

ГУАП

КАФЕДРА № 42

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ _____
ПРЕПОДАВАТЕЛЬ

старший преподаватель		Т.А. Суетина
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №8

РАБОТА С ДАННЫМИ ВИДЕОФАЙЛА ФОРМАТА MP4

по курсу: Техника аудиовизуальных средств информации

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. №	4329		Д.С. Шаповалова
		подпись, дата	инициалы, фамилия

Санкт-Петербург 2025

1. Цель работы:

получить теоретические знания о структуре видеофайла, типах кадров и принципов их обработки. На практике реализовать полученные знания.

2. Задание:

Для файла в формате mp4 выполнить:

1. Считать параметры заголовка и вывести на экран полученные данные.
2. Создать файл, в который загрузить до 100 последовательных кадров и сохранить их в формате JPG.
3. Вывести время и I-кадры из сохраненного файла.
4. Вывести время и P-кадры из сохраненного файла.
5. Вычислить максимальное значение межкадровой разности сохраненной последовательности кадров.
6. С помощью ffmpeg создать видео с внешним представлением для отображения векторов движения.

Для всех заданий указать используемые функции из библиотеки, описать их назначение, принцип работы.

3. Ход работы:

Чтение заголовка файла

С помощью библиотеки `python-ffmpeg` и Python программы был считан заголовок `mp4` файла, заголовок видеопотока и заголовок аудиопотока (рисунок 1).

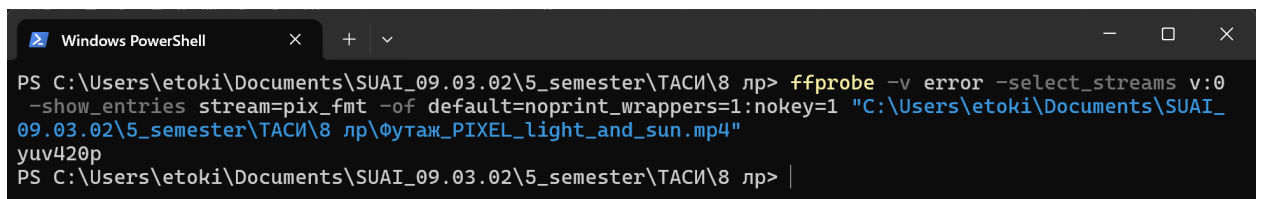
```
=== Основная информация о файле ===  
Формат: mov,mp4,m4a,3gp,3g2,mj2  
Длительность (сек): 6.806800  
Размер файла (байт): 14706199  
Битрейт: 17284126  
  
=== Видео ===  
Кодек: h264  
Разрешение: 1920x1080  
FPS: 30000/1001  
Битрейт: 16947243  
  
=== Аудио ===  
Кодек: aac  
Частота дискретизации: 48000  
Каналы: 2
```

Рисунок 1.1 – Заголовок `mp4` файла

С помощью `ffprobe`, путём ввода команды –

```
ffprobe -v error -select_streams v:0 -show_entries stream=pix_fmt -of  
default=noprint_wrappers=1:nokey=1  
"C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\TACI\8  
лр\Футаж_PIXEL_light_and_sun.mp4"
```

– в консоль, было получено значение битовой глубины (рисунок 1.2)



```
Windows PowerShell  
PS C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\TACI\8 лр> ffprobe -v error -select_streams v:0  
-show_entries stream=pix_fmt -of default=noprint_wrappers=1:nokey=1 "C:\Users\etoki\Documents\SUAI_  
09.03.02\5_semester\TACI\8 лр\Футаж_PIXEL_light_and_sun.mp4"  
yuv420p  
PS C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\TACI\8 лр> |
```

Рисунок 1.2 – Битовая глубина видео

Используемая функция:

- `ffmpeg.probe(video_path)` — из библиотеки `ffmpeg-python`

Назначение:

Получение метаданных видеофайла (формат, потоки, длительность, битрейт и пр.) без декодирования содержимого.

Принцип работы:

- Вызывает утилиту `ffprobe`.

- Возвращает структуру в формате JSON, содержащую:
 - `format` — общая информация о контейнере (MP4 и т.д.);
 - `streams` — список потоков (видео, аудио, субтитры), с деталями: кодек, разрешение, частота кадров, битрейт и др.
- Не требует распаковки видео — работает быстро и эффективно.

Кадры

С помощью Python были извлечены первые 100 кадров и сохранены в формате JPG.

В качестве примера представлены первые 3 кадра на рисунках 2.1-2.3:

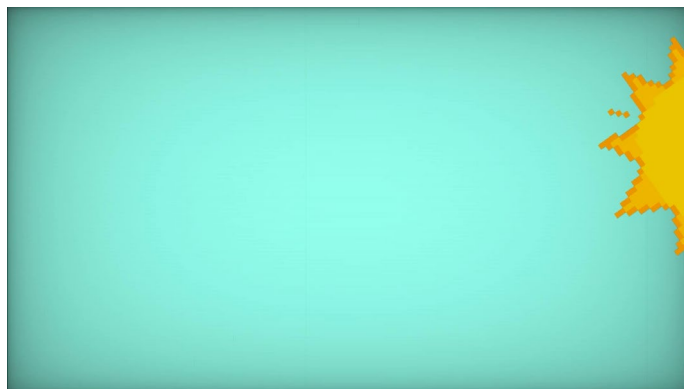


Рисунок 2.1 – Кадр 1

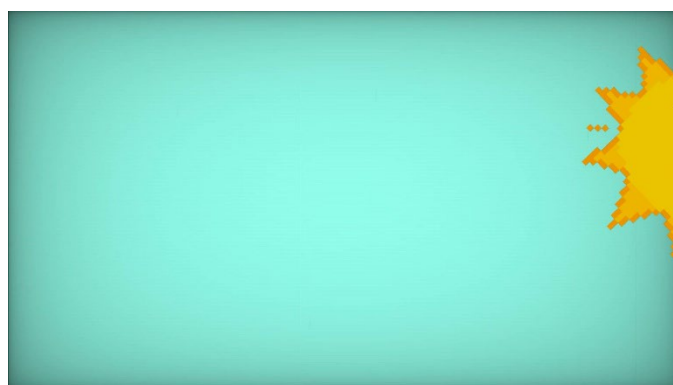


Рисунок 2.2 – Кадр 2

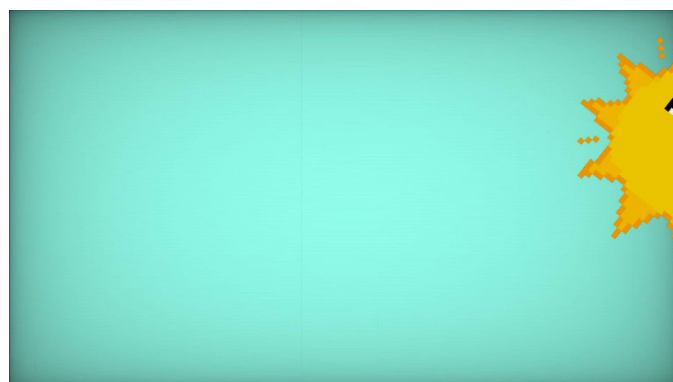


Рисунок 2.3 – Кадр 3

Используемые средства:

- Через ffmpeg-python:
python
1
- Или через утилиту ffmpeg:
bash
1

Назначение:

Извлечение фиксированного числа кадров из видео и сохранение в виде отдельных изображений.

Принцип работы:

- `-vframes N` — указывает, сколько кадров декодировать с начала видео.
- FFmpeg декодирует видео и сохраняет каждый кадр как отдельный файл в формате JPG.
- Имена файлов генерируются по шаблону (например, `frame_001.jpg`, `frame_002.jpg`, ...).

