



# Оглавление

Введение	
Кому будет полезна эта книга?	10
Как организована книга	10
Условные обозначения, используемые в этой книге	11
Почему <i>Blueprints</i> ?	11
Дополнительные материалы	11
Об авторе	11
<b>Часть I. Начало работы</b>	<b>13</b>
Глава 1. Терминология и лучшие практики	15
1.1. Терминология	16
1.1.1. Устройства	16
1.2. Лучшие практики	21
1.3. Заключение	22
Глава 2. Настройка шлема виртуальной реальности	23
2.1. <i>Gear VR</i>	24
2.1.1. Настройка проекта <i>Gear VR</i>	24
2.1.2. Настройка глобального меню <i>Gear VR</i>	30
2.1.3. Глобальное меню изменения материалов <i>Gear VR</i>	36
2.2. <i>Rift</i> и <i>Vive</i>	41
2.2.1. Настройка проекта <i>Rift</i> и <i>Vive</i>	42
2.2.2. <i>Rift</i> и <i>Vive</i> — режимы отслеживания	47
2.3. Заключение	50
Глава 3. Инструменты	51
3.1. Библиотека обобщенных функций	52
3.2. Библиотека функций <i>Oculus</i>	53
3.3. Библиотека функций <i>Steam VR</i>	54
3.4. Заключение	55
<b>Часть II. Рецепты</b>	<b>57</b>
Глава 4. Взаимодействие на основе трассировки	59
4.1. Понимание взаимодействия трассировки	60
4.1.1. Принципы взаимодействия на основе трассировки	60
4.1.2. Принципы взаимодействия с пользователем	64
4.1.3. Настройка взаимодействия трассировки	65

---

4.1.4. Начальная настройка проекта	66
4.1.5. Настройка интерфейса взаимодействия	67
4.1.6. Компоненты взаимодействия	69
4.1.7. Установка взаимодействия <i>Pawn</i>	80
4.2. Создание простого объекта взаимодействия	82
4.3. Заключение	86
4.4. Упражнения	86
<b>Глава 5. Телепортация</b>	<b>91</b>
5.1. Настройка телепортации	92
5.1.1. Параболическая трассировка	92
5.2. Визуализация телепорта	97
5.2.1. Визуализация материала	97
5.2.2. Визуализация актора	99
5.3. Простой регулятор телепортации	100
5.4. Заключение	104
5.5. Упражнения	105
<b>Глава 6. Графика движения и 2D-интерфейсы с пользователем в <i>Unreal</i></b>	<b>107</b>
6.1. Проблемы 2D-интерфейса с пользователем в <i>VR</i>	108
6.2. История и совместимость <i>UMG</i>	109
6.3. Простое <i>VR</i> -меню	109
6.3.1. Меню <i>Actor</i>	111
6.3.2. Меню <i>Pawn</i>	113
6.4. Взаимодействие с пользовательским меню	114
6.4.1. Реализация взаимодействия с меню. Первый способ	115
6.4.2. Реализация взаимодействия с меню. Второй способ	119
6.5. Заключение	123
6.6. Упражнения	124
<b>Глава 7. Инверсная кинематика персонажа</b>	<b>125</b>
7.1. Введение в систему инверсной кинематики	126
7.2. Настройка инверсной кинематики головы	128
7.2.1. Создание зеркала	128
7.2.2. <i>Pawn</i> для инверсной кинематики	131
7.2.3. <i>Blueprint</i> анимации инверсной кинематики головы	133
7.3. Настройка инверсной кинематики руки	138
7.3.1. Добавление контроллеров к <i>Pawn</i>	138
7.3.2. <i>Blueprint</i> анимации инверсной кинематики руки	139
7.4. Заключение	145
7.5. Упражнения	145

