S.T.A.L.K.E.R.: Чистое небо — интервью о проблемах выживания искусственного интеллекта в Чернобыльской Зоне

Материал из xrWiki

Статья скопирована с GAMETECH

Пока S.T.A.L.K.E.R. спешит обзавестись дополнением в виде приквела «Чистое небо», пока разработчики скупятся на новую информацию («старую» можно почерпнуть отсюда), самое время вспомнить прошлое и поговорить об одной из самых интересных особенностей прогремевшего в 2007 году шутера — системе симуляции жизни под названием A-Life. Предупреждаем сразу: предлагаемая вашему вниманию информация больше технического плана, требует внимательного отношения и мало-мальского опыта программирования. Она была опубликована на профессиональном портале AiGameDev.com, а мы с разрешения GSC Game World приводим вам русскоязычный вариант этого огромного интервью.

Из него вы сможете узнать, почему некоторые ранние обещания не были выполнены, почему от многих идей (реализованных на практике) пришлось отказаться, и какой сложный путь проделали разработчики, пока не пришли к финальному варианту. И на закуску — немного тематической информации о «Чистом небе».

Вопрос: Здравствуйте, Дмитрий. Спасибо, что вы нашли время ответить на наши вопросы. Не могли бы вы кратко представиться и рассказать нам немного об ИИ и разработке компьютерных игр.

Дмитрий Ясенев: Здравствуйте. Меня зовут Дмитрий Ясенев, я ведущий программист на проекте S.T.A.L.K.E.R.: Чистое Небо в GSC Game World.

На курсе искусственного интеллекта в университете я впервые познакомился с ИИ для игр на доске. Потом этот интерес перерос в создание программы, которая умеет играть в реверси, она не стала лучшей, но выиграла один чемпионат по синхронным случайным реверси. По окончании университета в 2002-м году, я начал работать программистом искусственного интеллекта в компании GSC Game World на проекте S.T.A.L.K.E.R.: Shadow of Chernobyl.

Вопрос: Одна из ключевых особенностей S.T.A.L.K.E.R. – это независимая система «A-Life». Я допускаю, что много разных компонентов отвечают за создание этой системы, но давайте начнем сначала. Можете описать, как работает система ИИ у персонажей?

Дмитрий Ясенев: Суть её заключается в том, что персонажи в игре живут своей жизнью и существуют всё время, а не только когда их видит игрок. Это идёт вразрез с привычными оптимизациями, используемыми при разработке игр (зачем делать действия, которые не видны игроку?). Поэтому такую систему имеет смысл использовать только тогда, когда вы чётко знаете, что вы хотите от неё получить. У нас были требования от гейм-дизайнеров получить персонажей, которые жили бы не только в пределах одного уровня, а могли и передвигаться между ними, запоминая информацию, полученную во время своей жизни. В результате, мы

решили, что тогда, в игре, каждому персонажу должна соответствовать одна логическая сущность, вне зависимости от того, на каком уровне он находится, в то время как мы могли попробовать реализовать это с помощью разнообразных трюков.

Вопрос: У вас присутствует тонкая грань между игрой и симуляцией. Насколько глубоко реализованы идеи симуляции жизни в S.T.A.L.K.E.R.?

Дмитрий Ясенев: Наша реализация системы симуляции жизни не претендует на звание полной и единственно правильной. Концепция живого мира, в котором каждый персонаж имеет свою цель, очень богата и разнопланова. Поэтому реализовать её в полной мере — задача непростая и объёмная. Особенно если мы говорим о шутерах, в которых детализация действий находится на высоком уровне, и красивость отыгрыша анимации может иметь большее значение, чем высокоуровневые действия персонажа. Об этом много говорят. Например, обсуждался вопрос о том, почему приёмы ИИ в RTS не переходят в FPS. Я бы ещё поинтересовался у разработчиков ИИ в шутерах, сколько времени они тратят на низкий уровень и сколько на высокий, мой прогноз: хорошо, если будет 10 к 1.

