Сравнительный анализ эффективности программных инструментов для разбивки видео на кадры на примере области оценки качества дорожной поверхности

А.А. Журавлев, К.А. Аксенов

Уральский федеральный университет

Аннотация: Дороги занимают важное место в жизни почти каждого человека. Качество покрытия является наиболее значимой характеристикой дорожного полотна. Для его оценки существует множество систем, среди которых есть те, которые анализируют дорожную поверхность с помощью потоков видеоинформации. В свою очередь, видео разбивается на кадры, и уже по изображениям происходит непосредственно оценка качества дороги. Разбивка видео на кадры в таких системах происходит на основе специальных программных инструментов. Чтобы понять, насколько эффективно конкретное программное обеспечение, необходим детальный анализ. В данной статье в качестве программных инструментов для анализа выбраны OpenCV, MoviePy и FFMpeg. Материалом исследования является двухминутное видео дорожной поверхности с частотой кадров 29,97 кадров/с и форматом mp4. В качестве показателя эффективности используется среднее время получения одного кадра из видео. Для каждого из трех программных инструментов проведено по 5 различных экспериментов, в которых размер кадра в пикселях последовательно увеличивается в 2 раза: 40000, 80000, 160000, 320000, 640000. Каждая программа обладает линейной зависимостью O(n) среднего времени получения кадра от разрешения, однако, FFMpeg имеет наименьшие абсолютные показатели времени, а также наименьшую скорость роста функции, поэтому является наиболее эффективным инструментом по сравнению с остальными (OpenCV, MoviePy).
Ключевые слова: сравнение, анализ, эффективность, программный инструмент, библиотека, программа, разбивка видео, размер кадра, разрешение, дорожная поверхность.

Введение

Оценка качества дорожного покрытия - одна из самых популярных задач во всем мире. Данной области посвящено большое количество как монографий [1, 2], так и научных работ, направленных на обработку видео и изображений дорожной поверхности различными методами, которые можно разделить на две категории: традиционные (без использования машинного обучения) [3] и с использованием машинного обучения [4, 5]. В некоторых исследованиях проводится анализ эффективности различных алгоритмов [6, 7]. В этой работе акцент делается именно на эффективность. В качестве области для анализа выбрана разбивка видео на кадры различными

программными инструментами (библиотеками): OpenCV [8], MoviePy [9] и FFMpeg [10]. Каждая из перечисленных программ может обладать определенными особенностями такими, как сложность обработки видео конкретного разрешения, что, в свою очередь, может оказать существенное влияние на функцию зависимости времени получения кадра от разрешения. Например, функция может быть квадратической - $O(n^2)$, или кубической - $O(n^3)$. Также программные инструменты в случае, если имеют одинаковую зависимость (например, линейную), могут обладать разной скоростью роста функции, или отличаться абсолютными показателями времени получения кадра из видео. Квадратичный или кубический характер функции, высокая скорость роста и большие абсолютные показатели времени оказывают существенное влияние на производительность. Для того, чтобы понять, какие характеристики имеют конкретные программные инструменты, необходимо провести соответствующий эксперимент.

Цель работы – сравнить эффективность существующих программных инструментов для разбивки видео на кадры от разрешения (размера кадра).

Материал исследования – видео дорожного покрытия.

В статье используется эмпирический метод исследования, поскольку основным источником результата являются сравнение и эксперимент.

Научная новизна заключается в методологии эксперимента для определения эффективности программных инструментов.

Задачи исследования:

1. Описать методологию проведения эксперимента для определения зависимости эффективности программ для разбивки видео на кадры от разрешения.
2. Дать краткую информацию о выбранных программных инструментах.
3. Провести сравнительный анализ библиотек на основе предложенной методологии.

4. Описать полученные результаты.

Описание анализируемых программных инструментов

В качестве программных инструментов для анализа выбраны OpenCV, MoviePy, FFMpeg. Краткая информация о каждой программе (библиотеке) представлена ниже.

