Описание функциональных характеристик программного обеспечения для сборки виртуальных туров по технологии Владдис Лайт

20.10.2025

Оглавление

1. Введение

ПО предназначено для сборки виртуальных туров по данным, отснятым при помощи системы лазерной координатно-измерительной сканирующей Владдис Лайт (далее – устройство Владдис Лайт). ПО может быть установлено и работает на ПК либо на сервере, под OS Windows 10, Windows 11 либо Ubuntu 24.04.

ПО представляет собой standalone приложение, запускаемое из командной строки. Название файла приложения: *BuildVladdisLightVirtualTour*.

ПО представляет собой консольное приложение и не имеет графического пользовательского интерфейса. В процессе работы ПО пишет лог *stdout*, выводя сообщения о статусе сборки туров, а также об ошибках и предупреждениях, в случае их возникновения в процессе сборки виртуального тура.

ПО работает в полностью автоматическом режиме. Возможность вмешательства пользователя (например, в процесс обработки изображений, генерации карты пола или 3D реконструкции по панорамным фотографиям) не предусмотрена.

ПО вызывается запуском из командной строки. На вход ПО принимает путь к папке с сырыми данными сцены, снятой при помощи устройства Владдис Лайт, а также опции и параметры сборки туров. На выходе ПО выдаёт zip-архив с виртуальным туром, готовым к отображению при помощи плеера виртуальных туров Контрум Плей.

Языки программирования, использованные при написании ПО: C++, Python, Matlab.

Сборка 3D-туров по данным, отснятым при помощи устройства Владдис Лайт, включат следующие этапы:

- вычисления траектории перемещения устройства Владдис Лайт по объекту во время съёмки;
- 3D реконструкция по данным с сенсора глубины (лидар), а также по парам панорамных снимков с предварительно вычисленным взаимным расположением их точек съёмки при помощи нейронной сети (стереореконструкция);
- обработка изображений (повышение контрастности, выставление баланса белого, шумоподавление и иные этапы обработки);
- многоступенчатая обработка облака точек (регуляризация, удаление движущихся объектов, подавление шумов и пр.);
- создание и текстурирование полигональной сетки;
- генерация карты пола;
- итоговая сборка виртуального тура.

2. Назначение и условия применения

2.1. Виды деятельности

ПО для сборки виртуальных туров по технологии Владдис Лайт работает исключительно с данными, полученными при съёмке окружающего пространства устройством Владдис Лайт, при этом устройство должно управляться специализированным программным обеспечением («Программное обеспечение для управления системой лазерной координатно-измерительной сканирующей Владдис Лайт (ОС Windows)»). По окончании съёмки, отснятые материалы (сцена) выгружаются на ПК или на сервер. Съёмка и выгрузка сцены на сервер должны быть полностью завершены до запуска приложения *BuildVladdisLightVirtualTour*.

Результатом работы ПО является zip-архив с виртуальным туром, готовым к отображению при помощи плеера виртуальных туров Контрум Плей.

Виртуальные туры могут быть использованы в следующих областях:

- строительство и девелопмент;
- недвижимость и инфраструктура;
- строительный контроль;
- разработка и проектирование;
- работы по сохранению историко-культурного наследия.

2.2. Программные и аппаратные требования

2.2.1. Аппаратные требования к целевой платформе

Минимальные требования:

- 32 ядра CPU;
- 128 GB RAM;
- SSD объёмом 1TB;
- видеокарта Nvidia класса GeForce RTX 4090 либо выше.

Рекомендуемые требования:

- 64 ядра CPU;
- 256 GB RAM;
- SSD объёмом 2TB;
- видеокарта Nvidia класса GeForce RTX 4090 либо выше.

2.2.2. Требования к базовой ОС

ПО для сборки виртуальных туров по технологии Владдис Лайт работает под ОС Linux Ubuntu 24.04, Windows 10/11.

2.3. Языки программирования, используемые при разработке

Для разработки ПО и модулей использовались следующие языки программирования: C++, Python, Matlab.