Вопрос: Не могли бы вы рассказать нам более подробно, как вы реализовали систему симуляции жизни?

Дмитрий Ясенев: Наша реализации симуляции достаточно проста. Мы ввели два термина, характеризующие 2 модели поведения персонажа, отличающихся степенью детализации: оффлайн и онлайн. Оффлайновое поведение персонажа является очень простым с точки зрения детализации: персонаж не отыгрывает анимации, звуки, не управляет активно инвентарём, не строит детализированные сглаженные пути (хотя строит пути по глобальному навигационному графу, но об этом позже) и т.д. Онлайновое поведение напротив имеет полную степень детализации. Т.о. можно считать, что оффлайновое поведение является лодом онлайнового.

В нашей системе, пока игрок играет на своём уровне, другие персонажи живут на других уровнях, т.е. находятся в оффлайне, т.е. используют оффлайновое поведение. Более того, в виду большой населённости, не все персонажи в пределах одного уровня имеют онлайновое поведение, а лишь те, кто находятся в заданном радиусе от игрока (это может зависеть от уровней, обычно в районе 150 метров) или же по желанию геймдизайнеров.

Для реализации этого симулятор следит за передвижением игрока и объектов в оффлайне и переводит их в онлайн/оффлайн. При вычислении перехода объектов используется стандартный трюк с инерцией: радиус перехода в оффлайн больше радиуса перехода в онлайн.

Далее стоит сказать о навигации объектов в онлайне и оффлайне. У нас в игре есть уровни, для каждого из которых создаётся свой навигационный граф, который используют персонажи для передвижения в онлайне. Мы называем его детальным графом. Для каждого детального графа также создаётся его менее детализированный аналог, вершины которого можно связать с вершинами такого же графа другого уровня/ей. Т.о. после объединения всех таких графов воедино мы получаем граф, который объединяет все уровни. Он и используется персонажами для передвижения в оффлайне. Также им пользуются персонажи в онлайне, когда они выполняют свои стратегические цели. Например, если персонаж в онлайне решил идти на другой уровень, то он строит путь по глобальному графу, затем строит путь по детальному графу своего уровня со своей позиции до точки глобального графа. Если эта точка уже на другом уровне, то он телепортируется туда и автоматически переходит в оффлайн. Для того, чтобы это не происходило на глазах у игрока, мы располагали точки перехода для NPC где-то «за углом», дальше точек перехода для игрока.

Вопрос: Какие еще важные составляющие A-Life системы вы использовали для разработки S.T.A.L.K.E.R.?

Дмитрий Ясенев: У персонажа в игре всегда есть цель. На ранних версиях симулятора это была цель разгадать загадку зоны. Персонаж знал одного или нескольких торговцев, которые имели набор заданий, которые они генерировали, исходя из карты аномальной активности и запроса организаций, которые хотели нажиться на найденных в зоне артефактах. Выполнение задания приближало персонажа к его цели. На тот момент был только один тип заданий – принести артефакт. Персонаж брал задание на поиск артефакта, шёл в место, где он приблизительно мог быть, судя по данным задания, если находил – возвращался к торговцу, торговал с ним и выбирал новое задание. По пути он мог встретить противников и воевать с ними, при этом расчёт в оффлайне вёлся на манер TBS: каждый противник делал ход по очереди, результат действия вычислялся по формуле + рандом. В результате, получалось, что иногда один от другого мог сбежать, иногда кто-то кого-то убивал, а иногда они даже друг друга не замечали или не решались нападать. Если же персонаж был сталкером и встречал нейтралов или друзей, то торговал с ними по разработанной схеме торговли. Всё это работало как в оффлайновой, так и в онлайновой схемах.

Планировалось, что потом количество типов заданий увеличится, а поведение персонажа станет более разнообразным за счёт его повседневных потребностей во сне и пище. Мы даже планировали, что наши персонажи смогут сами разгадать загадку Зоны раньше игрока, если соберут определённое количество информации.