OpenCV является бесплатной кроссплатформенной библиотекой для обработки видео и изображений в реальном времени. Данный программный инструмент работает под управлением самых популярных операционных систем, таких как Windows, OS X, GNU/Linux, iOS, Android и пр. Библиотека OpenCV содержит более 2500 алгоритмов, обширную документацию, исходный код и примеры кода для компьютерного зрения в реальном времени [8].

MoviePy – библиотека Python для редактирования видео: обрезка, конкатенация, вставка титров, компоновка (нелинейный монтаж), обработка и создание пользовательских эффектов. Этот инструмент может читать и записывать все наиболее распространенные форматы аудио и видео, включая GIF, и работает на Windows/Max/Linux с Python 2.7+ и 3 [9].

FFMpeg – универсальный программный инструмент, позволяющий считывать самые разные входные данные, включая устройства захвата/записи в реальном времени, фильтровать и кодировать их во множество выходных форматов [10]. Требует предварительной установки на компьютер с операционной системой Windows.

Методология эксперимента

Для эксперимента используется видео дорожной поверхности, которое имеет следующие характеристики: продолжительность - 2 минуты, частота кадра – 29,97 кадров/с, формат – mp4, исходные ширина и высота кадра – 200 (пикселей).

Исходное видео и его характеристики представлены на рисунке 1.

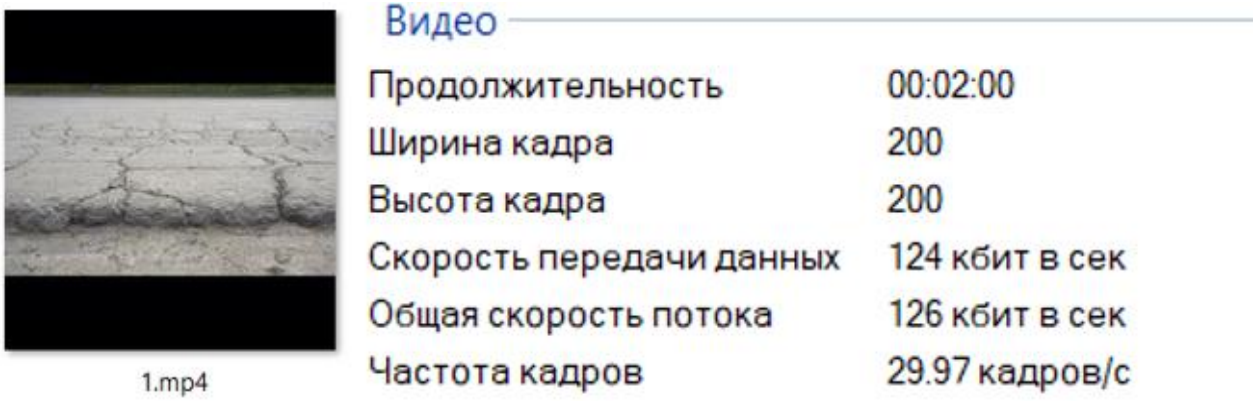

Рис. 1. – Исходное видео и его характеристики

Одним из важных показателей эффективности программного инструмента является время решения поставленной задачи (в данном случае разбивка видео на кадры). Поэтому в этой статье в качестве показателя эффективности будем считать время, которое затрачивается программой на то, чтобы получить один кадр из видео. Также большое влияние на эффективность оказывает размер данных, с которым работает программный инструмент. В нашем случае размером данных является разрешение кадра.

Компьютер, на котором проводится исследование, имеет следующие характеристики: процессор – Intel Core i5, оперативная память – 8 гигабайт, частота процессора – 8 гигагерц, тип системы – x64, операционная система – Windows 10 Pro. В качестве среды выполнения используется Visual Studio Code 2023 язык программирования – Python (версия 3.11.7), тип приложения – Jupyter-блокнот.

Описание эксперимента для определения зависимости среднего времени получения одного кадра из видео от размера (кадра) в пикселях:

1. В качестве исходного материала исследования для каждого из программных средств (OpenCV, MoviePy, FFMpeg) используется видео, информация о котором представлена на рис. 1.