3. ПО от сторонних производителей

Для работы ПО для сборки виртуальных туров по технологии Владдис Лайт требуется следующее стороннее ПО:

- нейронная сеть RAFT Stereo (https://github.com/princeton-vl/RAFT-Stereo), предназначенная для вычисления оптического потока (optical flow) по парам панорамных фотоснимков. Нейронная сеть должна быть предварительно обучена на большом количестве (не менее 100 тысяч) пар панорамных фотоснимков, с эталонным, заранее известным значением оптического потока. Возможна замена данной нейросети на близкие аналоги;
- SuperPoint (https://github.com/rpautrat/SuperPoint) – нейронная сеть для мэтчинга панорамных фотоснимков. Возможна замена данной нейросети на близкие аналоги;
- Matlab версии на ниже чем 2024a;
- библиотека g2o https://github.com/RainerKuemmerle/g2o (фреймворк C++ с открытым исходным кодом для оптимизации нелинейных функций ошибок на основе графов);
- библиотека poissonRecon (https://github.com/mkazhdan/PoissonRecon), реализующая алгоритм 3D реконструкции (метод Пуассона);
- библиотека Pymeshlab (https://pymeshlab.readthedocs.io/en/latest/) – содержащая алгоритмы обработки треугольной сетки;
- библиотека nmslib https://github.com/nmslib/nmslib – библиотека, используемая для быстрого поиска ближайших соседей в многомерном пространстве (используется при мэтчинге дескрипторов).

4. Программный состав и функционал системы

В состав ПО для сборки виртуальных туров по данным, отснятым при помощи системы лазерной координатно-измерительной сканирующей Владдис Лайт входят несколько блоков. Ниже приводится описание блоков с описанием функционала каждого из них.

Блок 1. Обработка изображений.

В состав блока входят следующие функции для обработки изображений:

- перепроецирование изображений из эквидистантной проекции в кубическую и обратно;
- повышение контраста и усиление деталей и локальных контрастов на изображениях;
- усиление резкости изображения специальными фильтрами.

Также производится корректировка освещённости изображений путем анализа гистограммы и трансформации освещённости изображений при помощи сплайн интерполяции (Akima spline). При этом сам сплайн строится по результатам анализа гистограммы. После этого усиливается насыщенность цветов, для чего изображения переводятся из пространства RGB в HSV или в HSL), увеличивается насыщенность (второй канал - saturation). Далее происходит обратное преобразование в RGB.

В состав блока также входят функционал для закрашивания (image inpainting) оператора и каркаса устройства Владдис Лайт на изображениях, а также функционал для выравнивания освещённости и баланса белого между смежными панорамами.

Блок 2. Вычисление траектории перемещения устройства Владдис Лайт по объекту во время съёмки.

В данном блоке рассчитываются ключевые точки и дескрипторы для панорамных изображений и облаков точек, полученных с лидара. При этом для вычисления ключевых точек и дескрипторов используются как классические методы (SURF, FPFH), так и методы, основанные на нейросетях (SuperPoint, Xfeat). Далее происходит мэтчинг, то есть сопоставление дескрипторов (при помощи библиотек FLANN, NMSLib, предназначенной для быстрого поиска ближайших соседей в многомерном пространстве) и нахождение наиболее подходящих наборов соответствий (мэтчей) для каждой пары точек съёмки. Все найденные соответствия подаются в алгоритм класса RANSAC, который отфильтровывает неправильно найденные соответствия (аутлаеры), а по оставшимся правильным соответствиям (инлаеры) находит матрицу преобразования (сдвиг + поворот) между системами координат этих точек съёмки.

После попарной привязки точек съёмки происходит анализ всей траектории перемещения устройства по объекту съёмки с целью поиска и замыкания циклов в траектории (loop closures). В случае нахождения циклов происходит расчёт накопленной за цикл ошибки, то есть разности расчётного и фактического положения точек съёмки. Далее решается задача оптимизации, при которой накопленная ошибка «распространяется» по графу позиций (pose graph). Данный метод позволяет улучшить вычисление траектории, минимизировав накопленные за время прохождения траектории ошибки.

Блок 3. 3D реконструкция по парам панорамных фотоснимков (стереореконструкция).