Но потом мы поменяли схему генерации заданий. Персонажи в игре стали получать задания не от торговцев, а от т.н. smart terrain-ов (это не то же самое, что в симсах). Т.е. теперь жизнь персонажа выглядела таким образом, что он выбирал задание, сгенерированное каким-то smart terrain-ом, и шёл на место, указанное в нём. После прихода ему начислялись очки, а smart terrain забирал персонажа под свой контроль на какое-то время. Находясь под контролем smart terrain-а, персонаж выполнял работы, которые были доступны, и которым он подходил с учётом приоритетов. Так в игре появились базы группировок, сталкеры возле костров и т.д.

Т.о. миграции персонажей с места на место и передвижение их между уровнями были обусловлены сменой задания.

Вопрос: Расскажите нам об интеллекте на уровне существа.

Дмитрий Ясенев: По поводу интеллекта на уровне существа – стоит рассмотреть две модели: оффлайновую и онлайновую отдельно, хотя они и имеют общее.

Оффлайновый интеллект существа выглядит очень просто: если нет выбранного задания – попробовать его выбрать. Если выбрать не получилось – бесцельно бродить. Если есть задание – идти на место его выполнения. Когда персонаж под контролем смарттеррейна, ему могут быть отданы дополнительные команды на перемещение, но никакие другие действия он не выполняет.

Онлайновый интеллект персонажа состоит из трёх слоёв:

- принятие информации и её обработка
- принятие решений
- множество низкоуровневых контроллеров

Персонаж имеет 3 вида перцепторов: визуальный, звуковой и получение повреждения. По информации, полученной от перцепторов, персонаж выбирает интересную ему информацию – враги, опасности, предметы, которые можно подобрать.

Вопрос: Как персонажи в S.T.A.L.K.E.R. принимают решения?

Дмитрий Ясенев: Модель принятия решений претерпела множество изменений во время процесса разработки. Всего было 4 итерации. Сначала мы использовали КА со стэком. Затем

мы стали использовать иерархические КА. Затем я прочитал замечательную статью GOAP (Goal Oriented Action Planning) Джефа Оркина в Game Programing Wisdom II, затем там же статью о мотивационных графах, и мы стали использовать мотивационные графы для выбора целей, а GOAP для выбора действий.

Есть несколько нюансов, которые я обсуждал в переписке с Джеффом Оркиным. Мы решили использовать планирование исключительно для выбора действий. Таким образом, нам нужен план только для получения первого его действия. План сам по себе не используется. Ценность выбранного действия в том, что оно выбрано в контексте плана кратчайшего веса, который достигает заданную цель(и). Это очень удобно для отладки – вы всегда знаете не только, что делает ваш персонаж, но, что более важно, почему.

Также радикально мы решали и проблему актуальности планов – наши планы всегда актуальны. Для гарантирования этого свойства, наша реализация планировщика GOAP опрашивает свойства мира, значения которых использовались при построении плана кратчайшего веса. Мы не перестраиваем планы постоянно, но лишь в том случае, если хотя бы одно из опрашиваемых свойств поменяло своё значение (на самом деле, есть ещё несколько условий – добавление/удаление операторов/свойств мира, изменение весов операторов).

С другой стороны, персонаж должен всегда иметь непустой план. Пустой план означает, что персонаж не знает, что ему делать, т.к. не существует последовательности действий, которые могут перевести текущее состояние мира в целевое (либо целевое состояние уже достигнуто) с помощью множества доступных операторов (либо же его достижение потребовало слишком много вершин, что случается крайне редко. В 99% случаев это означает, что такой последовательности действий не существует, но планировщик не может доказать это из-за ограничения на количество посещенных вершин). Условие существования непустого плана обязывает нас иметь недостижимые свойства мира, что, наверное, не слишком элегантно.