2. Для каждой программы проводится серия из 5 экспериментов, в которых последовательно увеличивается количество пикселей, содержащихся в одном кадре, в 2 раза: 40000 (ширина - 200, высота – 200), 80000 (≈283x283), 160000 (400x400), 320000 (≈566x566), 640000 (800x800). При этом видео для каждого эксперимента хранится в отдельной компьютерной папке (всего 15 различных папок: 3 (программных инструмента) x 5 (экспериментов с разным размером кадра)). Все кадры сохраняются в ту же папку, в которой находится видео.

3. Для определения среднего времени получения одного кадра из видео суммарное время, которое затрачивается на разбивку всего видео на кадры, делится на общее количество кадров (1).

$$t_{\text{ср}} = \frac{t_{\text{общ}}}{n} \quad (1)$$

где $t_{\text{ср}}$ – среднее время получения одного кадра из видео, $t_{\text{общ}}$ – общее время получения всех кадров из видео, n – количество кадров.

Результаты экспериментов

После проведения описанного ранее эксперимента для каждого из видео суммарно получается 3606 кадров (120 с (2 минуты в секундах) x 29,97 кадров/с ≈ 3606 кадров). Значение получается немного большим, поскольку продолжительность видео округляется компьютером. Изначально в папках для экспериментов находятся только видео формата mp4. После проведения эксперимента в папке также находятся кадры, полученные из исходного видео.

Результаты эксперимента для определения эффективности программных инструментов для разбивки видео на кадры на основе предложенной ранее методологии представлены в таблице 1.

Таблица № 1

Средняя время (в миллисекундах) получения одного кадра из видео в зависимости от размера (в пикселях)

Программный инструмент	**Размер кадра**				
	40000	80000	160000	320000	640000
OpenCV	1,6	2,8	5,2	9,7	17,6
MoviePy	1,5	2,1	3,2	5,0	9,1
FFMpeg	0,8	1,0	1,5	2,6	4,3

Для того, чтобы однозначно понять какой характер зависимости имеет конкретный программный инструмент от размера кадра необходимо построить соответствующий график (рис. 2).

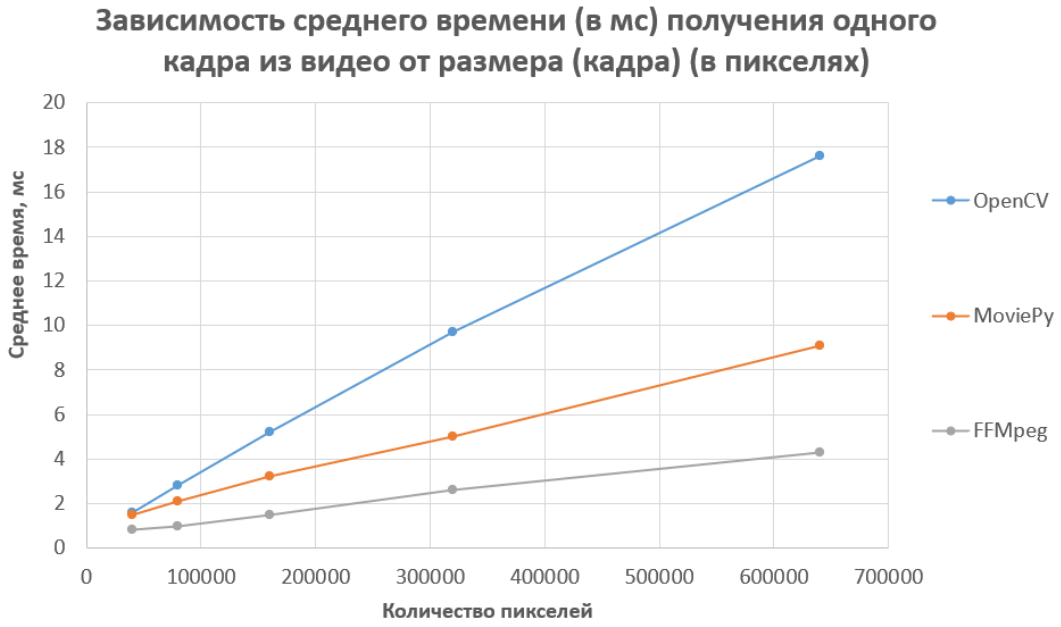

Рис. 2. – График зависимости среднего времени (в мс) получения одного кадра из видео от размера (кадра) в пикселях