---

Глава 8. Взаимодействие с контроллерами движения	147
8.1. Движения	148
8.1.1. Почему работает взаимодействие с контроллерами движения?	148
8.1.2. На что обратить внимание: важность доступности	148
8.1.3. Общие выходные данные текущего поколения контроллеров движения	149
8.1.4. Настройка проекта взаимодействия с миром	150
8.1.5. Взаимодействие с объектами	151
8.2. Создание интерактивных объектов	161
8.2.1. Создание интерактивного <i>Static Mesh Actor</i>	161
8.2.2. Создание интерактивной кнопки	163
8.2.3. Создание интерактивного рычага	171
8.3. Заключение	183
8.4. Упражнения	183
Глава 9. Перемещение в <i>VR</i>	185
9.1. Тренажерная болезнь	186
9.2. Типы перемещений	186
9.2.1. Естественное перемещение	187
9.2.2. Телепортация	187
9.2.3. Транспортные средства	188
9.2.4. Физическое перемещение	188
9.2.5. Искусственное перемещение	189
9.3. Реализация передвижения	190
9.3.1. Настройка <i>First Person</i> шаблона для отслеживания поворота	190
9.3.2. <i>First Person</i> шаблон для реализации бега на месте	196
9.4. Заключение	199
9.5. Упражнения	199
Глава 10. Оптимизация <i>VR</i>	201
10.1. Технические требования для рендеринга <i>VR</i>	202
10.2. Уменьшение задержки	203
10.3. Повышение производительности	210
10.3.1. Методы визуализации	211
10.3.2. <i>Instanced stereo</i>	216
10.3.3. Оптимизация сетки скрытой области	217
10.4. Настройка проекта <i>VR</i>	217
10.5. Заключение	225
10.6. Упражнения	225

<b>Часть III. Приложения</b>	<b>227</b>
<b>Приложение А. VR-редактор</b>	<b>229</b>
А.1. Включение VR-редактора	230
А.2. Управление VR-редактором	231
А.2.1. Навигация в виртуальном мире	232
А.2.2. Взаимодействие с объектами	233
А.2.3. Взаимодействие с меню	234
А.3. Заключение	237
<b>Приложение В. Ресурсы</b>	<b>239</b>
В.1. <i>Epic</i>	240
В.2. <i>Oculus</i>	240
В.3. <i>Valve</i>	240
В.4. <i>Google</i>	241
В.5. Сообщества	241
В.6. Офлайн-встречи	241
В.7. Конференции	241
В.8. Русскоязычные ресурсы и сообщества	242
<b>Приложение С. Глоссарий</b>	<b>243</b>
С.1. Основные понятия	244
С.2. Термины <i>Unreal Engine</i>	244
<b>Приложение D. Отладка приложений без шлема</b>	<b>245</b>
D.1. Отладка	246
D.2. Заключение	248
D.3. Задания	248
<b>Приложение Е. Список сокращений</b>	<b>249</b>

## ВВЕДЕНИЕ

Становление виртуальной реальности (VR) происходит на наших глазах, именно сейчас зарождается огромный спрос на новый и захватывающий опыт в VR. В качестве основы растущего многомиллиардного рынка VR ставит восхитительно сложные задачи перед разработчиками компьютерных игр, причем одновременно с погружением многих ранее не затронутых отраслей в мир компьютерной графики в реальном времени.

Создание аркадной игры в классическом стиле, визуализация изысканной виллы на травянистых холмах Тосканы — независимо от типа игры VR обеспечивает беспрецедентный уровень погружения в любую приглянувшуюся вам сферу. Однако это погружение вызывает множество проблем. При разработке под VR, хотите того или нет, вы пишете свои правила.

В этой книге рассмотрены общие парадигмы человеко-машинного взаимодействия, появившихся за последние несколько лет, и лучшие практики в этой области. Все сообщество VR — от крупнейших игроков до разработчиков-одиночек — с потрясающей скоростью вносит свой вклад в свод знаний VR. Эта книга не только продемонстрирует, как реализовать эти парадигмы средствами *Unreal Engine*, но и поможет выбрать решения, подходящие именно вашему проекту.

В основе книги рецептов лежит практический подход к изучению особенностей VR-разработки. Каждый «рецепт» представляет собой вариант сборки достаточно общей системы, подходящей для разнообразных VR-игр. Независимо от того, разрабатываете вы игру-стрелялку от первого лица или симулятор для отдыха, содержание каждого примера достаточно абстрактно для использования в игре любого жанра, но дополнительно предлагает конкретные решения, хорошо работающие для определенных типов игр.

## Кому будет полезна эта книга?

Эта книга предназначена для людей, которые уже знакомы с навигацией в *Unreal Engine 4 (UE4)* и *Blueprints*. Если у вас мало опыта, ознакомьтесь с документацией по *Unreal Engine* перед чтением этой книги в интернете по адресу: <http://docs.unrealengine.com>.