В этом блоке для каждой пары смежных (в контексте графа позиций pose graph) точек съёмки, сферические панорамы разворачиваются вдоль соответствующего этой паре вектора перемещения $\vec{T}$, направленного из центра первой панорамы в центр второй панорамы. Далее при помощи нейронной сети RAFT Stereo или подобных происходит вычисление оптического потока (optical flow) между развёрнутыми панорамами. На основании вычисленного оптического потока между двумя панорамами рассчитывается трёхмерное облако точек методом триангуляции в областях перекрытия зон видимости двух панорам.

Блок 4. Обработка облаков точек.

В блоке обработки облаков точек происходит многоуровневая обработка облаков точек. На первом этапе совмещаются облака точек, полученных с сенсора LIDAR и методом стереореконструкции. Далее облака точек проходят многоуровневую фильтрацию для удаления шумов, движущихся объектов и прочих артефактов, вычисляются нормали для облака точек, производится регуляризация облака точек.

Блок 5. Создание, обработка и текстурирование треугольной сетки

На первом этапе выполняется создание треугольной сетки методом Пуассона, используя обработанное облако точек с предварительно вычисленными нормалями.

Далее выполняется обработка треугольной сетки, её сжатие, устранение различных дефектов сетки, выравнивание границ сетки, интерполяция небольших дыр в сетке и т.п.

Затем производится правильная ориентация треугольников в сетке и их разворот в случае необходимости.

После этого выполняется Ray cast для проверки видимости каждого треугольника из каждой точки съёмки. Это нужно для генерации текстурной карты и последующего текстурирования треугольной сетки.

Далее генерируется текстурная карта с одновременным наложением текстуры на треугольную сетку, результат сохраняется в файл формата *.obj.

В конце генерируется карта пола из *.obj файла.

Блок 6. Финальная сборка виртуального тура

В конце обработки происходит финальная сборка виртуального тура, состоящего из текстурированной 3D-модели в формате *.obj, панорамных фотоснимков в кубической проекции (в нескольких разрешениях) для каждой точки съёмки, карт пола для каждого этажа (в двух разрешениях), а также метаданных. Описание формата виртуального тура хранится в отдельном файле (может быть представлен по запросу).

5. Параметры ПО

ПО может быть установлено и работает на ПК либо на сервере (ОС Windows 10,11 либо Linux Ubuntu 24.04). ПО вызывается запуском из командной строки.

На вход ПО принимает путь к папке с сырыми данными сцены, снятой устройством Владдис Лайт, а также опции и параметры сборки туров. На выходе ПО выдаёт zip архив с виртуальным туром, готовым к воспроизведению посредством плеера туров Контрум Плей.

ПО не имеет графического интерфейса. Пример вызова из командной строки:

Вызов приложения из командной строки происходит следующим образом:

```
BuildVladdisLightVirtualTour.exe rootData
parameterJsonFileRAFT configFileNameExternalApps
OutPutFolder ArchiveFolder numCoresDesired archiveName
parameterJsonFile
```

Параметры приложения:

`rootData` – путь к папке со сценой, выгруженной с устройства Владдис Лайт.

`parameterJsonFileRAFT` – путь к файлу (в формате json) с конфигом нейросети RAFT Stereo.

`configFileNameExternalApps` – путь к файлу с конфигом вспомогательных приложений и утилит.

`OutPutFolder` – путь к папке, куда будут складываться ассеты виртуального тура (панорамы, 3D модель, текстурные карты) в процессе работы ПО.

`ArchiveFolder` – папка, в которую будет записан итоговый zip архив с виртуальным туром.

`numCoresDesired` – максимальное количество воркеров в параллельном пуле (Parallel pool), который используется при параллельных и асинхронных вычислениях.

`archiveName` – имя архива с итоговым виртуальным туром.

`parameterJsonFile` – файл с параметрами сборки виртуального тура.

Пример запуска ПО:

```
BuildVladdisLightVirtualTour.exe
C:\Input_Data\scene_20_10_2025   parameterJsonFileRAFT.json
configFileNameExternalApps.txt          C:\ReadyTours\Assets
C:\ReadyTours 64 myAppartmentTour.zip parameterJsonFile.json
```

Состав файла `parameterJsonFileRAFT.json`:

`RaftFolderName` – путь к папке, в которой находится нейросеть.

