, можно использовать стандартный инструмент Rotate, который будет использоваться в скриптах (в Unity используется язык C#), с помощью вызова соответствующего API (листинг 1).Метод Rotate используется для вращения объекта вокруг осей, определяемых в первом передаваемом параметре. Рассмотрим пример использования Rotate для поворота башни ЗСУ.

Перед рассмотрением программного кода важно отметить, что, в отличие от многих пакетов 3D-моделирования, Unity использует другую систему координатных осей: вертикальная ось здесь Y, а ось Zдолжна соответствовать направлению движения (фронтальной проекции) ТС.

Рис. 1. Визуализация поворота башни ЗСУ комплекса Тунгуска-М

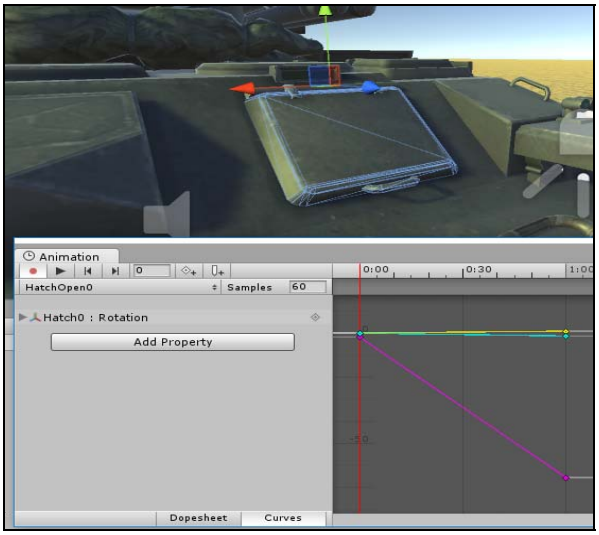

Рис. 2. Пример создания анимации: люк в положении «закрыто». На нижней части изображения находится панель анимаций

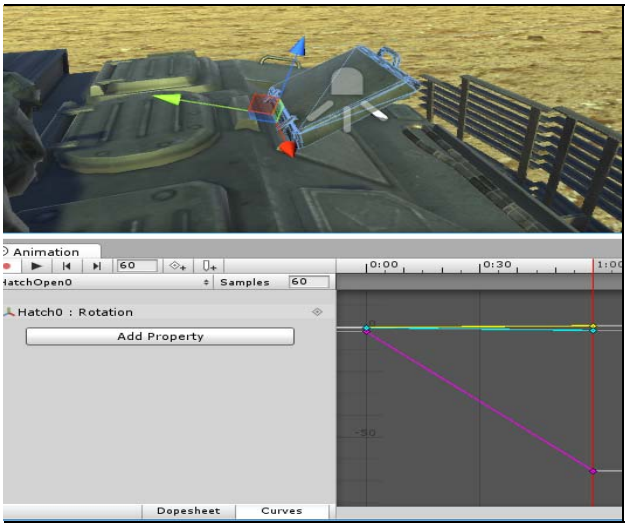

Рис. 3. Люк в положении «открыто»

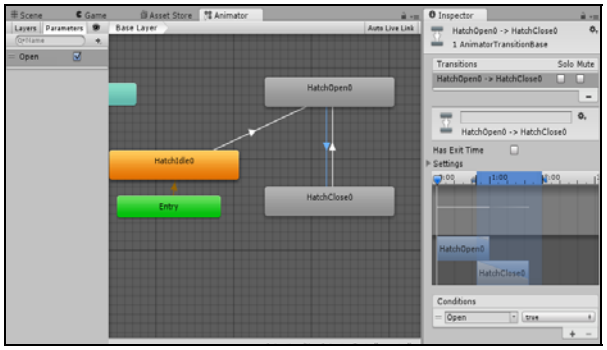

Рис. 4. Панель Animator

Листинг 1. Функция поворота башни с помощью Rotate

```
    public Transform turret;
    public int speed;
    void Update() {
        if(Input.GetKey(KeyCode.LeftArrow)) {
                turret.Rotate(new Vector3(0, -5,
0),speed * Time.deltaTime, Space.Self);
        } else if
     (Input.GetKey(KeyCode.RightArrow)) {
                turret.Rotate(new Vector3(0, 5,
0),speed * Time.deltaTime, Space.Self);
        }
    }
```