С помощью команды —

```
ffmpeg -i "C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\TACH8
np\Фумахс_PIXEL_light_and_sun.mp4" -vf "select='eq(pict_type|i)'" -vsync vfr
"C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\TACH8 np\video i frame %04d.jpg"
```

– были извлечены из видеопотока все кадры и сохранены только те, у которых параметр *pict type* равен I.

```
PS C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\tACI\8 np> ffmpeg -i "C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\tACI\8 np\Oтpaж_PIXEL_light_and_sun.mp4" -vf "select=eq(pict_type,I)" -vsync vfr "C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\tACI\8 np\video_i_frame_%04d.jpg"
```

```
ffmpeg version 8.0.1-full-build-www.gyan.dev Copyright (c) 2000-2025 the FFmpeg developers
built with gcc 15.2.0 (Rev8, Built by MSYS2 project)
configuration: --enable-gpl --enable-version3 --enable-static --disable-w32threads --disable-autodetect --enable-fontconfig --enable-iconv --enable-gnutls --enable-lcms2 --enable-libxml2 --enable-gmp --enable-bzlib --enable-lzma --enable-libsnappp --enable-zlib --enable-librist --enable-libsrt --enable-libssh --enable-libzmq --enable-avisynth --enable-libbluray --enable-libcaca --enable-libdav1d --enable-libdvdread --enable-sdl2 --enable-libaribb24 --enable-libaribcaption --enable-libdav1d --enable-libdav1d --enable-libdav1d --enable-libopenjpeg --enable-libquirc --enable-libuavs3d --enable-libxevd --enable-libzvb1 --enable-liboapv --enable-libgqrencode --enable-librav1e --enable-libsvtav1 --enable-libsvtenc --enable-libwebp --enable-libx264 --enable-libx265 --enable-libxavs2 --enable-libxvid --enable-libxvid --enable-libaoim --enable-libjxl --enable-libvpx --enable-mediafoundation --enable-libass --enable-frei0r --enable-libfreetype --enable-libfribidi --enable-libharfbuzz --enable-liblensfun --enable-libvidstab --enable-libvmaf --enable-libzimg --enable-amf --enable-cuda-llvm --enable-cuvid --enable-dxva2 --enable-d3d11va --enable-d3d12va --enable-ffnvcodec --enable-libvpl --enable-nvdec --enable-nvenc --enable-vaapi --enable-libshaderc --enable-vulkan --enable-libplacebo --enable-openccl --enable-libcdio --enable-openal --enable-libgme --enable-libmodplug --enable-libopenmpt --enable-libopencore-amrwb --enable-libomp3lame --enable-libshine --enable-libtheora --enable-libtwolame --enable-libvo-amrwbenc --enable-libcodec2 --enable-libilbc --enable-libgsm --enable-libl3 --enable-libopenh264 --enable-libopus --enable-libspeex --enable-libvorbis --enable-ladspa --enable-libbs2b --enable-libbrotli --enable-libbrotli --enable-librubberband --enable-libsoxr --enable-chromaprint --enable-whisper
```

```
libavutil      60. 8.100 / 60. 8.100
libavcodec     62. 11.100 / 62. 11.100
libavformat    62. 3.100 / 62. 3.100
libavdevice    62. 1.100 / 62. 1.100
libavfilter    11. 4.100 / 11. 4.100
libswscale     9. 1.100 / 9. 1.100
libswresample  6. 1.100 / 6. 1.100
```

```
-vsync is deprecated. Use -fps_mode
```

```
Input #0, mov, mp4, m4a, 3gp, 3g2, mj2, from 'C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\tACI\8 np\Oтpaж_PIXEL_light_and_sun.mp4':
Metadata:
  major_brand       : mp42
  minor_version     : 0
  compatible_brands: mp42mp41
  creation_time     : 2025-12-22T10:35:10.000000Z
Duration: 00:00:06.81, start: 0.000000, bitrate: 17284 kb/s
Stream #0:0[0x1](eng): Video: h264 (High) (avc1 / 0x31637661), yuv420p(tv, progressive), 1920x1080 [SAR 1:1 DAR 16:9], 16947 kb/s, 29.97 fps, 29.97 tbr, 30k tbn (default)
```

```
Metadata:
  creation_time     : 2025-12-22T10:35:10.000000Z
  handler_name      : ?Mainconcept Video Media Handler
  vendor_id         : [0][0][0][0]
  encoder           : AVC Coding
Stream #0:1[0x2](eng): Audio: aac (LC) (mp4a / 0x6134706D), 48000 Hz, stereo, fltp, 317 kb/s (default)
```

```
Metadata:
  creation_time     : 2025-12-22T10:35:10.000000Z
  handler_name      : #Mainconcept MP4 Sound Media Handler
  vendor_id         : [0][0][0][0]
```

```
Stream mapping:
  Stream #0:0 -> #0:0 (h264 (native) -> mjpeg (native))
Press [q] to stop, [?] for help
Output #0, image2, to 'C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\tACI\8 np\video_i_frame_%04d.jpg':
Metadata:
  major_brand       : mp42
  minor_version     : 0
  compatible_brands: mp42mp41
  encoder           : Lavf62.3.100
Stream #0:0(eng): Video: mjpeg, yuv420p(pc, progressive), 1920x1080 [SAR 1:1 DAR 16:9], q=2-31, 200 kb/s, 29.97 fps, 29.97 tbn (default)
```

```
Metadata:
  encoder           : Lavc62.11.100 mjpeg
  creation_time     : 2025-12-22T10:35:10.000000Z
  handler_name      : ?Mainconcept VIdeo Media Handler
  vendor_id         : [0][0][0][0]
```

```
Side data:
cpb: bitrate max/min/avg: 0/0/200000 buffer size: 0 vbv_delay: N/A
[out#0/image2 @ 000002a21689c240] video:513KiB audio:0KiB subtitle:0KiB other streams:0KiB global headers:0KiB muxing overhead: unknown
frame= 7 fps=0.0 q=9.9 Lsize=N/A time=00:00:06.03 bitrate=N/A speed=13.1x elapsed=0:00:00.46
PS C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\tACI\8 np>
```

Рисунок 3.1.1 – Выполнение команды

```
PS C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\TACI\8 lp> ffprobe -v quiet -select_streams v:0 -show_frames -show_entries frame=pict_type,best_effort_timestamp_time -of csv "C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\TACI\8 lp\футаж_PIXEL_light_and_sun.mp4" | findstr ",I"
frame,0.000000,I
frame,1.001000,I
frame,2.002000,I
frame,3.003000,I
frame,4.004000,I
frame,5.005000,I
frame,6.006000,I
```