Обсуждение результатов

Как видно из графика, каждый из программных инструментов имеет линейную зависимость *O(n)* среднего времени получения одного кадра из видео от размера в пикселях. Однако, абсолютные показатели времени и скорость роста функций отличаются. Чтобы иметь четкую картину

эффективности конкретного программного инструмента в выбранной системе из трех библиотек (OpenCV, MoviePy, FFMpeg), рассчитаем эффективность, зависящую от скорости изменения функции, таким образом, чтобы сумма всех показателей (эффективности) давала 1. Необходимы следующие формулы:

$$v_{\text{ср}} = \frac{\Delta t}{\Delta x} = \frac{t_{max} - t_{min}}{x_{max} - x_{min}} \quad (2)$$

где $v_{\text{ср}}$ – средняя скорость изменения функции, Δt – разница максимального (t_{max}) и минимального (t_{min}) абсолютного значения времени, Δx – разница максимального (x_{max}) и минимального (x_{min}) значения разрешения.

$$k_{\text{норм}} * \left(\sum_{i=1}^{N} \frac{1}{v_{\text{ср}\,i}}\right) = k_{\text{норм}} * \left(\sum_{i=1}^{N} \frac{\Delta x_i}{\Delta t_i}\right) = 1 \quad (3)$$

где $k_{\text{норм}}$ – коэффициент нормализации, $v_{\text{ср}\ i}$ – скорость изменения конкретного программного инструмента, N – количество программных инструментов (в нашем случае 3).

$$E = \frac{k_{\text{норм}}}{v_{\text{ср}}} = \frac{k_{\text{норм}} * \Delta x}{\Delta t} \quad (4)$$

где E – эффективность конкретного программного инструмента.

Подставляя нужные значения в формулы (2), (3) и (4), получаем следующие показатели эффективности для каждой из программ (значения округлены до 3 знаков после запятой): $E_{OpenCV} \approx 0{,}130$; $E_{MoviePy} \approx 0{,}275$; $E_{FFMpeg} \approx 0{,}595$.

FFMpeg имеет существенно больший показатель по сравнению с OpenCV и MoviePy.

Проведенное исследование может быть очень полезно для дальнейших работ, поскольку в ходе эксперимента получено достаточно большое количество изображений, что в значительной степени упрощает сбор данных, а также дает представление об эффективности конкретного программного

инструмента, что поможет в будущем разработать приложение для оценки качества дорожного покрытия с высокой производительностью.

Выводы

В статье проведен сравнительный анализ эффективности программных инструментов для разбивки видео на кадры в зависимости от размера (кадра) в пикселях на примере области оценки качества дорожной поверхности. Материалом исследования является двухминутное видео дорожного покрытия с частотой 29.97 кадров/с и форматом mp4. В качестве программных инструментов для анализа выбраны OpenCV, MoviePy и FFMpeg. Для каждой из перечисленных программ проведена серия из 5 экспериментов с последовательным увеличением размера кадра в 2 раза: 40000, 80000, 160000, 320000, 640000. Показателем эффективности выбрано среднее время получения одного кадра из видео. Каждый из выбранных программных инструментов обладает линейной зависимостью $O(n)$, однако, FFMpeg имеет наименьшие абсолютные показатели времени, а также наименьшую скорость роста функции, поэтому является наиболее эффективным по сравнению с остальными (OpenCV, MoviePy).