Как видно из листинга 1, при нажатии клавиши «Стрелка влево» башня вращается по осям, соответственно вектору (0, −5, 0) со скоростью speed градусов в секунду. Аналогичная трансформация происходит и при нажатии на клавишу «Стрелка вправо», только в этом случае башня вращается в обратную сторону – вектор (0, 5, 0). Вращение происходит и при удержании клавиши (GetKey вместо GetKeyDown).

Так как переменная скорости вращения speed имеет модификатор доступа public, то её можно изменять в инспекторе Unity, не редактируя код скрипта. Использование метода Rotate в данном случае удобно тем, что угол поворота можно менять динамически с нужной скоростью,

максимально точно реагируя на ввод пользователя (время, которое зажата клавиша), и данный метод не имеет ограничений по углам поворота башни: она может вращаться на 360 градусов.

Недостатком метода Rotate является то, что при вращении объекта на определённый угол необходимо знать текущий угол поворота этого объекта и углы, соответствующие двум положениям объекта: начальному и конечному. В некоторых случаях вычисление вектора поворота может быть неудобным и громоздким с точки зрения логики скрипта.

Использование анимаций в Unity

Задачу создания динамических имитационных моделей можно выполнить и с использованием анимаций. С помощью анимаций можно моделировать и визуализировать большое количество изменений объекта: вращение, перемещение, изменение размеров и многое другое. Анимации удобны в тех случаях, когда необходимо переключать параметр между несколькими, заранее известными значениями, которые связаны с ключевыми точками временной линии анимации и не могут быть изменены: модель двери, пребывающая в одном из двух состояний (открыто/закрыто), модель люка, заслонок пусковых установок, локатора обнаружения целей и т. д.

Рассмотрим пример использования анимации на примере открытия и закрытия люка механика-водителя.

Для создания анимации достаточно задать два положения, одно будет находиться в начале анимации, другое – в конце. На панели анимаций видна прямая, показывающая изменение угла люка с течением времени. Вертикальная прямая показывает текущее время относительно начала анимации. С перемещением этой прямой будет производится изменение угла люка, соответствующее прямой.

Анимация закрытия реализуется аналогичным образом, только начальное и конечное состояние поменяются местами.

Для реализации трансформаций таким способом недостаточно одних заранее записанных анимаций, необходим специальный контроллер, компонентAnimator, который будет воспроизводить эти анимации.

На панели Animator создаются состояния, между состояниями создаются связи, при прохождении которых будет запускаться анимация конечного состояния. К примеру, при переходе из состояния HatchOpen0 в HatchClose0 будет запускаться анимация закрытия люка. Также можно создавать различные параметры, которые будут контролировать переходы из одного состояния в другое. В данном случае создан параметр Open типа bool, который показывает, открыт люк или нет. На каждую связь можно добавить условие. Если условие истинно, то совершается переход. Для запуска анимации достаточно изменить этот параметр (листинг 2).

Листинг 2. Открытие двери с помощью анимации

```
private Animator anim;
void Start() {
      anim = GetComponent<Animator>();
}
void Update() {
      if (Input.GetKeyUp(open)) {
anim.SetBool("Open", !anim.GetBool("Open"));
      }
}
```

Логика работы скрипта следующая:

1. Объявляется объект Animator.
2. При выполнении функции Start (перед рендерингом кадров и началом выполнений функций Update, если скрипт включён) происходит инициализация аниматора.
3. При отпускании (GetKeyUp) пользователем кнопки открытия/закрытия люка происходит переключение флага, отвечающего за переход между состояниями анимации.