Рисунок 3.1.2 – Время появления I кадров, вывод в консоль

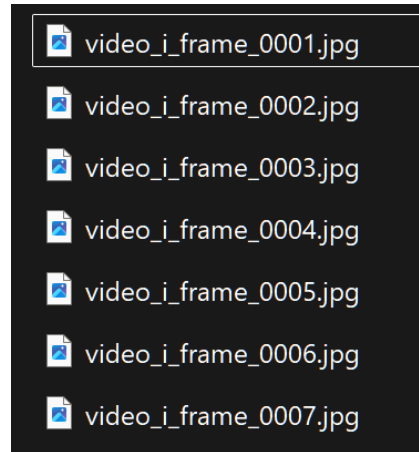


Рисунок 3.2 – Полученные I-кадры в проводнике

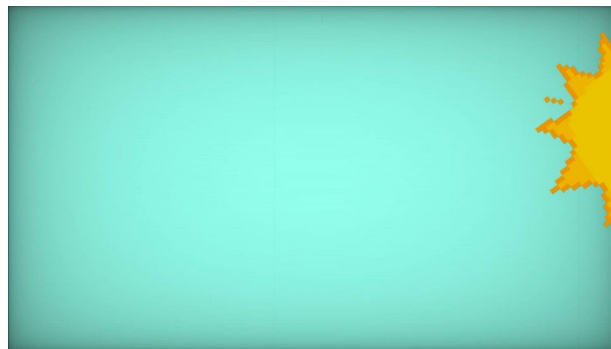


Рисунок 3.3.1 – Первый I-кадр

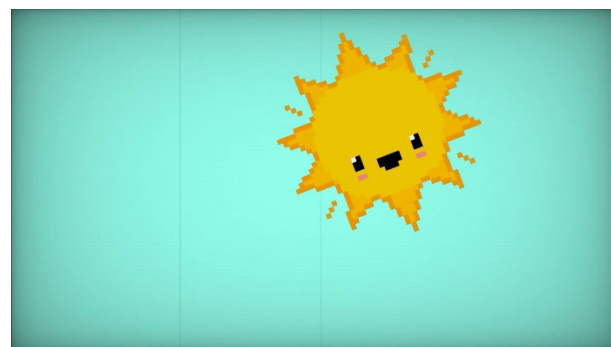


Рисунок 3.3.1 – Второй I-кадр

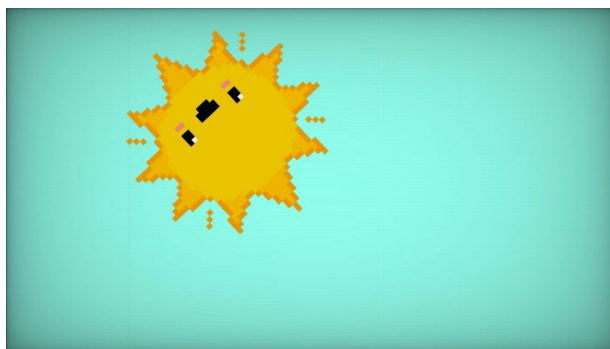


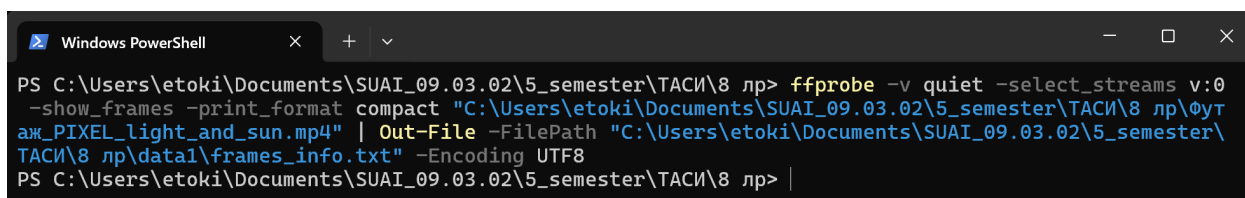
Рисунок 3.3.1 – Третий I-кадр

С помощью команды –

```
ffprobe -v quiet -select_streams v:0 -show_frames -print_format compact  
"C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\ТАСИ\8  
лр\Футаж_PIXEL_light_and_sun.mp4" | Out-File -FilePath  
"C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\ТАСИ\8 лр\data1\frames_info.txt" -  
Encoding UTF8
```

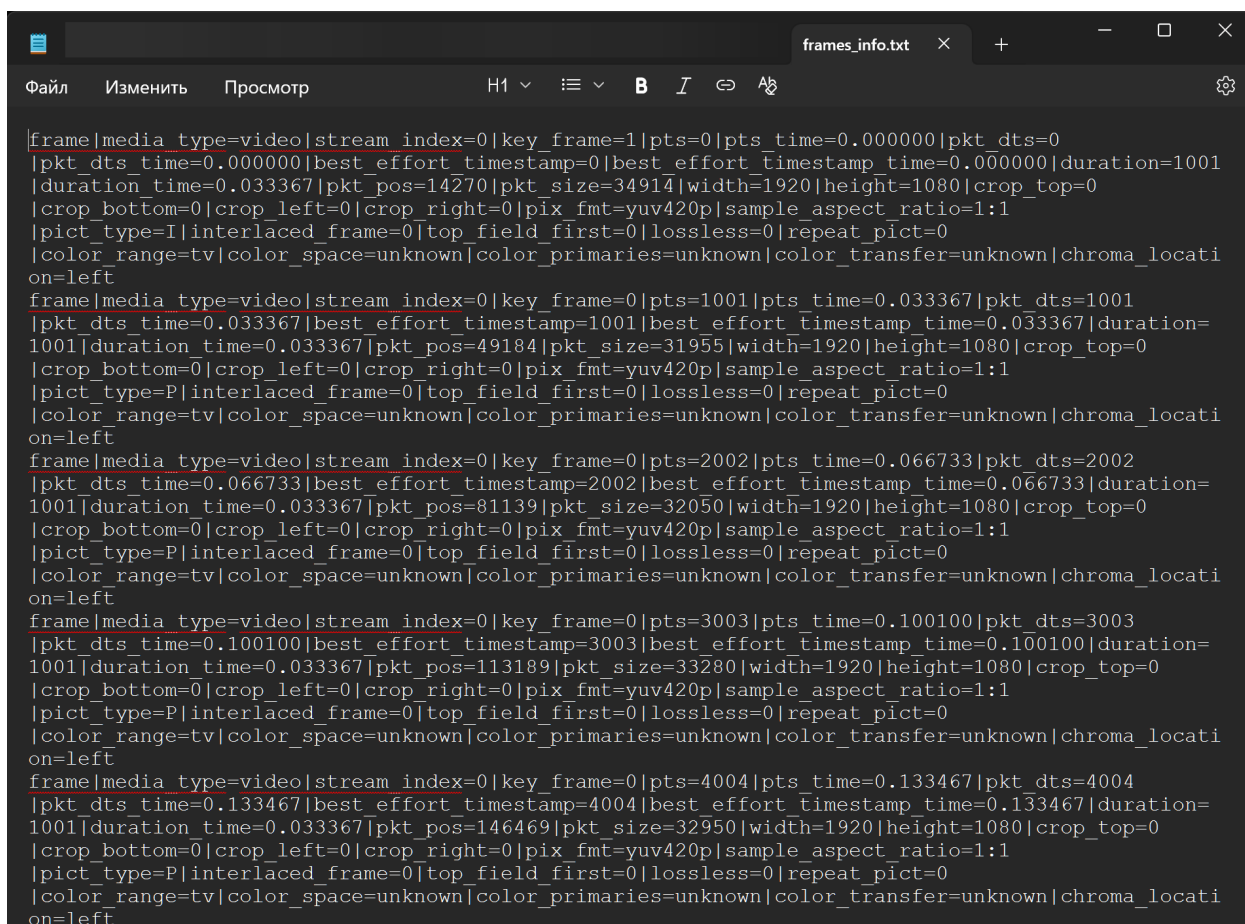
– были извлечены данные о временных метках всех кадров.

Далее, с помощью Python были отобраны I-кадры и сохранены их временные метки.



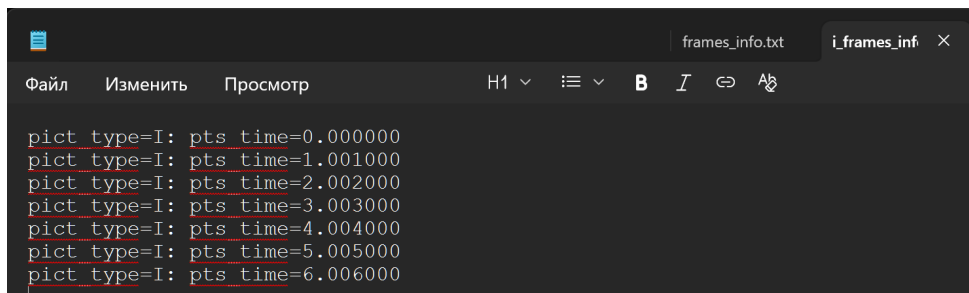
```
Windows PowerShell
PS C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\ТАСИ\8 лр> ffprobe -v quiet -select_streams v:0  
-show_frames -print_format compact "C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\ТАСИ\8 лр\Фут  
аж_PIXEL_light_and_sun.mp4" | Out-File -FilePath "C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\  
ТАСИ\8 лр\data1\frames_info.txt" -Encoding UTF8  
PS C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\ТАСИ\8 лр> |
```

