Санкт-Петербургский государственный университет

Кафедра Системного программирования

Группа 21.Б10-мм

Разработка инструмента для визуализации виртуальной дороги

Минязев Роман Русланович

Отчёт по учебной практике
в форме «Решение»

Научный руководитель:
доцент кафедры системного программирования,
к.т.н., Литвинов Юрий Викторович

Консультант:
инженер-программист, АО «Кама»,
Осечкина Мария Сергеевна

Санкт-Петербург
2022

Оглавление

Введение **3**

1. Постановка задачи **4**

2. Обзор **5**

2.1. Blender 5

2.2. Carla 8

3. Реализация **11**

3.1. Используемые инструменты 11

3.2. Архитектура 11

4. Эксперимент **14**

Заключение **15**

Список литературы **16**

Введение

С каждым годом растет количество автомобилей. Для повышения безопасности дорожного движения необходима разработка системы дополненной реальности для помощи водителю на дороге (ADAS). Среди функций ADAS предупреждение о покидании полосы, предупреждение о столкновении, дополненная навигация и прочие. В системах ADAS используются различные датчики, в том числе камера.

Анализируя изображение с камеры, можно попытаться решить некоторые задачи, однако для таких целей, как расчет расстояний до объектов, одного кадра с камеры будет недостаточно, т.е. обнаружить изменение профиля дороги при спуске и подъеме становится невозможным. Кроме того, если дорога достаточно кривая (её границы представляют собой недостаточно длинные отрезки, например, горная местность), то распознать сужение или расширение дороги также становится трудной задачей.

Для решения этих проблем нужно реализовать инструмент для визуализации проекции виртуальной дороги на плоскость камеры, который будет использоваться для анализа построения профиля дороги. Используя различные модели дорог, представления их границ и анализируя получившуюся проекцию, можно сделать вывод, какая модель дороги лучше описывает реальность, и использовать ее в ADAS алгоритмах.

1. Постановка задачи

Целью работы является реализовать инструмент для визуализации проекции виртуальной дороги на плоскость камеры.

Для её выполнения были поставлены следующие задачи:

1. Провести обзор существующих решений.
2. Реализовать абстракцию модели дороги, границы которой представляют из себя разные объекты (кривые второго порядка, отрезки прямых и т.д.).
3. Реализовать архитектуру так, чтобы представление границ дороги было легко поменять: например, в первом используются сплайны, а во втором кривые второго порядка.
4. Реализовать абстракцию камеры.
5. Задать движение камеры по дороге, проецируя при этом точки дороги на картинную плоскость камеры.
6. Реализовать GUI, для того чтобы пользователь мог изменять угол наклона камеры, ее сдвиг и модель дороги.
7. Провести оценку удобства использования инструмента.

2. Обзор

В настоящее время существует множество инструментов для моделирования и рендеринга объектов. Среди инструментов для обзора выбирались те, в которых можно смоделировать дорогу и получить её изображение. Поиск производился по запросам «3d visualization tools» и «3d visualization tools for automotive roads» в поисковой системе Google по первым 20 ссылкам.

2.1. Blender

Blender [1] — программа для создания трёхмерной компьютерной графики и рендеринга. Основные преимущества:

- Бесплатное программное обеспечение.
- Удобный пользовательский интерфейс.
- Кроссплатформенность.
- Небольшой размер исполняемого файла.
- Множество параметров для настройки рендеринга.

2.1.1. Моделирование в Blender

В Blender присутствуют объекты следующих типов:

1. Кривые: кривая Безье и кривая NURBS (неоднородный рациональный B-сплайн).
2. Меш: куб, плоскость, окружность, цилиндр, конус.
3. Поверхности: сфера, тор, цилиндр, NURBS-поверхность.

Все кривые в Blender по сути являются сплайнами. Отличие кривых Безье от кривых NURBS заключается в том, что кривая Безье проходит через направляющие точки. Получается, что для задания разных

моделей кривых так или иначе придется пользоваться сплайнами. То есть у пользователя появляется возможность смешивать разные модели границ для построения дороги, что является минусом, потому что одно из применений визуализатора дороги — понять, насколько хорошо конкретная модель описывает реальность. Также, для изменения представления границ или сужения и расширения дороги, её придется долго перестраивать вручную.