Для задания поведения сталкера мы использовали несколько десятков операторов (около 70). Для удобного менеджмента такого количества операторов мы использовали иерархии планировщиков. Таким образом, оператор сам мог быть планировщиком. Взаимодействие планировщиков разных уровней осуществлялось следующим образом: если планировщик верхнего уровня изменил план, и в нем первый оператор отличался от первого оператора предыдущего плана, то этот старый оператор уведомлялся о своем завершении. Если оператор был к тому же и планировщиком, он информировал свой текущий оператор о завершении и т.д. Стоит заметить, что окончание оператора было мгновенным действием. Таким образом, если действие не могло быть остановлено таким образом, это должно было быть учтено не только на его уровне, но и на всех уровнях выше, что добавляет дополнительную сложность.

Именно поэтому мы решили не использовать GOAP для задания поведения монстров, т.к. они существенным образом зависели от анимаций, которые не могли быть прерваны или «погашены». Поэтому мы использовали для них иерархический КА, хотя, на самом деле, он не решил проблему немгновенного окончания действия. В приквеле мы получили решение этой проблемы, переместив часть логики в низкоуровневые контроллеры: высокоуровневая логика задает цели низкоуровневой. Благодаря этому, мгновенное изменение оператора в высокоуровневой логике не означает мгновенное изменение действия на низком уровне.

Есть также нюанс в комбинировании двух методов задания логики. Сначала мы решили использовать мотивационные графы для уменьшения нагрузки на планировщик GOAP-а. Однако, оказалось, что планировщик отлично справляется со своей задачей и поиск никогда не посещает более 200 вершин. Кроме того, задание топологии графа для выбора целей переносит часть работы планировщика на создателя графа. В итоге мы перестали использовать мотивационные графы, т.к. удобнее задавать логику одним способом. Разные цели использовались только для живых и мертвых персонажей (для зажатия спускового крючка во время смерти).

Низкоуровневые контроллеры отвечают за выбор анимаций, передвижение, управление объектами, проигрыванием звуков, управление ориентацией персонажей. Их реализация не

представляет особого интереса, кроме, разве что, управления объектами, для которого мы использовали GOAP (и очень успешно). Было бы интересно посмотреть на менеджер низкоуровневых контроллеров, который использовал бы частично упорядоченное планирование для управления. Тут, конечно, не стоит забывать, что скорость нам очень важна, т.к. обычно низкоуровневые контроллеры обновляются несколько раз в секунду. С другой стороны, такое решение взяло бы на себя все внутренние взаимодействия между котроллерами. Возможно, такое планирование стоит использовать не только для низкоуровневых контроллеров, но и для задания всей логики персонажа, как уже обсуждалось здесь.

Вопрос: Поскольку в финальной игре был сюжет, значит некоторые территории в игре, были скриптоваными, в традиционной манере. Не могли бы вы рассказать нам как вы объединили систему A-Life со скриптами?

Дмитрий Ясенев: Как уже было сказано выше, задания в игре генерируются smart terrain-ами. Это же касается и сюжетной линии. Сюжетные задания имеют больший приоритет. Кроме того, сюжетные области ограничены т.н. space restrictor-ами. Их задача заключается в том, чтобы сюжетные персонажи не разбегались слишком далеко, а несюжетные персонажи не ломали сюжетные элементы. Т.е. в этом доля контроля над симуляцией есть, и доля немалая. С другой стороны, после прохождения игроком сюжетной сценки, сюжетные персонажи становятся обычными и следуют обычным правилам, т.е. выбирают себе задания и идут их выполнять в соответствующие смарт террэйны, а с самой зоны снимаются все ограничения и на неё разрешается выдавать задания.

Вопрос: Тяжело ли было дизайнерам интегрировать эти скриптованые территории вместе с A-Life? Как они подошли к процессу?

Дмитрий Ясенев: Трудно было без ограничителей пространства. Тогда любой «пришлый» персонаж либо ломал сюжетную сценку, отвлекая на себя персонажей, либо, если им было запрещены реакции на «пришлых», случались дурацкие ситуации, когда сюжетного персонажа грызёт собака, а он на неё никак не реагирует и в результате погибает при полном обмундировании.