Рисунок 4.1 – Выполненная команда



```
frames_info.txt
Файл  Изменить  Просмотр  H1  B  I  A
[frame|media_type=video|stream_index=0|key_frame=1|pts=0|pts_time=0.000000|pkt_dts=0  
|pkt_dts_time=0.000000|best_effort_timestamp=0|best_effort_timestamp_time=0.000000|duration=1001  
|duration_time=0.033367|pkt_pos=14270|pkt_size=34914|width=1920|height=1080|crop_top=0  
|crop_bottom=0|crop_left=0|crop_right=0|pix_fmt=yuv420p|sample_aspect_ratio=1:1  
|pict_type=I|interlaced_frame=0|top_field_first=0|lossless=0|repeat_pict=0  
|color_range=tv|color_space=unknown|color_primaries=unknown|color_transfer=unknown|chroma_locati  
on=left  
[frame|media_type=video|stream_index=0|key_frame=0|pts=1001|pts_time=0.033367|pkt_dts=1001  
|pkt_dts_time=0.033367|best_effort_timestamp=1001|best_effort_timestamp_time=0.033367|duration=  
1001|duration_time=0.033367|pkt_pos=49184|pkt_size=31955|width=1920|height=1080|crop_top=0  
|crop_bottom=0|crop_left=0|crop_right=0|pix_fmt=yuv420p|sample_aspect_ratio=1:1  
|pict_type=P|interlaced_frame=0|top_field_first=0|lossless=0|repeat_pict=0  
|color_range=tv|color_space=unknown|color_primaries=unknown|color_transfer=unknown|chroma_locati  
on=left  
[frame|media_type=video|stream_index=0|key_frame=0|pts=2002|pts_time=0.066733|pkt_dts=2002  
|pkt_dts_time=0.066733|best_effort_timestamp=2002|best_effort_timestamp_time=0.066733|duration=  
1001|duration_time=0.033367|pkt_pos=81139|pkt_size=32050|width=1920|height=1080|crop_top=0  
|crop_bottom=0|crop_left=0|crop_right=0|pix_fmt=yuv420p|sample_aspect_ratio=1:1  
|pict_type=P|interlaced_frame=0|top_field_first=0|lossless=0|repeat_pict=0  
|color_range=tv|color_space=unknown|color_primaries=unknown|color_transfer=unknown|chroma_locati  
on=left  
[frame|media_type=video|stream_index=0|key_frame=0|pts=3003|pts_time=0.100100|pkt_dts=3003  
|pkt_dts_time=0.100100|best_effort_timestamp=3003|best_effort_timestamp_time=0.100100|duration=  
1001|duration_time=0.033367|pkt_pos=113189|pkt_size=33280|width=1920|height=1080|crop_top=0  
|crop_bottom=0|crop_left=0|crop_right=0|pix_fmt=yuv420p|sample_aspect_ratio=1:1  
|pict_type=P|interlaced_frame=0|top_field_first=0|lossless=0|repeat_pict=0  
|color_range=tv|color_space=unknown|color_primaries=unknown|color_transfer=unknown|chroma_locati  
on=left  
[frame|media_type=video|stream_index=0|key_frame=0|pts=4004|pts_time=0.133467|pkt_dts=4004  
|pkt_dts_time=0.133467|best_effort_timestamp=4004|best_effort_timestamp_time=0.133467|duration=  
1001|duration_time=0.033367|pkt_pos=146469|pkt_size=32950|width=1920|height=1080|crop_top=0  
|crop_bottom=0|crop_left=0|crop_right=0|pix_fmt=yuv420p|sample_aspect_ratio=1:1  
|pict_type=P|interlaced_frame=0|top_field_first=0|lossless=0|repeat_pict=0  
|color_range=tv|color_space=unknown|color_primaries=unknown|color_transfer=unknown|chroma_locati  
on=left
```

Рисунок 4.2 – Временные метки всех кадров (первые 5 кадров)

A screenshot of a text editor window with a dark theme. The window has two tabs: 'frames_info.txt' and 'i_frames_inf'. The 'frames_info.txt' tab is active. The menu bar includes 'Файл', 'Изменить', and 'Просмотр'. The toolbar shows 'H1', a list icon, 'B', 'I', '↔', and 'Aa'. The text content consists of seven lines, each representing an I-frame: 'pict_type=I: pts time=0.000000', 'pict_type=I: pts time=1.001000', 'pict_type=I: pts time=2.002000', 'pict_type=I: pts time=3.003000', 'pict_type=I: pts time=4.004000', 'pict_type=I: pts time=5.005000', and 'pict_type=I: pts time=6.006000'. The words 'pict_type' and 'pts time' are underlined in red in the original image.

```
pict_type=I: pts time=0.000000
pict_type=I: pts time=1.001000
pict_type=I: pts time=2.002000
pict_type=I: pts time=3.003000
pict_type=I: pts time=4.004000
pict_type=I: pts time=5.005000
pict_type=I: pts time=6.006000
```

Рисунок 4.3 – Временные метки I-кадров

Используемые средства:

- Утилита `ffprobe`.
- Обработка вывода в Python (через `subprocess.run()` или `ffmpeg.probe()` с JSON).

Назначение:

Получение временных меток и типов всех кадров в видеопотоке.

Принцип работы:

- `ffprobe` проходит по каждому закодированному кадру и извлекает:
 - `pict_type` — тип кадра (I, P, B);
 - `best_effort_timestamp_time` — время в секундах.
- В Python фильтруются строки с `pict_type=I`, и из них сохраняются временные метки.


```
Windows PowerShell
PS C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\TACI\8 np> ffprobe -v quiet -select_streams v:0 -show_frames -show_entries frame=pict_type,best_effort_timestamp_time -of csv "C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\TACI\8 лр\утаж_PIXEL_light_and_sun.mp4" | findstr ",P"
frame,0.033367,P
frame,0.066733,P
frame,0.100100,P
frame,0.133467,P
frame,0.166833,P
frame,0.200200,P
frame,0.233567,P
frame,0.266933,P
frame,0.300300,P
frame,0.333667,P
frame,0.367033,P
frame,0.400400,P
frame,0.433767,P
frame,0.467133,P
frame,0.500500,P
frame,0.533867,P
frame,0.567233,P
frame,0.600600,P
frame,0.633967,P
frame,0.667333,P
frame,0.700700,P
frame,0.734067,P
frame,0.767433,P
frame,0.800800,P
frame,0.834167,P
frame,0.867533,P
frame,0.900900,P
frame,0.934267,P
frame,0.967633,P
frame,1.034367,P
frame,1.067733,P
frame,1.101100,P
frame,1.134467,P
frame,1.167833,P
frame,1.201200,P
frame,1.234567,P
frame,1.267933,P
```

Рисунок 5.1.2 - Время появления Р-кадров, вывод в консоль

Общее число Р-кадров – 197, частично представлены на рисунках 5.2 – 5.3.3:



Рисунок 5.2 – Полученные Р-кадры в проводнике

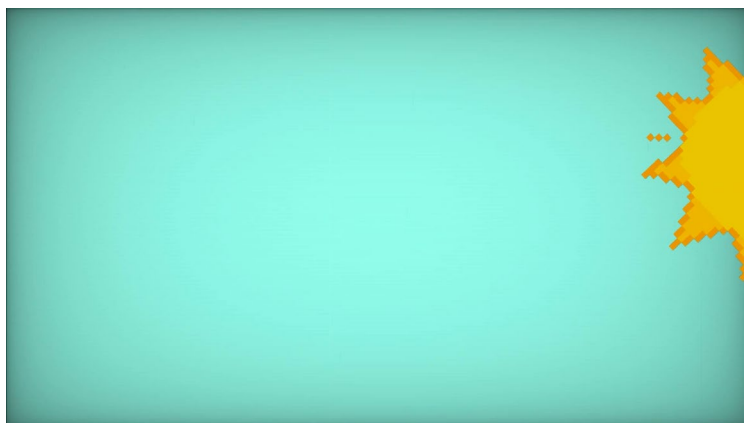


Рисунок 5.3.1 – Первый Р-кадр

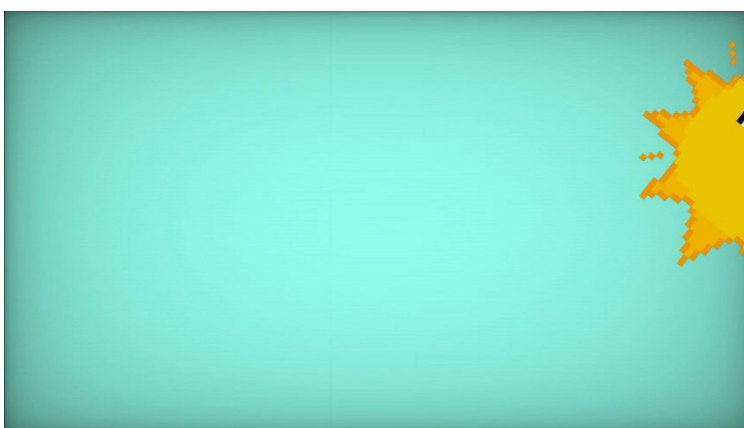


Рисунок 5.3.2 – Второй Р-кадр

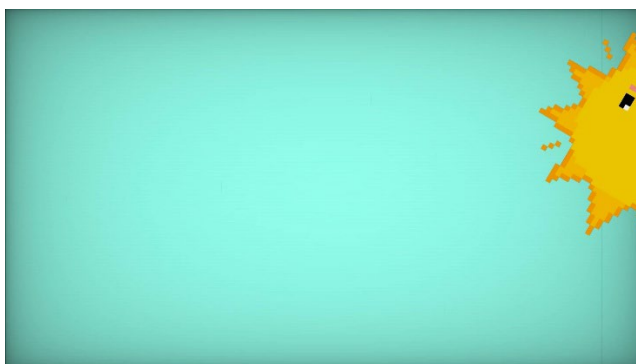
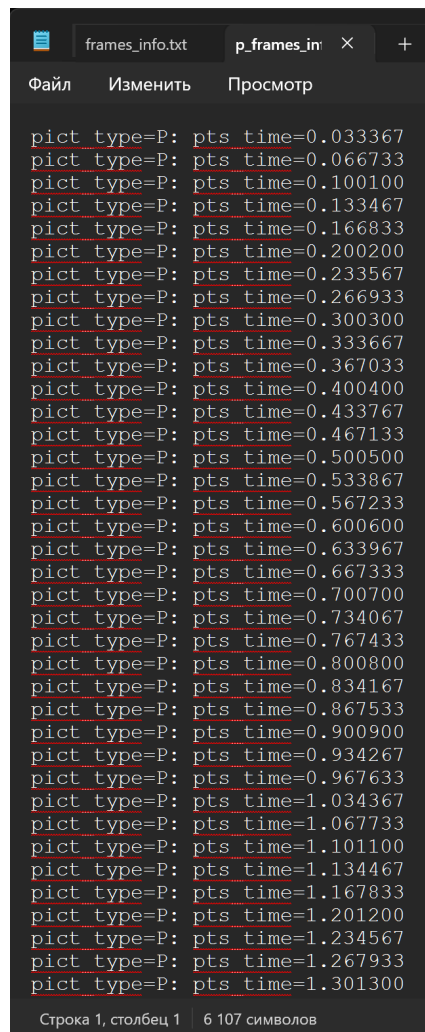