Литература

1. Визильтер Ю.В., Желтов С.Ю., Бондаренко А.В., Ососков М.В., Моржин А.В. Обработка и анализ изображений в задачах машинного зрения. - М.: ФИЗМАТКНИГА, 2010. - 672 с.

2. Гонсалес Р. Цифровая обработка изображений. - М.: Техносфера, 2012. - 1101 с.

3. Fan R., Liu M. Road Damage Detection Based on Unsupervised Disparity Map Segmentation // IEEE Transactions on Intelligent Transportation Systems. 2020. Vol. 21, Issue 11, pp. 4906–4911.

4. Danti A., Kulkarni J.Y., Hiremath P.S. An Image Processing Approach to Detect Lanes, Pot Holes and Recognize Road Signs in Indian Roads // International Journal of Modeling and Optimization. 2012. Vol. 2, Issue 6, pp. 658–662.

5. Silva L.A., Leithardt V.R.Q., Batista V.F.L., González G.V., Santana J.F.D.P. Automated Road Damage Detection Using UAV Images and Deep Learning Techniques // IEEE Access. 2023, Vol. 11, pp. 62918–62931.

6. Журавлев А.А. Сравнение эффективности классификации методов выделения контуров на примере изображений дорожного покрытия // XXI век: итоги прошлого и проблемы настоящего плюс. 2023. Т. 12, № 1, С. 23–28.

7. Zhuravlev A.A., Aksyonov K.A. Comparison of Contour Detection Methods in Images on the Example of Photos with Road Surface Damage // Institute of Electrical and Electronics Engineers Inc. Damage // Institute of Electrical and Electronics Engineers Inc., 2023 IEEE Ural-Siberian Conference on Biomedical Engineering, Radioelectronics and Information Technology (USBEREIT). 2023, pp. 183-186.

8. What is OpenCV? The Complete Guide (2024). URL: viso.ai/computer-vision/opencv/

9. MoviePy. URL: pypi.org/project/moviepy/

10. FFMpeg Documentation. URL: ffmpeg.org/ffmpeg.html#:≈:text=ffmpeg%20is%20a%20universal%20media,a%20plethora%20of%20output%20formats.

References

1. Vizil`ter Yu.V., Zheltov S.Yu., Bondarenko A.V., Ososkov M.B., Morzhin A.V. Obrabotka i analiz izobrazhenij v zadachax mashinnogo zreniya [Image processing and analysis in machine vision problems]. M.: FIZMATKNIGA, 2010. 672 p.

2. Gonsales R. Cifrovaya obrabotka izobrazhenij [Digital image processing]. Texnosfera. 2012. 1101 p.

3. Fan R., Liu M. IEEE Transactions on Intelligent Transportation Systems. 2020. Vol. 21, Issue 11, pp. 4906–4911.

4. Danti A., Kulkarni J.Y., Hiremath P.S. International Journal of Modeling and Optimization. 2012. Vol. 2, Issue 6, pp. 658–662.

5. Silva L.A., Leithardt V.R.Q., Batista V.F.L., González G.V., Santana J.F.D.P. IEEE Access. 2023, Vol. 11, pp. 62918–62931.

6. Zhuravlev A.A. XXI vek: itogi proshlogo i problemy` nastoyashhego plyus. 2023. Vol. 12, № 1, pp. 23-28.

7. Zhuravlev A.A., Aksyonov K.A. Institute of Electrical and Electronics Engineers Inc. Damage // Institute of Electrical and Electronics Engineers Inc., 2023 IEEE Ural-Siberian Conference on Biomedical Engineering, Radioelectronics and Information Technology (USBEREIT). 2023, pp. 183-186.

8. What is OpenCV? The Complete Guide (2024). URL: viso.ai/computer-vision/opencv/

9. MoviePy. URL: pypi.org/project/moviepy/

10. FFMpeg Documentation. URL: ffmpeg.org/ffmpeg.html#:≈:text=ffmpeg%20is%20a%20universal%20media,a%20plethora%20of%20output%20formats.

Дата поступления: 4.02.2024

Дата публикации: 13.03.2024