Тем не менее я объясняю большинство вещей, когда дело касается непосредственно кодирования, а большая часть математики раскрывается в приложениях и покрывается основным содержанием книги, поэтому серьезных навыков разработки программного кода от читателя не требуется.

## Как организована книга

Книга состоит из трех частей.

1. Часть 1 «Начало работы»: в главах 1–3 содержится введение в некоторые термины, используемые в этой книге и в отрасли VR. В этой части также содержатся инструкции по созданию простых проектов для различных VR-шлемов.
2. Часть 2 «Рецепты»: главы 4–10 содержат основные «рецепты» книги. Эта часть охватывает все: от взаимодействия с контроллером до схем движения в VR.
3. Часть 3 «Приложения»: это вспомогательная информация о редакторе VR и ресурсах, которые помогут вам в VR-разработке.

## Условные обозначения, используемые в этой книге

В этой книге используются следующие соглашения при оформлении текста:

- моноширинным шрифтом набраны блоки программного кода;
- курсивным начертанием выделены ключевые слова или фразы.

### Заметка

Означает подсказку, предложение или общее примечание.

### ДОПОЛНЕНИЕ

Содержит вспомогательную информацию к основному тексту, такую как объяснение используемых математических принципов или работ, связанных с основным содержанием.

### Предупреждение

Обозначает предупреждение или предостережение.

## Почему *Blueprints*?

Когда вы программируете в UE4, есть два способа реализации логики игры: язык визуального программирования *Blueprints* и ставший традиционным язык программирования C++.

По сравнению с *Blueprints*, C++ может быть сложнее, поскольку изучение синтаксиса может занять некоторое время; однако он предлагает различные варианты в использовании скрытых возможностей ядра. Слабое владение C++ не создаст проблем при работе с этой книгой, потому что большая часть материала представлена на уровне, чтобы возможности *Blueprints* позволяли его освоить.

Использование *Blueprints* также облегчает способы миграции вашей работы из одного проекта в другой, что позволит вам взять любую работу, проделанную по этой книге, и легко применить ее в ваших проектах.

## Дополнительные материалы

По адресу [https://eksmo.ru/files/ue4\\_vr\\_projects.zip](https://eksmo.ru/files/ue4_vr_projects.zip) можно скачать архив с файлами с исходным кодом для каждой части. Это позволит быстро проверить работоспособность рецептов, приведенных в книге.

## Об авторе

Митч Маккефри — независимый разработчик игр и создатель многих общественных ресурсов для разработчиков виртуальной реальности на *Unreal Engine*. Он обучает лучшим практикам разработки игр на своем YouTube-канале *Mitch's VR Lab*, где демонстрируются примеры, предложенные членами сообщества VR-разработчиков. Адрес его сайта <https://mitchellmccaffrey.com>.





ЧАСТЬ I

# НАЧАЛО РАБОТЫ



# ТЕРМИНОЛОГИЯ И ЛУЧШИЕ ПРАКТИКИ

Мир разработки виртуальной реальности (VR) может показаться сложным из-за разнообразия доступного на рынке оборудования и программного обеспечения. Кроме того, поскольку VR — новая среда, многие принципы, принятые разработчиками игр, неприменимы и могут не работать в мире VR-игр.

Если вы не уверены, что понимаете разницу между *Oculus VR*, *OSRV* и *OpenVR*, или просто ищете общие рекомендации о том, как начать работать с VR, эта глава для вас.

## 1.1. Терминология

Экосистема VR непрерывно усложняется и развивается благодаря обилию технологий, программного и аппаратного обеспечения. Чтобы убедиться, что мы верно понимаем друг друга, рассмотрим ключевые части этой экосистемы, которые должен знать каждый VR-разработчик, использующий *Unreal Engine 4 (UE4)*. Если вы знакомы с текущим состоянием отрасли виртуальной реальности и используемыми технологиями, можете пропустить эту главу.

### 1.1.1. Устройства

Для работы с виртуальной реальностью вам потребуются *шлем виртуальной реальности [Head Mounted Display — HMD]* и контроллер. *Unreal Engine* на уровне коробочного решения поддерживает большинство из представленных на рынке решений, что избавляет разработчиков от необходимости однозначного выбора конкретного VR-шлема на первых этапах проекта. Кроме того, в *UE4* для VR реализован высокий уровень абстракции, что позволяет легко справиться со сменой устройства, для которого вы разрабатываете проект (или если вы ориентируетесь сразу на несколько устройств).