Рисунок 5.3.3 – Третий Р-кадр

Из ранее извлечённых данных о временных метках всех кадров были отобраны с помощью Python Р-кадры и сохранены их временные метки отдельно.



```
frames_info.txt  p_frames_in  X  +
Файл  Изменить  Просмотр

pict type=P: pts time=0.033367
pict type=P: pts time=0.066733
pict type=P: pts time=0.100100
pict type=P: pts time=0.133467
pict type=P: pts time=0.166833
pict type=P: pts time=0.200200
pict type=P: pts time=0.233567
pict type=P: pts time=0.266933
pict type=P: pts time=0.300300
pict type=P: pts time=0.333667
pict type=P: pts time=0.367033
pict type=P: pts time=0.400400
pict type=P: pts time=0.433767
pict type=P: pts time=0.467133
pict type=P: pts time=0.500500
pict type=P: pts time=0.533867
pict type=P: pts time=0.567233
pict type=P: pts time=0.600600
pict type=P: pts time=0.633967
pict type=P: pts time=0.667333
pict type=P: pts time=0.700700
pict type=P: pts time=0.734067
pict type=P: pts time=0.767433
pict type=P: pts time=0.800800
pict type=P: pts time=0.834167
pict type=P: pts time=0.867533
pict type=P: pts time=0.900900
pict type=P: pts time=0.934267
pict type=P: pts time=0.967633
pict type=P: pts time=1.034367
pict type=P: pts time=1.067733
pict type=P: pts time=1.101100
pict type=P: pts time=1.134467
pict type=P: pts time=1.167833
pict type=P: pts time=1.201200
pict type=P: pts time=1.234567
pict type=P: pts time=1.267933
pict type=P: pts time=1.301300

Строка 1, столбец 1  6 107 символов
```

Рисунок 6 – Временные метки Р-кадров (первые 38 кадров)

Используемые средства:

- Те же, что и в п. 3: ffprobe + фильтрация по pict_type=P.

Назначение:

Аналогично I-кадрам, но для Р-кадров.

Принцип работы:

- В том же выводе ffprobe ищутся строки с pict_type=P.
- Сохраняются соответствующие временные метки.

Максимальное значение межкадровой разности

С помощью Python и библиотек `opencv`, `numpy`, `os` были найдены и сохранены 2 кадра с максимальным значением межкадровой разности, а также представлена их межкадровая разность.

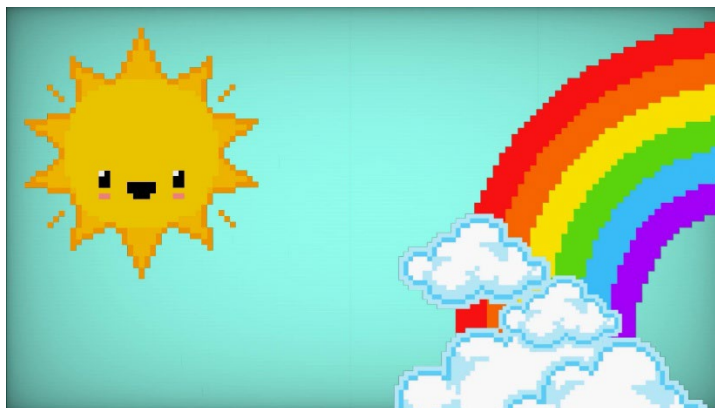


Рисунок 7.1 – 166 кадр

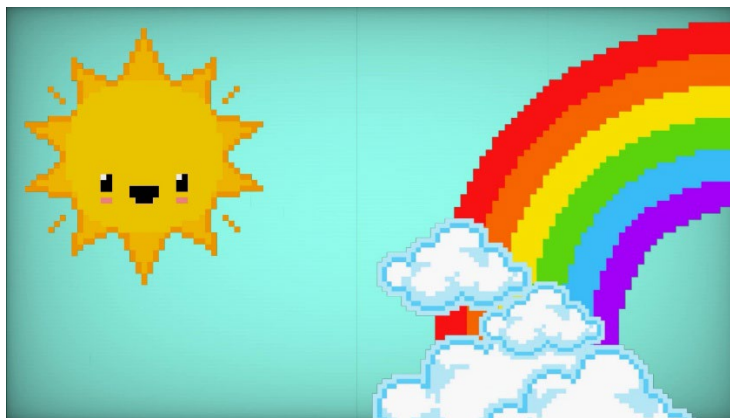


Рисунок 7.2 – 167 кадр

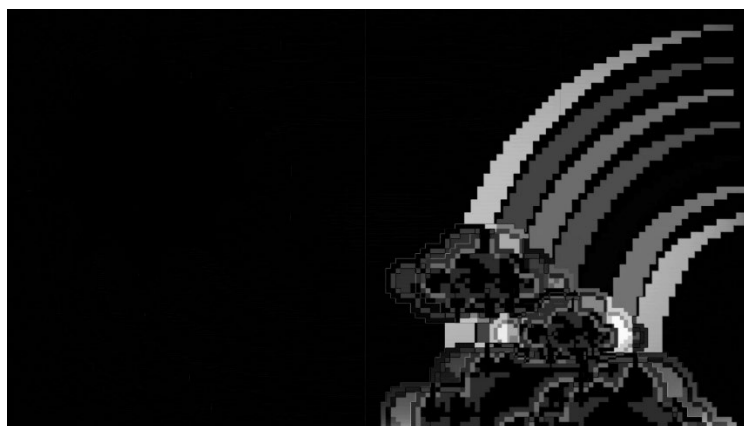


Рисунок 7.3 – Межкадровая разность

Межкадровая разность в 11,9 (условных единиц яркости).

Используемые функции из `OpenCV (cv2)`:

- `cv2.VideoCapture(video_path)` — чтение видео по кадрам.
- `cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)` — преобразование в оттенки серого.

- `cv2.absdiff(img1, img2)` — поэлементная абсолютная разность двух изображений.
- `np.mean(diff)` или `np.sum(diff)` — усреднение или суммирование разности для получения скалярной меры.

Назначение:

Оценка степени изменения сцены между соседними кадрами.

Принцип работы:

- Видео читается кадр за кадром.
- Каждый кадр преобразуется в градации серого.
- Для каждой пары соседних кадров вычисляется матрица разности пикселей.
- Из этой матрицы вычисляется скаляр — среднее/сумма значений.
- Отслеживается максимальное значение и сохраняются соответствующие кадры.

Опционально: визуализация разности через `cv2.normalize()` и `cv2.imwrite()`.