`RaftScriptFileName` – скрипт для активации виртуальной среды и запуска нейросети для набора стереопар.

`AnacondaPath` – путь к папке, в которую установлена Anaconda3.

`ResH` – разрешение панорам, для которого происходит вычисление оптического потока.

`cameraHeightInMeters` – опциональный параметр, высота точек съёмки над уровнем пола (или земли).

`SuperPointFolderName` – путь к папке с нейросетью SuperPoint для мэтчинга панорамных фотографий.

`SuperPointExchangeFolder` – путь к папке, куда записываются входные данные для нейросети SuperPoint.

`SuperPointResultFolder` – путь к папке, куда записываются результаты мэтчинга.

Пример файла:

```
{
  "RaftFolderName": "C:\\NNs\\RAFT-Stereo",
  "RaftScriptFileName": "RAFT-Stereo.bat",
  "AnacondaPath": "C:\\Users\\user\\anaconda3",
  "ResH": 1536,
  "cameraHeightInMeters": 1.5,
  "SuperPointFolderName":
"C:\\NNs\\SuperPointTrial_M\\matching-cv-vs-nn\\",
  "SuperPointExchangeFolder":
"C:\\NNs\\SuperPointTrial_M\\matching-cv-vs-nn\\data\\",
  "SuperPointResultFolder":
"C:\\NNs\\SuperPointTrial_M\\matching-cv-vs-nn\\results\\"
}
```

Состав файла `parameterJsonFile.json`:

`pano_resolution` – разрешение итоговых панорам, в готовом виртуальном туре (для кубической проекции, длина грани куба).

`color_processing_mode` – режим цветообработки. Поддерживается 2 режима (1 и 2).

`auto_horizon_adjustment` – автоматическое выравнивание горизонта панорам и 3D модели.

Пример файла:

```
{
  "tour_options": {
    "pano_resolution": 1536,
    "color_processing_mode": 1,
    "auto_horizon_adjustment": false
  }
```

```
}
```

Состав файла `configFileNameExternalApps.txt`:

`OutputCloudsFolder` – папка, в которую записываются временные и промежуточные файлы с облаками точек.

`TempFolder` – папка для хранения временных файлов.

`vladdis_kd_tree_folder` – путь к папке с C++ библиотекой для быстрого построения kd-деревьев, и быстрого поиска ближайших соседей.

`PoissonReconFolder` – Путь к папки с утилитой для 3D реконструкции методом Пуассона.

`MeshlabExchangeFolder` – папка для обмена данными между основным приложением и скриптом, реализующим функционал библиотеки PyMeshlab.

`MeshlabPythonSourseFolder` – путь к папке, где лежат скрипты, реализующие функционал библиотеки PyMeshlab.

`TopViewMapFromObjApp` – путь к приложению для построения карты пола из 3D модели.

`ReconstructionTempFolder` – временная папка для хранения результатов обработки 3D моделей.

`loop_correction_file_name` – путь к приложению, реализующему алгоритм замыкания циклов (loop Closure) на базе библиотеки g2o.

`loopClosureDataExcangeFolder` – путь к папке для обмена данными между основным приложением, и приложением для замыкания циклов.

Пример файла `configFileNameExternalApps.txt`

```
E:\ ProcessingTempArtefacts\OutputCloudsVladdisLight\
E:\Temp\
E:\Postprocessing_Libraries_and_toolboxes\vladdis_kd_tree-
develop\
E:\Postprocessing_Libraries_and_toolboxes\AdaptiveSolvers.x64\
PoissonRecon
E:\ProcessingTempArtefacts\MeshlabExchangeFolder\
E:\Postprocessing_Libraries_and_toolboxes\Meshlab_python_Sours
e\
E:\Postprocessing_Libraries_and_toolboxes\TopViewMapFromObj\bi
n\TopViewMapFromObj.exe
E:\ProcessingTempArtefacts\ReconstructionTempFilesSFM\3D_model
s\
E:\Postprocessing_Libraries_and_toolboxes\loop_closure_binary\
loop_closure-master\bin\win64\loop_correction.exe
E:\Temp\loopClosureDataExcange\
```