2.1.2. Анимация в Blender

В анимации можно задать движение камеры по сцене. Перемещая камеру в мировой системе координат и меняя ее наклон в зависимости от желаемого результата, необходимо задать ключевые кадры. Цель ключевых кадров — сделать интерполированную анимацию. То есть пользователю достаточно указать два ключевых кадра, выбрать метод интерполяции и на всех кадрах между ключевыми Blender автоматически должен определить положение объектов. Результат применения такого метода [5]. В качестве метода интерполяции был выбран линейный (с постоянной скоростью у камеры). Дорога, смоделированная в Blender представлена на рис. 1.

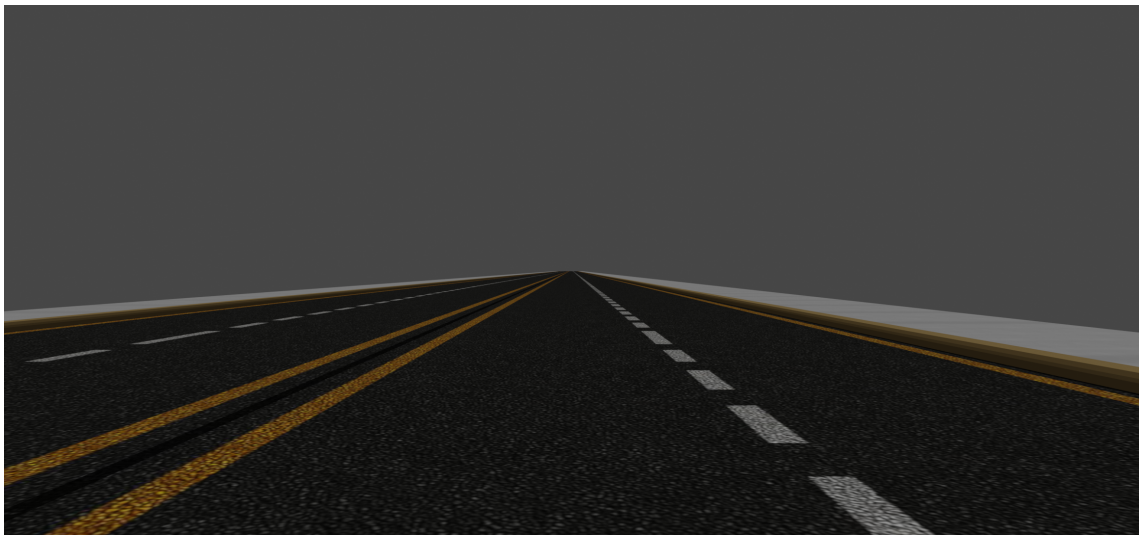

Рис. 1: Изображение дороги

Еще один способ задать движение камеры — нарисовать сплайн, точки которого находятся на одной высоте относительно дороги и добавить

камере модификатор follow path, указав целью построенный сплайн. Тогда камера будет двигаться вдоль этого сплайна.

Также в анимации можно добавить несколько полезных свойств. Например, шум, который сможет создать тряску камеры (ситуация с проездом лежачего полицейского). Для наложения модификатора шум необходимо выбрать траекторию камеры (траектория камеры представляет из себя сплайн) по одной из осей и выделить интервал, на котором будет действовать этот модификатор.

2.1.3. Рендеринг в Blender

В Blender есть три режима для изменения скорости рендеринга:

1. Default.
2. Faster Render. В этом режиме увеличивается скорость рендеринга и количество используемой памяти.
3. Lower Memory. В этом режиме уменьшается скорость рендеринга и количество используемой памяти.

Также можно влиять на скорость рендеринга с помощью изменения освещения, обработки теней, максимального количества отражений света и прочих параметров.

2.1.4. Наложение и сравнение моделей в Blender

В Blender нет возможности без переноса моделей сравнить их методом наложения, но в видеоредакторе Blender есть возможность наложить модель дороги на видео, тем самым визуально оценив ее качество.

2.1.5. Минусы Blender

Подводя итог, отметим минусы Blender для целей учебной практики:

- Возможность смешивания разных моделей границ, так как все кривые в Blender являются сплайнами.