Заключение

Итак, рассмотрены два различных способа создания динамических имитационных моделей: посредством вызова API-функций (на примере Rotate) и с помощью воспроизведения анимаций. Как было продемонстрировано выше, ареал применения данных методов по большей части зависит от типа воспроизводимой динамической модели. Так, вызов API-функций в скриптах удобно использовать, когда параметры трансформации объекта зависят от ввода пользователя, т. е. являются динамическими. Анимацию же удобно применять как воспроизведение трансформаций [объектов виртуальной среды симулятора] с заранее определёнными параметрами. Стоит также отметить, что анимацию можно импортировать напрямую из пакетов 3D-моделирования (3dsMAX, Blender), что делает возможным тесное сотрудничество и глубокое взаимопонимание 3d-дизайнера и разработчика скриптов. Таким образом, значительно упрощается командное взаимодействие группы разработчиков при реализации сложных имитационных моделей и симуляторов.

СПИСОК ИСТОЧНИКОВ

1. Русскоязычное сообщество разработчиков на Unity. URL: http://www.unity3d.ru/ (дата обращения: 01.06.2016 г.).

2. Официальный сайт Unity3D. URL: http://unity3d.com/ru/ (дата обращения: 01.06.2016 г.).

3. Трансформации объектов в Unity. URL: http://docs.unity3d.com/ru/current/Manual/Transforms.html (дата обращения: 01.06.2016 г.).

4. Справочник API Unity. URL: http://docs.unity3d.com/ru/current/ScriptReference/index.html (дата обращения: 01.06.2016 г.).

5. Джамбруно, М. Трёхмерная графика и анимация. – М. : Вильямс, 2002. – 640 с.

6. Торн А. Основы анимации в Unity3D. – М. : ДМК-Пресс, 2015. – 250 с.

7. Goldstone W. Unity Game Development Essentials – Packt Publishing Ltd., 2009. – 315 с.

• • • • • • • • • • • • • • • • • • •

***Бочкарев Николай Алексеевич**, студент УлГТУ, кафедра «Вычислительная техника.*
***Молотов Роман Сергеевич,** младший научный сотрудник НИО УНИ УлГТУ.*

Поступила 16.08.2016 г.

УДК 004.946

С. Д. ЛИЗЯЕВ, Р. С. МОЛОТОВ

ОСОБЕННОСТИ СОЗДАНИЯ АНИМАЦИИ ПРИ РАЗРАБОТКЕ ОБУЧАЮЩИХ СИМУЛЯТОРОВ В СРЕДЕ UNITY

Рассматриваются особенности создания и редактирования анимации при разработке имитационных моделей транспортных средств в среде Unity.

Ключевые слова: unity, моделирование транспортных средств, симуляторы, создание анимации.

Введение

Современный образовательный процесс активно применяет программное обеспечение для обучения и контроля знаний. Помимо прочих информационных технологий, для обеспечения процесса обучения всё более активно используются обучающее симуляторы и тренажёрные системы. Их применение возможно на различных уровнях образования: как в учреждениях дошкольного и общего, так и среднего профессионального, высшего и дополнительного образования.

Симуляторы, использующиеся для подготовки специалистов в технической сфере, в большинстве своём содержат имитационные модели сложных машин, механизмов и их систем: транспортных средств (ТС), станков, инструментов, приборов и т. д. Все эти модели должны быть динамическими: обеспечивать визуализацию процесса работы пользователя и взаимодействия друг с другом в составе виртуальной среды симулятора. Таким образом, перед разработчиком симулятора стоит сложная задача по реализации логики воспроизведения моделями (объектами среды симулятора) динамических свойств реальных объектов предметной области.

Среда разработки Unity, выбранная для иллюстрации решения подобных задач, содержит мощный инструмент для создания, редактирования и воспроизведения динамических моделей – анимации. Использование анимаций позволяет с помощью визуального редактора построить автомат динамической модели объекта: определить основные статические состояния, переходы и условия переходов между ними.

Далее предлагается рассмотреть реализацию анимации модели транспортного средства (на примере БМ 9К33М3 комплекса «Оса-АКМ»), которая будет интегрирована в обучающий симулятор военного полигона.

Создание анимации ТС

Для осуществления поставленной задачи предлагается использовать стандартные инструменты для создания анимации, такие как Animation,