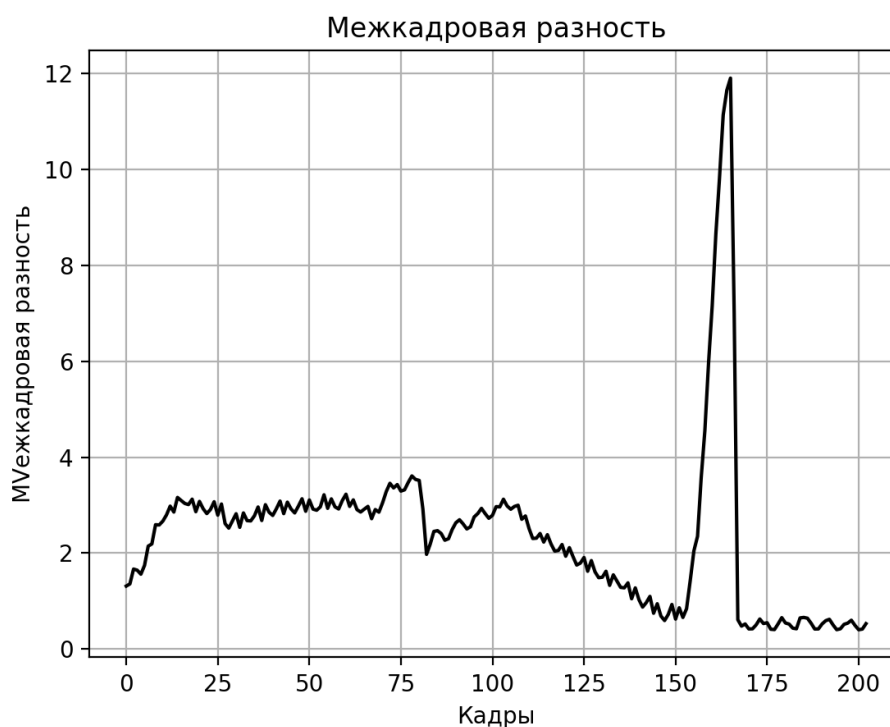


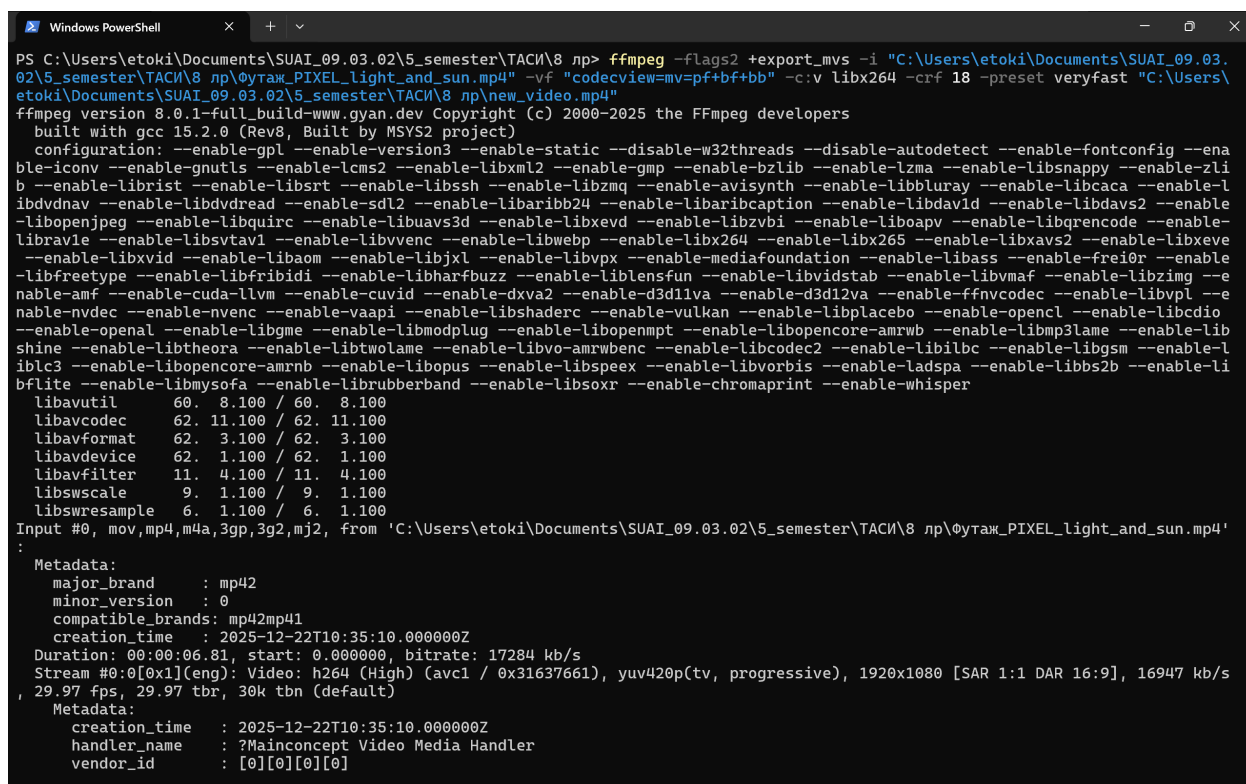
Рисунок 7.4 – График изменения межкадровой разности

Векторы движения

С помощью команды –

```
ffmpeg -flags2 +export_mvs -i "C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\TACИ\8  
лр\Футаж_PIXEL_light_and_sun.mp4" -vf "codecview=mv=pf+bf+bb" -c:v libx264 -crf 18 -  
preset veryfast "C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\TACИ\8  
лр\new_video.mp4"
```

– было сделано видео, демонстрирующее перемещение элементов в видео, векторы движения.



```
PS C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\TACИ\8 лр> ffmpeg -flags2 +export_mvs -i "C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\TACИ\8 лр\Футаж_PIXEL_light_and_sun.mp4" -vf "codecview=mv=pf+bf+bb" -c:v libx264 -crf 18 -preset veryfast "C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\TACИ\8 лр\new_video.mp4"
ffmpeg version 8.0.1-full_build-www.gyan.dev Copyright (c) 2000-2025 the FFmpeg developers
  built with gcc 15.2.0 (Rev8, Built by MSYS2 project)
  configuration: --enable-gpl --enable-version3 --enable-static --disable-w32threads --disable-autodetect --enable-fontconfig --enable-iconv --enable-gnutls --enable-lcms2 --enable-libxml2 --enable-gmp --enable-bzlib --enable-lzma --enable-libsnappp --enable-zlib --enable-librist --enable-libsrt --enable-libssh --enable-libzmq --enable-avisynth --enable-libbluray --enable-libcaca --enable-lbavformat --enable-lbavdevice --enable-lbavfilter --enable-lbavresample --enable-lbavutil --enable-lbavcodec --enable-lbavformat --enable-lbavdevice --enable-lbavfilter --enable-lbavresample --enable-lbavutil --enable-lbavcodec --enable-lbavformat --enable-lbavdevice --enable-lbavfilter --enable-lbavresample --enable-lbavutil --enable-lbavcodec
  Input #0, mov,mp4,m4a,3gp,3g2,mj2, from 'C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\TACИ\8 лр\Футаж_PIXEL_light_and_sun.mp4':
    Metadata:
      major_brand      : mp42
      minor_version    : 0
      compatible_brands: mp42mp41
      creation_time    : 2025-12-22T10:35:10.000000Z
    Duration: 00:00:06.81, start: 0.000000, bitrate: 17284 kb/s
    Stream #0:0[0x1](eng): Video: h264 (High) (avc1 / 0x31637661), yuv420p(tv, progressive), 1920x1080 [SAR 1:1 DAR 16:9], 16947 kb/s, 29.97 fps, 29.97 tbr, 30k tbn (default)
    Metadata:
      creation_time    : 2025-12-22T10:35:10.000000Z
      handler_name     : ?Mainconcept Video Media Handler
      vendor_id        : [0][0][0][0]
```

Рисунок 8.1.1 – Выполнение команды

```

encoder      : AVC Coding
Stream #0:1[0x2](eng): Audio: aac (LC) (mp4a / 0x6134706D), 48000 Hz, stereo, fltp, 317 kb/s (default)
Metadata:
  creation_time   : 2025-12-22T10:35:10.000000Z
  handler_name    : #Mainconcept MP4 Sound Media Handler
  vendor_id      : [0][0][0][0]
Stream mapping:
  Stream #0:0 -> #0:0 (h264 (native) -> h264 (libx264))
  Stream #0:1 -> #0:1 (aac (native) -> aac (native))
Press [q] to stop, [?] for help
[libx264 @ 0000020c7b4121c0] using SAR=1/1
[libx264 @ 0000020c7b4121c0] using cpu capabilities: MMX2 SSE2Fast SSSE3 SSE4.2 AVX FMA3 BMI2 AVX2
[libx264 @ 0000020c7b4121c0] profile High, level 4.0, 4:2:0, 8-bit
[libx264 @ 0000020c7b4121c0] 264 - core 165 r3223 0480cb0 - H.264/MPEG-4 AVC codec - Copyleft 2003-2025 - http://www.videolan.org/x264.html - options: cabac=1 ref=1 deblock=1:0:0 analyse=0x3:0x113 me=hex subme=2 psy=1 psy_rd=1.00:0.00 mixed_ref=0 me_range=16 chr oma_me=1 trellis=0 8x8dct=1 cqm=0 deadzone=21,11 fast_pskip=1 chroma_qp_offset=0 threads=24 lookahead_threads=8 sliced_threads=0 nr =0 decimate=1 interlaced=0 bluray_compat=0 constrained_intra=0 bframes=3 b_pyramid=2 b_adapt=1 b_bias=0 direct=1 weightb=1 open_gop =0 weightp=1 keyint=250 keyint_min=25 scenecut=40 intra_refresh=0 rc_lookahead=10 rc=crf mbtree=1 crf=18.0 qcomp=0.60 qpmin=0 qpmax =69 qstep=4 ip_ratio=1.40 aq=1:1.00
Output #0, mp4, to 'C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\TACI\8_lp\new_video.mp4':
Metadata:
  major_brand     : mp42
  minor_version   : 0
  compatible_brands: mp42mp41
  encoder         : Lavf62.3.100
Stream #0:0(eng): Video: h264 (avc1 / 0x31637661), yuv420p(tv, progressive), 1920x1080 [SAR 1:1 DAR 16:9], q=2-31, 29.97 fps, 30k tbn (default)
Metadata:
  encoder         : Lavc62.11.100 libx264
  creation_time   : 2025-12-22T10:35:10.000000Z
  handler_name    : ?Mainconcept Video Media Handler
  vendor_id      : [0][0][0][0]
Side data:
  cpb: bitrate max/min/avg: 0/0/0 buffer size: 0 vbv_delay: N/A
Stream #0:1(eng): Audio: aac (LC) (mp4a / 0x6134706D), 48000 Hz, stereo, fltp, 128 kb/s (default)
Metadata:
  encoder         : Lavc62.11.100 aac
  creation_time   : 2025-12-22T10:35:10.000000Z
  handler_name    : #Mainconcept MP4 Sound Media Handler