Таблица 1.1 содержит список и описание VR-шлемов, штатно поддерживаемых *UE4*.

Таблица 1.1. Совместимые VR-шлемы (HMD)

VR-шлем	Описание
<i>Samsung Gear VR</i>	<i>Gear VR</i> — аппаратура, совместимая со смартфонами <i>Samsung</i> , разработанная в партнерстве <i>Oculus</i> и <i>Samsung</i> с целью обобщить возможности <i>Oculus</i> в области программного обеспечения для мобильной виртуальной реальности. При работе с <i>Gear VR</i> вы можете использовать все функции и преимущества, которые дает среда разработки <i>Oculus Mobile software development kit (Oculus Mobile SDK)</i> . <i>Gear VR</i> в настоящее время поддерживает только отслеживание поворотов головы виртуальной модели
<i>HTC Vive</i>	<i>HTC Vive</i> разработан в результате партнерства, сходного с <i>Gear VR</i> , но в котором <i>HTC</i> выступили поставщиком аппаратного обеспечения, а <i>Valve</i> — программного SDK в виде <i>SteamVR/OpenVR</i> . <i>Vive</i> обеспечивает отслеживание вращений, а также взаимно однозначное отслеживание положения шлема и включенных контроллеров движения. С прицелом на разработку под <i>Vive</i> , <i>Valve</i> предоставляет <i>OpenVR SDK</i> (для получения дополнительной информации о <i>OpenVR</i> см. таблицу 1.3), который предоставляет доступ к различным датчикам и данным настройки, необходимым для VR
<i>Oculus Rift</i>	Аппаратура <i>Oculus Rift</i> прошла через много итераций, представленных как выпуски для разработчиков, не предназначенные для широкой публики. В 2016 году <i>Oculus</i> выпустила первую массовую версию (CV1), предназначенную для продажи пользователям. CV1 — это полностью интегрированная система, состоящая из оборудования <i>Oculus</i> и программного обеспечения (см. <i>SDK Oculus</i> в таблице 1.3). <i>Rift</i> поддерживает отслеживание вращения, подобно <i>HTC Vive</i> . В комплект также входит контроллер <i>Xbox One</i> и пульт дистанционного управления <i>Oculus Remote</i> , являющийся устройством ввода по умолчанию; VR-контроллеры <i>Oculus Touch</i> (см. таблицу 1.2) продаются отдельно

VR-шлем	Описание
<i>Google Cardboard</i>	<i>Google Cardboard</i> — это небольшая и недорогая (350–1500 рублей) картонная коробочка, в которой есть одна кнопка и линзы. Коробка позволяет закрепить смартфон и просматривать стереометрический контент. «Картон» использует встроенные датчики смартфона для обнаружения вращения головы пользователя (разумеется, на неоткалиброванном телефоне измерения будут неточными). <i>Google Cardboard</i> использует собственные <i>SDK Android</i> для обеспечения совместимости с устройствами (что может приводить к существенным задержкам рендеринга по сравнению с другими решениями). Тем не менее «Картон» на сегодня является самой дешевой VR-гарнитурой
<i>Google VR/Daydream VR</i>	<i>Google VR</i> или <i>Daydream VR</i> — еще одна инициатива компании <i>Google</i> . По сравнению с <i>Google Cardboard</i> эта программно-аппаратная система контролируется жестче. Для работы с ней ваш смартфон и гарнитура должны быть классифицированы как <i>Daydream Ready</i> , что позволяет <i>Google</i> использовать более продвинутые функции программного обеспечения. <i>Daydream</i> также поддерживает контроллер <i>Daydream</i> (см. таблицу 1.2), предоставляющий сенсорную панель и обеспечивающий отслеживание вращения. Таргетинг <i>Daydream</i> реализуется через <i>Google VR SDK</i> (таблица 1.3)
<i>PlayStation VR</i>	<i>PlayStation VR</i> является периферийным устройством для VR-игр на игровых приставках (консолях) <i>Sony PlayStation 4</i> и <i>PlayStation 4 Pro</i> . Отслеживание положения осуществляется при помощи внешней камеры <i>PlayStation</i> , отслеживание вращения реализовано гиростабилизаторами ( <i>IMUs, inertial measurement units</i> ) примерно так же, как у других шлемов, представленных в этой таблице. Игроки могут использовать контроллеры <i>PlayStation Move</i> (таблица 1.2). Также можно использовать контроллер <i>DualShock 4</i> , в том числе для отслеживания положения
<i>OSRV</i>	Виртуальная реальность с открытым исходным кодом ( <i>Open Source Virtual Reality, OSRV</i> ) — это инициатива, поддерживаемая несколькими корпоративными партнерами. В настоящее время <i>OSRV</i> предлагает <i>Hacker Development Kit (HDK)</i> в виде <i>HDK 1.3</i> и <i>HDK 2</i> , разработанных компанией <i>Razer</i> . Обе гарнитурки обеспечивают отслеживание положения и вращения