Ограничителями пространства со временем полностью решили эту проблему.

Вопрос: Оглядываясь назад, какими частями системы и/или процессом разработки, вы особенно довольны?

Дмитрий Ясенев: Я был очень счастлив, когда я прошёлся за сталкером с одного уровня на другой, смотрел, как он ищет артефакты, находит их, потом возвращается на уровень к торговцу, подходит, торгует, выбирает новое задание и идёт дальше – жаль, что это не вошло в оригинальную игру, наблюдать за этим было очень интересно.

Также на заре внедрения GOAP, было очень интересно наблюдать за двумя враждующими безоружными персонажами, которые знали где находятся оружие и боеприпасы: сначала они бегут к оружию, затем, обнаружив, что в нём нет боеприпасов, они бегут к боеприпасам, тот, кто первый успел, срывает второго в панику, если до оружия/боеприпасов ещё далеко, убегающий в панике получает несколько ранений и падает раненым, а второй идёт его добивать (это так и не пошло в оригинал в виду того, что у всех персонажей сделали неограниченное количество боеприпасов и такие ситуации просто не случались).

Было очень интересно следить за битвой нескольких собак и одного сталкера – это просто захватывающее зрелище, когда ты видишь картину, как он, раз за разом перевыбирая укрытие, спасается от прыгающих на него сзади собак. Случалось, что он погибал практически сразу, а случалось, что и убивал 4-5 собак, а иногда и всех 6-х.

Вопрос: А как насчет разочарований? Было ли что-то в системе или ее разработке, чтобы вы хотели изменить?

Дмитрий Ясенев: По симуляционной части: я бы очень хотел поиграть в игру, в которой персонажи жили бы своей жизнью, каждый имел бы свою цель в игре, у каждого были нормальные человеческие (или специфичные, если речь идёт о монстрах) потребности которые персонажу необходимо выполнять. Сейчас вместо составления алгоритма выбора текущих желаний и их удовлетворения, был использован более простой вариант, который имитировал потребности персонажей, переложив эту задачу на smart terrain-ы, которые отвечали за смену работ персонажами и могли иметь работу, например, «поспать», что, конечно, не одно и то же.

По визуальной части: есть большое поле для улучшений боевых действий, командной работы, эффективности боя и эффектности действий персонажей.

Разное: также очень хотелось научить персонажей обходить динамические препятствия, друг друга, разнообразно использовать terrain (пути и смарты), уметь водить машины, управлять вертолётом и т.д. Есть ещё много вещей, которые они по разным причинам ещё не умеют, но в следующих итерациях нашей игры мы попробуем постепенно добавлять недостающие элементы.

Вопрос: Как насчет приобретенного опыта? Если бы у вас было 30 секунд, для того чтобы дать совет другим разработчикам, чтобы вы посоветовали?

Дмитрий Ясенев:

1. Не изобретать колёса. Используйте Интернет для поиска решений возникших у вас проблем.

2. Чётко знать, какую игру вы делаете, что в ней будет и чего не будет, использовать прототипирование для того, чтобы избежать большого количества фич, которые не войдут в релиз.

3. Отлаживайте персонажей с удобными инструментами: каждая компонента должна иметь отладочную отрисовку/режим/экран. Рисуйте пути (все итерации, все стадии сглаживания), рисуйте проверки на видимость, чтобы знать, почему персонаж не видит, рисуйте информацию об укрытиях и т.д. Для отладки поведения персонажей мы создали отладочный экран со всеми данными, так что мы можем легко выбрать персонажа и выяснить что с ним не так. Пауза и замедление времени бесценны для отладки анимаций/передвижения/ориентации.

4. Пусть программисты программируют, а дизайнеры – дизайнят. Дизайнер – плохой программист, а программист – плохой дизайнер. Это может быть проблемой при использовании скриптов. Пусть люди занимаются тем, в чём они сильны. Вместо обучения дизайнеров программированию, сделайте WYSIWYP редактор с кучей настроек.