```

Рисунок 8.1.2 – Выполнение команды

```

vendor_id      : [0][0][0][0]
[Out#0/mp4 @ 0000020c7ab60180] video:35957KiB audio:116KiB subtitle:0KiB other streams:0KiB global headers:0KiB muxing overhead: 0.023658%
frame= 204 fps= 55 q=-1.0 Lsize= 36081KiB time=00:00:06.74 bitrate=43854.0kb/s speed= 1.8x elapsed=0:00:03.74
[libx264 @ 0000020c7b4121c0] frame I:17 Avg QP:21.15 size:260814
[libx264 @ 0000020c7b4121c0] frame P:68 Avg QP:22.89 size:174340
[libx264 @ 0000020c7b4121c0] frame B:119 Avg QP:25.02 size:172524
[libx264 @ 0000020c7b4121c0] consecutive B-frames: 20.1% 3.9% 7.4% 68.6%
[libx264 @ 0000020c7b4121c0] mb I I16..4: 38.0% 13.7% 48.3%
[libx264 @ 0000020c7b4121c0] mb P I16..4: 24.5% 15.2% 14.6% P16..4: 13.2% 10.6% 11.5% 0.0% 0.0% skip:10.4%
[libx264 @ 0000020c7b4121c0] mb B I16..4: 10.8% 6.8% 5.9% B16..8: 12.9% 17.0% 16.6% direct:10.2% skip:19.7% L0:46.3% L1:42.2%
% BI:11.5%
[libx264 @ 0000020c7b4121c0] 8x8 transform intra:25.3% inter:0.6%
[libx264 @ 0000020c7b4121c0] coded y,uvDC,uvAC intra: 57.8% 16.4% 5.6% inter: 39.2% 6.4% 3.6%
[libx264 @ 0000020c7b4121c0] i16 v,h,dc,p: 36% 43% 11% 10%
[libx264 @ 0000020c7b4121c0] i8 v,h,dc,ddl,ddr,vr,hd,vl,hu: 12% 71% 16% 0% 0% 0% 0% 0%
[libx264 @ 0000020c7b4121c0] i4 v,h,dc,ddl,ddr,vr,hd,vl,hu: 28% 32% 24% 3% 2% 2% 3% 4%
[libx264 @ 0000020c7b4121c0] i8c dc,h,v,p: 65% 21% 12% 2%
[libx264 @ 0000020c7b4121c0] Weighted P-Frames: Y:0.0% UV:0.0%
[libx264 @ 0000020c7b4121c0] kb/s:43273.51
[aac @ 0000020c7ab6a40] Qavg: 984.992
PS C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\TACI\8_lp>

```

Рисунок 8.1.3 – Выполнение команды

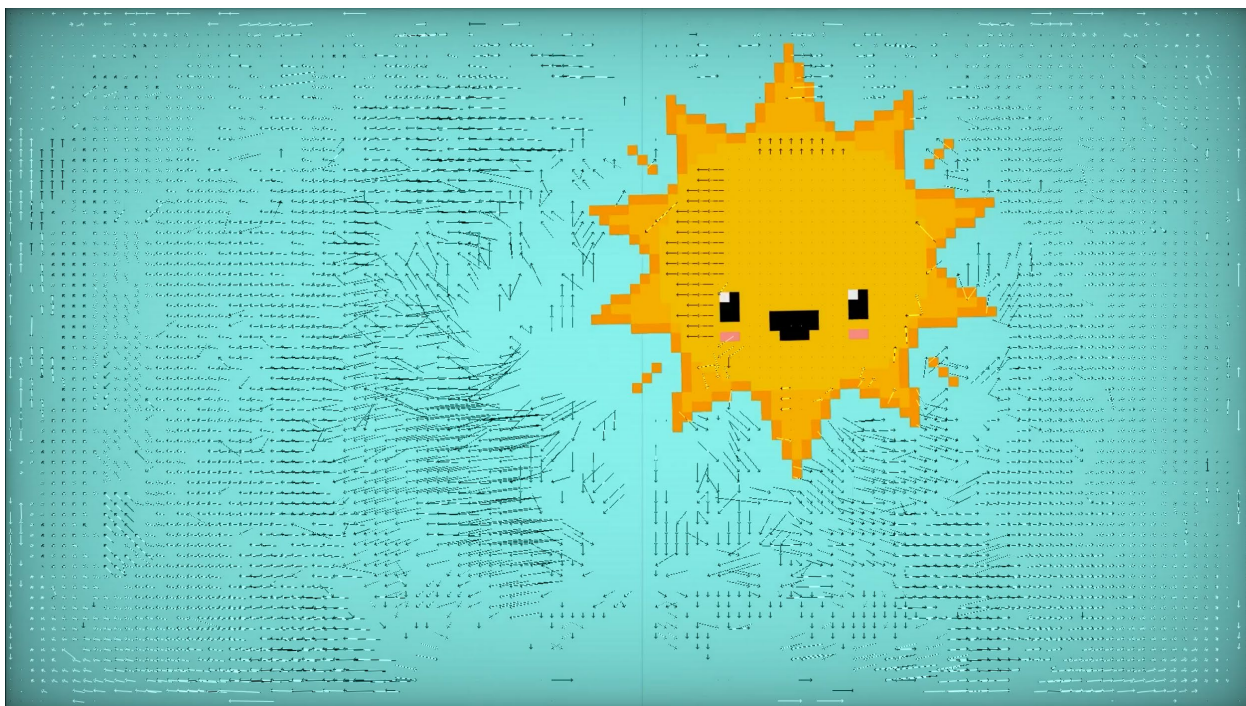


Рисунок 8.2 – Кадр из полученного видео

Назначение используемой команды:

Визуализация векторов движения, используемых кодеком при межкадровом сжатии.

Принцип работы:

- `-flags2 +export_mv` — заставляет декодер экспортировать векторы движения (motion vectors).
- Фильтр `codecview=mv=pf+bf+bb` отображает:
 - `pf` — векторы из Р-кадров (предсказание от предыдущего кадра),
 - `bf` — векторы из В-кадров (вперёд),
 - `bb` — векторы из В-кадров (назад).
- На выходе — видео, где поверх кадров нарисованы стрелки, показывающие направление и величину движения участков сцены.

Команда требует, чтобы исходное видео было закодировано с motion vectors (например, H.264/H.265).

ВЫВОД

В ходе выполнения лабораторной работы были получены теоретические и практические знания о структуре современного видеофайла, принципах межкадрового сжатия и типах кадров (I-, P- и B-кадры). Было продемонстрировано, что видеопоток строится на чередовании опорных (I) и предсказанных (P, B) кадров, что позволяет значительно сократить объём данных без существенной потери визуального качества.

Практическая часть работы позволила:

- извлечь и проанализировать метаданные видеофайла (формат, длительность, битрейт, разрешение, частота кадров);
- сохранить первые 100 кадров в формате JPG для визуального анализа;
- выделить I- и P-кадры как с помощью утилиты `ffmpeg`, так и путём программной обработки данных, полученных через `ffprobe`;
- рассчитать межкадровую разность и определить кадры с максимальным изменением содержимого, что подтверждает наличие динамических сцен в видео;
- визуализировать векторы движения в отдельном видео с использованием фильтра `codecsview`, что наглядно демонстрирует направление и интенсивность перемещения объектов между кадрами.