- Для изменения представления границ дорог придется сделать много ручной работы.
- Нельзя выбрать, с каким шагом проецировать точки дороги на камеру.
- Длительный процесс рендеринга.

Для устранения первых двух минусов можно реализовать генератор дорог, который будет генерировать определенные представления границ и сохранять модели дорог в формате, импортируемом в Blender. Тогда появляется возможность использовать Blender для рендеринга. Выбор ключевых кадров для дороги также можно автоматизировать. Однако сам процесс рендеринга в Blender относительно долгий, что повлияет на производительность программы в целом.

2.2. Carla

Carla [3] — симулятор, созданный для обучения систем автономного вождения и основанный на клиент-серверной архитектуре.

Основные преимущества:

- Бесплатное программное обеспечение.
- Возможность воздействовать на всех актеров при помощи API. Актеры в Carla — элементы, выполняющие действия в симуляции, влияющие на действия других элементов, то есть машины, пешеходы, датчики.
- Большое количество датчиков, таких как RGB-камера, лидар, детектор столкновений, датчик GNSS (глобальная навигационная спутниковая система) и другие.
- Несколько режимов рендеринга.

2.2.1. Возможности Carla

Взаимодействовать с сервером можно посредством API на Python. Используя API, пользователь получает возможность моделировать различные дорожные ситуации и влиять на состояние мира в режиме реального времени: менять количество актеров и их поведение, погодные условия, считывать данные с датчиков, на основе которых корректировать действия автомобиля.

2.2.2. Генерирование карт в Carla

Карта — 3d модель города. Есть список карт, доступных «из коробки», но пользователь может генерировать и импортировать в Carla свои карты. Для этого понадобится Unreal Engine и RoadRunner (или его бесплатные аналоги). RoadRunner — удобный редактор для построения дорог, позволяющий добавлять дорожные знаки, светофоры, ограждения, разметку и прочее. Файлы, сгенерированные RoadRunner (.xodr и .fbx), нужно экспортировать в Unreal Engine. Теперь построенная дорога доступна в Unreal Editor, в котором можно модифицировать карту, добавляя модели (например, здания или автомобили), построенные в различных 3d визуализаторах (в том числе в вышеупомянутом Blender). Для генерирования карт потребуется хорошее аппаратное обеспечение. Помимо симулятора Carla, который займет 32 гб, дополнительно потребуется больше 100 гб (на движок Unreal и остальное ПО) и как минимум 6-8 гб оперативной памяти. Для такой несложной операции, как изменение представления дороги, придется перестраивать карту, и даже если автоматизировать этот процесс, получится долго и ресурсозатратно.

2.2.3. Рендеринг в Carla

В Carla есть возможность выбрать качество графики. Существует два режима: low и epic. В режиме low уменьшается дальность прорисовки и исчезает постобработка, на рендеринг тратится меньше ресурсов. Можно вообще отключить рендеринг, но тогда показатели камеры и других графических датчиков будут нулевыми. Есть возможность за-

кадрового рендеринга — без изображения, но со всеми возможностями ренденринга.

2.2.4. Минусы Carla

- Для изменения модели дороги и ее границ придется перестраивать карту.
- Нет механизма для сравнения разных моделей дорог.
- Потребуется большое количество памяти: 32 гб на Carla и больше 100 гб на остальное программное обеспечение.

Несмотря на перечисленные минусы для целей учебной практики, стоит отметить, что Carla предназначена для решения других задач и отлично с ними справляется.

3. Реализация

3.1. Используемые инструменты

Для реализации инструмента был выбран язык программирования C++, так как он является производительным языком программирования и поддерживает объектно-ориентированное программирование. На C++ реализована библиотека компьютерного зрения OpenCV, которая использовалась для реализации, так как содержит удобный класс, представляющий матрицы и матричные операции, такие как взятие определителя, взятие обратной матрицы, умножение, сложение матриц и функции для получения проекции точки из мировых координат на плоскость камеры, вывода изображения на экран и прочие.

3.2. Архитектура

Диаграмма классов, написанных для реализации инструмента, изображена на рис. 2.