5. Будьте в курсе всех новых веяний и течений в вашей области, читайте интернет-ресурсы.

6. Я бы пожелал каждому разработчику в каждой своей игре пробовать что-то новое. Возможно, всего одну вещь (т.к. несколько сразу могут выйти за рамки одного проекта), но обязательно новую. Мы видим прекрасную реализацию деревьев решений в Black & White, Jeff Orkin сделал отменную презентацию планировщиков GOAP в F.E.A.R., мы попробовали сделать в каком-то приближении A-Life в шутере, завтра кто-то сделает ещё что-то новое - от этого выиграют как игроки, которые получат разные игры, так и сами разработчики, которые хорошим примером обратят внимание коммьюнити на интересный метод/подход.

Вопрос: Можете рассказать нам немного о приквеле к S.T.A.L.K.E.R., над которым вы сейчас работаете? Какие изменения вы внесли в A-Life технологию?

Дмитрий Ясенев: Сюжет «S.T.A.L.K.E.R.: Чистое Небо» переносит игроков за год до событий

оригинальной игры S.T.A.L.K.E.R. в 2011 год.

Группа сталкеров впервые подобралась к самому сердцу Зоны - Чернобыльской АЭС, спровоцировав катаклизм, который едва не привел к катастрофе. Грандиозный Выброс аномальной энергии изменяет Зону. Проверенных и относительно безопасных дорог больше нет. Целые уровни пропадают в сполохах аномалий. Сталкеры и даже экспедиции гибнут или оказываются запертыми на пропавших территориях. Новые территории, о которых ничего не было известно со времени появления Зоны, появляются на карте Зоны. Зону продолжают сотрясать выбросы. Зона нестабильна. Аномальная активность максимальна.

Изменения известной сталкерам карты Зоны нарушает хрупкий баланс сил в Зоне. Между группировками разгорается вражда за новые территории, поля артефактов и сферы влияния. Больше нет старых врагов или друзей - теперь каждый сам за себя. Между фракциями началась Война Группировок.

Главный герой - наемник, волей судьбы оказавшийся на самом острие противостояния между фракциями сталкеров, Стрелком и даже самой Зоной. Главному герою предстоит сыграть основную роль в событиях, которые привели историю Зоны к той точке, с которой начинается оригинальная игра S.T.A.L.K.E.R.

Что ждет сталкеров в открывшихся глубинах Зоны? Какие новые опасности таятся на новых территориях? Почему выбросы сотрясают Зону? Почему изменилась Зона? Как устранить ее нестабильность? Какая фракция одержит верх в противостоянии группировок? Почему Стрелок попал в грузовик смерти? Что произошло со Стрелком до этого? Был ли другой выбор? На эти и многие другие вопросы Вы найдете ответ в официальном приквеле «S.T.A.L.K.E.R.: Чистое Небо».

Изменения коснулись и роли группировок в симуляции. Т.е. если раньше это был smart terrain, один из сотни, который генерировал задания, по условиям которых на их выполнение подходили только персонажи из конкретной группировки, то теперь группировка является игровой сущностью, она решает свои задачи в симуляции, борется с другими группировками, а игрок может ей помочь в этом и сразу увидеть результат своих действий.

Также мы существенно изменили схему поведения персонажей в бою. Теперь они воюют более эффектно и эффективно.

Источник — «https://xray-engine.org/index.php?title=S.T.A.L.K.E.R.:_Чистое_небо_—_интервью_о_проблемах_выживания_и_искусственного_интеллекта_в_Чернобыльской_Зоне&oldid=813»

Категория:

Интервью

- Страница изменена 19 мая 2018 в 10:46.
- К этой странице обращались 1241 раз.
- Содержимое доступно по лицензии GNU Free Documentation License 1.3 или более поздняя (если не указано иное).