Все задачи реализованы с применением библиотек `ffmpeg-python`, `OpenCV`, `NumPy` и `matplotlib`, что подтверждает их эффективность для анализа и обработки видеоданных. Работа показала, как теоретические аспекты видеокодеков (например, предсказание движения, опорные кадры) находят своё отражение в реальных инструментах обработки видео.

Таким образом, цель лабораторной работы — освоить структуру видеофайла и методы его анализа на основе типов кадров и межкадровых характеристик — была успешно достигнута.

ПРИЛОЖЕНИЕ А
Листинг Программы
Считывание заголовка:

```
import ffmpeg

def get_mp4_info(path: str) -> None:
    try:
        probe = ffmpeg.probe(path)

        # Достаём данные видео-потока
        video_streams = [s for s in probe["streams"] if s["codec_type"] ==
"video"]
        audio_streams = [s for s in probe["streams"] if s["codec_type"] ==
"audio"]

        print("=== Основная информация о файле ===")
        print("Формат:", probe["format"]["format_name"])
        print("Длительность (сек):", probe["format"]["duration"])
        print("Размер файла (байт):", probe["format"]["size"])
        print("Битрейт:", probe["format"]["bit_rate"])

        if video_streams:
            v = video_streams[0]
            print("\n=== Видео ===")
            print("Кодек:", v.get("codec_name"))
            print("Разрешение:", f"{v.get('width')}x{v.get('height')}")
            print("FPS:", v.get("avg_frame_rate"))
            print("Битрейт:", v.get("bit_rate"))

        if audio_streams:
            a = audio_streams[0]
            print("\n=== Аудио ===")
            print("Кодек:", a.get("codec_name"))
            print("Частота дискретизации:", a.get("sample_rate"))
            print("Каналы:", a.get("channels"))

    except ffmpeg.Error as e:
        print("Ошибка FFmpeg:", e)

# Пример вызова
video_path = r"C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\ТАСИ\8
лр\Футаж_PIXEL_light_and_sun.mp4"
print()
get_mp4_info(video_path)
```

Сохранение первых 100 кадров видео:

```
def extract_100_frames(video_path: str, output_dir: str):
    os.makedirs(output_dir, exist_ok=True)
    output_pattern = os.path.join(output_dir, "frame_%03d.jpg")
    try:
        (
            ffmpeg
            .input(video_path)
            .output(output_pattern, vframes=100, loglevel="quiet")
            .overwrite_output()
            .run(capture_stdout=True, capture_stderr=True)
        )
        print(f"Сохранено до 100 кадров в: {output_dir}")
    except ffmpeg.Error as e:
        print("Ошибка при извлечении кадров:", e.stderr.decode())
        raise
```

Нахождение временных меток I-кадров и Р-кадров:

```
def write_frames_info(all_frames_path: str, i_frames_info_path: str,
p_frames_info_path: str) -> None:
    with (
        open(all_frames_path, "r", encoding="utf") as all_frames_file,
        open(i_frames_info_path, "w", encoding="utf8") as i_file,
        open(p_frames_info_path, "w", encoding="utf8") as p_file,
    ):
        all_frames_file.read(1)

        # 5 - Временная метка в секундах
        # 22 - тип кадра

        for line in all_frames_file:
            line_info = line.split("|")
            match line_info[22]:
                case "pict_type=I":
                    print(f"{line_info[22]}: {line_info[5]}", file=i_file)
                case "pict_type=P":
                    print(f"{line_info[22]}: {line_info[5]}", file=p_file)

if __name__ == "__main__":
    all_frames_path =
r"C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\TACИ\8
mp\data1\frames_info.txt"
    i_frames_info_path =
r"C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\TACИ\8
mp\data1\i_frames_info.txt"
    p_frames_info_path =
r"C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\TACИ\8
mp\data1\p_frames_info.txt"

    write_frames_info(all_frames_path, i_frames_info_path,
p_frames_info_path)
```


Нахождение максимальной межкадровой разницы:

```
import cv2
import numpy as np
import os

def extract_max_diff_frames(video_path, output_dir="output_frames"):
    os.makedirs(output_dir, exist_ok=True)

    cap = cv2.VideoCapture(video_path)
    if not cap.isOpened():
        raise IOError(f"Не удалось открыть видео: {video_path}")

    prev_frame_gray = None
    max_diff_value = 0
    max_diff_frames = (None, None)
    frame_index = 0
    best_index = -1

    while True:
        ret, frame = cap.read()
        if not ret:
            break

        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

        if prev_frame_gray is not None:
            diff = cv2.absdiff(gray, prev_frame_gray)
            diff_value = np.sum(diff)

            if diff_value > max_diff_value:
                max_diff_value = diff_value
                max_diff_frames = (prev_frame.copy(), frame.copy())
                best_index = frame_index

        prev_frame_gray = gray
        prev_frame = frame
        frame_index += 1

    cap.release()

    if max_diff_frames[0] is None:
        raise RuntimeError("Недостаточно кадров для анализа")

    frame1_path = os.path.join(output_dir, "frame_max_diff_1.jpg")
    frame2_path = os.path.join(output_dir, "frame_max_diff_2.jpg")

    cv2.imwrite(frame1_path, max_diff_frames[0])
    cv2.imwrite(frame2_path, max_diff_frames[1])

    diff_image = cv2.absdiff(
        cv2.cvtColor(max_diff_frames[0], cv2.COLOR_BGR2GRAY),
        cv2.cvtColor(max_diff_frames[1], cv2.COLOR_BGR2GRAY)
    )

    # Нормализация для лучшей визуализации
    diff_image_norm = cv2.normalize(diff_image, None, 0, 255,
        cv2.NORM_MINMAX)

    diff_path = os.path.join(output_dir, "frame_difference.jpg")
    cv2.imwrite(diff_path, diff_image_norm)
```

```

return {
    "frame1": frame1_path,
    "frame2": frame2_path,
    "difference_image": diff_path,
    "difference_value": int(max_diff_value),
    "second_frame_index": best_index,
}

if __name__ == "__main__":
    video_file = r"C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\ТАСИ\8
лр\Футаж PIXEL light and sun.mp4" # путь к mp4 файлу
    result = extract_max_diff_frames(video_file)
    print("Сохранены кадры с максимальной разностью:")
    for k, v in result.items():
        print(f"{k}: {v}")

```

Визуализация межкадровой разницы:

```
import cv2

def write_frame_diff(video_path: str, output_path: str) -> None:
    cap = cv2.VideoCapture(video_path)

    if not cap.isOpened():
        raise Exception("Не удалось открыть видео: " + video_path)

    ret, prev_frame = cap.read()
    if not ret:
        raise Exception("Не удалось прочитать первый кадр в видео.")

    # Переводим в grayscale
    prev_gray = cv2.cvtColor(prev_frame, cv2.COLOR_BGR2GRAY)

    diff_values = []

    while True:
        ret, frame = cap.read()
        if not ret:
            break

        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

        # абсолютная разница между кадрами
        diff = cv2.absdiff(gray, prev_gray)

        # средняя разница кадра (одно число)
        mean_diff = diff.mean()

        diff_values.append(mean_diff)

        prev_gray = gray

    cap.release()

    # запись результата
    with open(output_path, "w", encoding="utf-8") as f:
        for value in diff_values:
            f.write(f"{value}\n")

    print(f"Готово! Межкадровая разница записана в:\n{output_path}")

def show_graph():
    import matplotlib.pyplot as plt

    with open(
        r"C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\ТАСИ\8
        лп\data1\frame_diff.txt", "r", encoding="utf8"
    ) as txt_file:
        data = list(map(float, txt_file))
        t = range(len(data))

        plt.plot(t, data, color="black")
        plt.title("Межкадровая разность")
        plt.xlabel("Кадры")
        plt.ylabel("MМежкадровая разность")
        plt.grid()
        plt.show()
```

```

if __name__ == "__main__":
    video_path = r"C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\ТАСИ\8
лп\Фытаж_PIXEL_light_and_sun.mp4"
    output_path = r"C:\Users\etoki\Documents\SUAI_09.03.02\5_semester\ТАСИ\8
лп\data1\frame_diff.txt"

    # print()
    # with open(output_path, "r", encoding="utf8") as txt_file:
    #     print(f"Максимальное значение межкадровой разности:
    {max(map(float, txt_file))}")

    show_graph()

```