Все примитимы, используемые для моделирования дороги, наследуются от класса Models, который представляет собой абстракцию модели. Каждая модель имеет свою систему координат, и у каждой модели можно запросить её точки в собственной и мировой системе координат. Также модели можно свободно перемещать и вращать вокруг осей. Модели хранятся в формате .obj. Для поддержки этого формата в классе Models есть поле indexes, в котором записаны индексы вершин полигонов. Формат .obj выбран для совместимости с другими визуализаторами.

Камера представлена классом Camera, который тоже наследуется от класса Models, то есть камеру можно перемещать и менять её угол наклона. В классе Camera присутствуют два метода для проецирования точек. Их отличие в том, что с помощью одного можно быстро проецировать набор точек на плоскость камеры, не являющихся моделями. То есть для случая, если хочется быстро спроецировать точки, не инициализируя поле indexes. Если же мы хотим спроецировать модель,

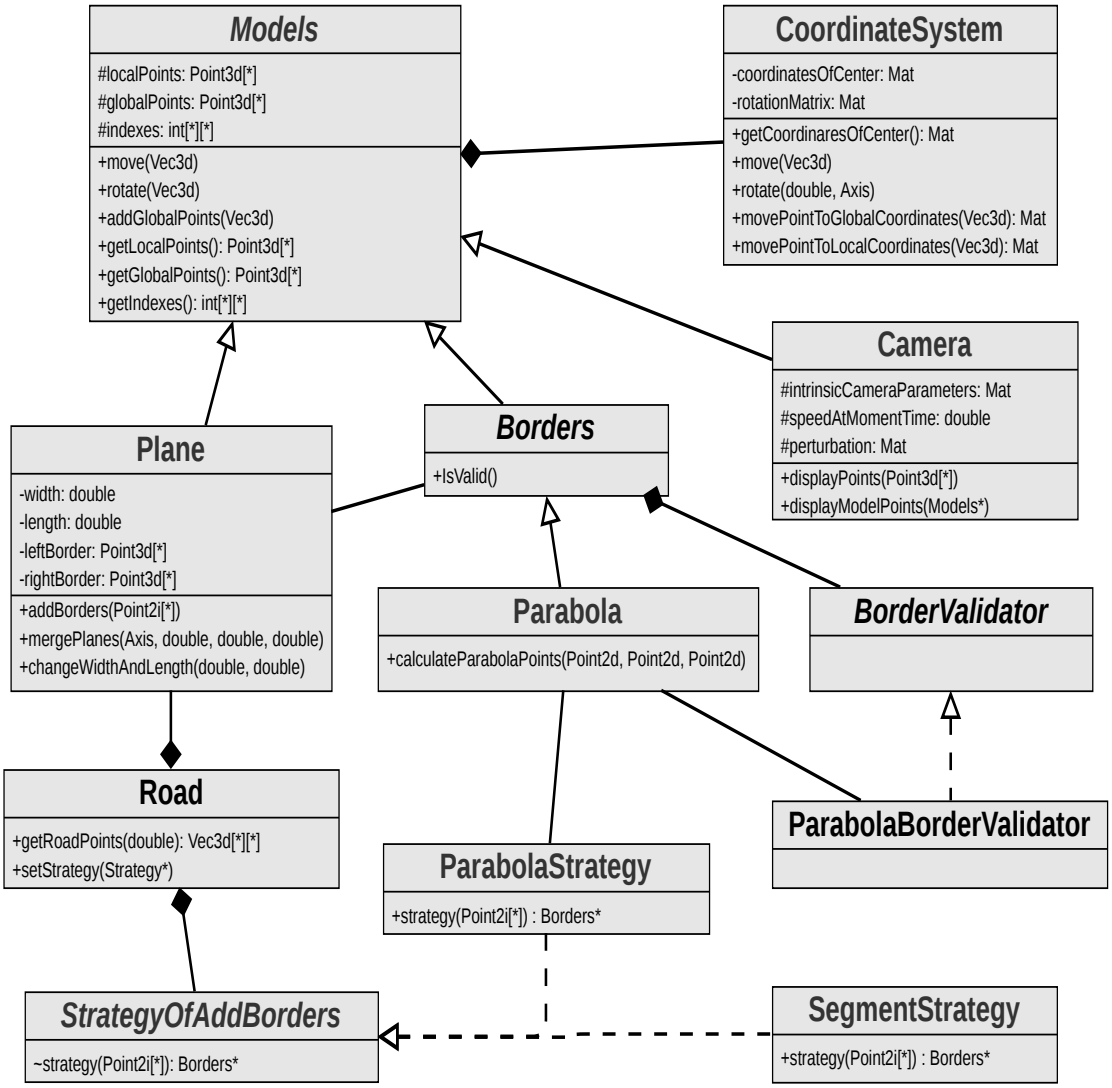

Рис. 2: Диаграмма классов

среди точек которой есть много связей, лучше пользоваться методом displayModelPoints.

Границы дорог представлены классом Borders, от которого наследуется конкретные типы границ. Один из таких типов — парабола, представленная классом Parabola. Точки параболы считаются с помощью метода calculateParabolaPoints по трём точкам, которые пользователь ввел в ручную или отметил кликом мыши по изображению. Восстановление коэффициентов параболы происходит при помощи правила Крамера для решения систем уравнений. Коэффициенты параболы восстанавливаются однозначно, так как при рисовании пользователем точек параболы, известна информация о расстоянии до плоскости, на которой

лежит парабола.

Для валидации границ используется паттерн Мост. В классе Borders могут быть методы для проверки валидности всех типов границ (например, левая и правая граница дороги не могут пересекать сплошную осевую линию дороги), но также должны присутствовать проверки валидности для конкретных типов границ, которые определены в классах наследниках BorderValidator.

Дорога представлена классом Road. Дорогая представляет из себя объединение объектов Plane и Borders. Для генерации плавных подъемов, спусков и поворотов было решено использовать плоскости. Также у дороги можно запросить точки с помощью метода getRoadPoints.