В *UE4* контроллеры перемещения поддерживаются с помощью единого компонента *Motion Controller Component*, который упрощает взаимодействие, в том числе с несколькими контроллерами. Список контроллеров перемещения, поддерживаемых *UE4*, приведен в таблице 1.2.

Таблица 1.2. Совместимые контроллеры

Контроллеры перемещения	Описание
<i>Oculus Touch</i>	Контроллеры <i>Oculus Touch</i> позволяют отслеживать как вращение, так и положение при помощи системы <i>Constellation</i> . Каждый контроллер имеет пару кнопок ( <i>A</i> и <i>B</i> на правом, <i>X</i> и <i>Y</i> на левом контроллере) и аналоговый джойстик под большой палец, кнопку под указательный палец («спусковой крючок») и кнопку для остальных пальцев. Поддержка этих контроллеров осуществляется через <i>Oculus SDK</i>
<i>Vive</i>	Контроллеры <i>Vive</i> позволяют отслеживать вращение и перемещение при помощи системы <i>Lighthouse</i> . Каждый контроллер содержит круговой трекпад, который работает как аналоговый джойстик или кнопка, а также кнопку под указательный палец («спусковой крючок»), кнопку меню и кнопку «Захват». Работа с контроллерами <i>Vive</i> обеспечивается при помощи <i>OpenVR SDK</i>

Контроллеры перемещения	Описание
<i>PlayStation Move</i>	Контроллер <i>Move</i> позволяет отслеживать вращение и перемещение при помощи камеры <i>PlayStation</i> и светящегося шара на самом контроллере. Помимо обычных для контроллеров <i>PlayStation</i> кнопок <i>Cross</i> (крест), <i>Circle</i> (окружность), <i>Triangle</i> (треугольник), <i>Square</i> (квадрат), <i>Start</i> (запуск) и <i>Select</i> (выбор), контроллер <i>Move</i> имеет кнопку <i>Move</i> (перемещение), а также «спусковой крючок» (или кнопку <i>T</i> )
<i>Daydream</i>	Контроллер <i>Daydream</i> поддерживает только отслеживание вращения. У контроллера также имеется сенсорная панель, которая может выступать в качестве кнопки, и кнопка <i>App</i> под ней. Работа с контроллером <i>Daydream</i> осуществляется при помощи <i>Google VR SDK</i>

### 1.1.1.1. Программное обеспечение

Существует множество программных библиотек, *SDK\** и *API\*\**, помогающих взаимодействовать с оборудованием виртуальной реальности. Идеология *UE4* построена на повышении уровня абстракции до единых аппаратно-независимых интерфейсов и компонентов, что облегчает совместимость. При необходимости в ваших руках остается возможность прямого взаимодействия с различными *SDK*. Вам как разработчику будет полезно ознакомиться различиями в концепциях проектирования при помощи этих *SDK*, поскольку при разработке игры или реализации другого проекта могут требоваться преимущества конкретных функций выбранного *SDK*. Для работы с пакетами *SDK*, вам не нужно скачивать какие-либо отдельные файлы, поскольку *UE4* включает их при скачивании модулей.

Список и описание совместимых *SDK* приведены в таблице 1.3.

Таблица 1.3. Совместимые *SDK*

SDK	Описание
<i>Oculus PC SDK</i>	<i>Oculus PC SDK</i> обеспечивает <i>UE4</i> всей необходимой информацией для рендера подходящего представления. Эта информация включает в себя положение и ориентацию головы пользователя, а также его межзрачковое расстояние ( <i>interpupillary distance, IPD</i> ). <i>UE4</i> может сгенерировать кадр и загрузить его в <i>Oculus compositor</i> , где будут применены искажения, необходимые для настройки линз <i>VR</i> -шлема. К счастью, движок в большинстве случаев справляется с этим и не требует от вас вмешательства. <i>Oculus SDK</i> также предоставляет слои, которые дают вам возможность рисовать на экране объекты, имеющие разное разрешение. Это полезно для пользовательского интерфейса ( <i>UI</i> ), где может потребоваться, например, отобразить текст с разрешением выше, чем у фона. Такие функции, как <i>ATW (Asynchronous Timewarp)</i> ; см. таблицу 1.4) автоматически обрабатываются во время выполнения и не требуют от вас дополнительных действий

\* *Software Development Kit* — комплект для разработки программного обеспечения. — Прим. ред.