Для быстрого перехода из одного представление границ в другое используется паттерн Стратегия. При смене стратегии стирается информация о предыдущих границах и пользователь может строить границы нового типа.

4. Эксперимент

Основной показатель того, что программа написана хорошо — удобство использования. Оценка удобства использования программы будет проводиться по методике System usability scale [2].

Критерии, по которым будут отбираться испытуемые:

- Принадлежность к определенной сфере профессиональной деятельности.
- Непредвзятость оценивания.

Для непредвзятости оценивания в качестве испытуемых будут выбираться люди, имеющие опыт в оценивании учебных практик или иных работ (например, консультант или научный руководитель).

В качестве заданий, предложенных испытуемым, будут задания по типу:

- Смоделировать дороги, меняя представления их границ. Например: смоделировать подъем, спуск, поворот, сужение или расширение дороги.
- Меняя положение камеры, получить проекции дорог на картинную плоскость камеры.
- Оценить полученные проекции с помощью метода наложения проекции на видео.

Заключение

Текущие результаты:

- Реализована абстракция камеры.
- Реализованы некоторые примитивы (плоскости, поверхности, представляющие из себя объединение плоскостей, параболы, отрезки прямых) и функции для моделирования дороги и ее границ.
- Архитектура реализована так, что представление границ дороги легко поменять.

Планируется сделать:

- Реализовать сплайны и некоторые кривые второго порядка для моделирования границ дороги.
- Реализовать GUI.
- Реализовать механизм для сравнения и наложения дорог.
- Провести оценку удобства использования инструмента.

Реализация инструмента доступна на Github [4].

Список литературы

[1] Blender. Документация Blender. — URL: https://docs.blender.org/ (online; accessed: 2022-11-22).

[2] Brooke John. System usability scale. — URL: https://en.wikipedia.org/wiki/System_usability_scale#cite_note-1 (online; accessed: 2022-12-03).

[3] Carla. Документация Carla. — URL: https://carla.readthedocs.io/en/0.9.13/ (online; accessed: 2022-11-22).

[4] Minyazev Roman. Implementation tool for visualizing a virtual road. — URL: https://github.com/osechkina-masha/adas_spbu/pull/2 (online; accessed: 2022-12-04).

[5] Minyazev Roman. Рендеринг дороги. — URL: https://drive.google.com/file/d/1JFQDLacxZBcdiKa1xEhRJ4L58ZstvqEi/view?usp=sharing (online; accessed: 2022-11-22).