\*\* *Application Programming Interface* — интерфейс прикладного программирования. — Прим. ред.

SDK	Описание
<i>Oculus Mobile SDK</i>	<i>Oculus Mobile SDK</i> предоставляет доступ ко многим возможностям, соответствующим <i>PC SDK</i> , включая информацию о положении головы и другую пользовательскую информацию, необходимую для рендеринга. Как и среда исполнения <i>PC</i> , мобильная среда исполнения использует <i>ATW</i> , но она дополнительно поддерживает рендеринг переднего (или кадрового) буфера, также известный как <i>scanline racing</i> (см. таблицу 1.4)
<i>Oculus Audio SDK</i>	<i>Oculus Audio SDK</i> предоставляет возможности реалистично распределить звук в пространстве и создать 3D-звук для <i>VR</i> . Этот <i>SDK</i> позволяет применять <i>HRTF*</i> ( <i>head-related transfer function</i> ) к аудиоисточникам для имитации направленного звука (посредством моделирования геометрии человеческого уха). Он также может имитировать расстояние от звука до уха, гася громкость. <i>HRTF Oculus Audio SDK</i> реализован в <i>UE4</i> , позволяя вам настраивать кривые, ограничивающие распространения звука. Однако в настоящее время реализовать их можно только в проектах на базе <i>DirectX</i> (например, <i>PC</i> )
<i>OpenVR SDK</i>	<i>OpenVR SDK</i> во многом схож с <i>Oculus PC SDK</i> и предоставляет доступ к информации о <i>VR</i> -шлеме (включая положение и ориентацию в пространстве). <i>OpenVR SDK</i> также предоставляет интерфейсы для доступа к системе <i>Chaperone</i> (описывающей виртуальные границы, обозначающие игровое пространство для пользователя) и <i>Overlay</i> для генерации <i>2D</i> -контента в компоновщике аналогичном слоям <i>Oculus</i> . <i>OpenVR</i> также поддерживает перепроецирование (аналогично <i>ATW</i> ) и предсказание
<i>OSRV SDK</i>	<i>OSVR SDK</i> дает вам доступ к набору интерфейсов для создания <i>VR</i> -сюжетов. Построенный по принципу «поддержка всего», данный <i>SDK</i> включает наиболее значимые интерфейсы для получения такой информации <i>VR</i> -шлема, как положение и ориентация в пространстве и интерфейсы для отслеживания глаза, все-направленных беговых дорожек ( <i>omnidirectional treadmills</i> ), жестов и так далее. <i>Unreal Engine</i> интегрируется с <i>OSRV SDK</i> , начиная с версии 4.12
<i>Google VR SDK</i>	<i>Google VS SDK</i> дает вам доступ к данным отслеживания положения головы <i>VR</i> -шлема и набор функций для оптимизации вашего <i>VR</i> -проекта. Это позволяет включить устойчивый режим производительности, что важно для запусков на смартфоне, потому что смартфон может перегреться от высокой нагрузки и резко снизить производительность, что может плохо повлиять на восприятие пользователем <i>VR</i> . <i>Google VS SDK</i> также предоставляет доступ к функциям <i>scanline racing</i> , которые оптимизируют задержку в линии рендеринга (см. таблицу 1.4)

Помимо *SDK* и библиотек, которые *UE4* использует для взаимодействия с различной *VR*-аппаратурой, в ядре также реализованы некоторые программные функции, позволяющие значительно улучшить впечатления пользователя от *VR*. Однако существуют и другие программные функции (например, *ATW*), реализованные в различных средах исполнения, как следствие, включенные в ядро по умолчанию и неподконтрольные разработчикам. Таблица 1.4 содержит примеры функциональности обоих видов, доступных в *UE4*.

\* *Head-related transfer function* — передаточная функция, описывающая положение источника звука относительно слушателя. — Прим. ред.