Лабораторная работа 2. Потоки. Фильтры. Кодеки.

Отправить свои ответы вы можете на сайте laborant.auditory.ru. У вас будет только одна попытка для отправки ответов.

Пожалуйста, не набирайте свои ответы в Ворде, Гуглодоках и прочих “умных” текстовых редакторах: они заменяют кавычки и дефисы на другие, после чего ни командная строка, ни скрипт не могут распарсить ваши команды.

ВНИМАНИЕ

Обратите особое внимание на мелочи именования файлов: основная часть вашей оценки будет выставлена не за команду, а за результат, который получается при её выполнении. Информация по именованиями файлов указана в каждом задании.

1. Фильтры: слои и потоки

a. **Потоки в Ffmpeg.**

Продолжаем знакомство с Ffmpeg. Напомним: используя несколько ключей -i, вы можете подать на вход ffmpeg несколько файлов (видео, изображения, аудио). Файлы будут нумероваться с нуля, и в выводе ffmpeg о входных файлах вы сможете увидеть слова Stream #0, Stream #1 … Каждый файл он раскладывает на потоки, если это возможно. Например, в видео со звуком будет содержаться не менее двух потоков (дорожек): видеодорожка и аудиодорожка (аудиорожек может быть несколько, кроме того, там могут содержаться субтитры и т.д.). Потоки тоже нумеруются с нуля, через двоеточие: Stream #0:0, Stream #0:1…

Вот что написано про это в документации ffmpeg:

By default, ffmpeg includes only one stream of each type (video, audio, subtitle) present in the input files and adds them to each output file. It picks the "best" of each based upon the following criteria: for video, it is the stream with the highest resolution, for audio, it is the stream with the most channels, for subtitles, it is the first subtitle stream. In the case where several streams of the same type rate equally, the stream with the lowest index is chosen.

Вы можете указывать конкретные потоки вручную. Это можно сделать несколькими способами: указать номер потока и/или тип потока через двоеточие после номера входного файла. По номеру потока ffmpeg выберет конкретный поток, например (0:1 соответствует потоку, который им описан как второй (нумерация с нуля!) поток в первом файле. Чаще всего это аудио). По типу потока можно выбрать v - все видеопотоки, a - все аудиопотоки.

При использовании фильтров такие знания оказываются как нельзя кстати: например, фильтру надо объяснить, какой поток на какой накладывать.

b. **Фильтры.**

Ffmpeg предоставляет возможность использовать различные фильтры при работе с видео. Ознакомиться со всеми вы можете на официальном сайте Ffmpeg: ffmpeg.org. В этой работе мы рассмотрим один из вариантов: `-filter_complex`. Этот ключ сообщает ffmpeg, что сейчас вы будете использовать один (или несколько) из встроенных фильтров. Нас будут интересовать два: наложение (видео на видео, изображения на видео, видео на изображение и проч.) и создание текста. Но сначала поговорим об общих принципах работы видеофильтров.

Текст фильтра строится таким образом:

`[in_link_1]...[in_link_N]filter_name=arguments[out_link_1]...[out_link_M], где`

- [in_link_n] — ссылка на входной поток фильтра;
- [out_link_m]— ссылка на выходной поток фильтра;
- filter_name — имя используемого фильтра;
- arguments — аргументы фильтра.

Ссылка на выходной поток — обычно заданное пользователем имя потока, указанное в квадратных скобках. По умолчанию n-ый выходной поток направляется в n-ый выходной файл.

Ссылки на входной поток — это указанные в квадратных скобках номера файлов (в таком случае ffmpeg выберет поток так, как указано в п. a) с указанным при необходимости номером потока или его буквенным обозначением (v или a). Кроме того, это может быть пользовательское имя потока, которое было задано как выходное имя для одного из предыдущих фильтров.

Аргументы фильтра — обычно пары key=value, определяющие параметры фильтра. (Напр.: координаты наложения), такие пары внутри фильтра разделяются двоеточием.

В значении ключа filter_complex может быть указан не один фильтр, а цепочка фильтров. В таких случаях фильтры разделяются запятыми, по умолчанию выходной поток фильтра подается на вход следующему за ним фильтру.

Текст цепочки фильтров (значение ключа filter_complex) для удобства можно поместить в кавычки, тогда возможно будет использовать пробелы в тексте фильтров для лучшей читаемости.

c. **Фильтр ‘overlay’.**

Фильтр overlay позволяет производить наложение видеопотоков друг на друга. Основными его аргументами являются координаты наложения. Например, фильтр

```
-filter_complex overlay=x=30:y=50
```

наложит второй видеофайл на первый в координатах (30;50). Указывать ключи (x= и y=) в данном случае не обязательно, позволяется оставить только значения координат через двоеточие. Если координаты не будут указаны, фильтр наложит файл в левый верхний угол, соответствующий координатам (0;0).

Для того, чтобы изменить порядок наложения потоков, достаточно указать ссылки на потоки перед именем фильтра: файл из второй ссылки будет наложен на файл из первой ссылки. Фильтр

```
-filter_complex [1][0]overlay=0:700
```

наложит первый входной файл на второй в координатах (0;700).

d. **Фильтр ‘drawtext’.**

Фильтр drawtext позволяет наложить текст на видео.

Обязательными к указанию являются два атрибута: fontfile и text. В fontfile задается путь (относительный или абсолютный) к файлу со шрифтом, в ключе text - текст, который должен быть наложен. Чтобы задать текст с пробелами, необходимо значение ключа взять в кавычки (двойные или одинарные). Все спецсимволы, включая запятые и двоеточия, должны быть экранированы обратным слэшом.

Кроме того, можно указать координаты текста (x и y), кегль ключом fontsize (по умолчанию 16), цвет шрифта ключом fontcolor (по умолчанию черный).

Фильтр

```
-filter_complex “drawtext = x=50: y=560: fontfile=arial.ttf: textcolor=green: text=’This is a great text!’”
```

наложит текст “This is a great text”, написанный 16 кеглем шрифта Arial зеленого цвета на видео в координатах (50;560).

e. **Применение фильтров на заданном промежутке времени**
Иногда требуется применить фильтр к видео не на протяжении всей его длительности. Например, наложить изображение или текст только в определенном промежутке времени. Для таких случаев у фильтров имеется атрибут `enable`, позволяющий задавать время действия фильтра. Его значение - выражение, определяющие, когда фильтр должен быть применен. Выражение `gte(t, 10)` говорит о том, что фильтр будет применяться ко всем кадрам, начиная с секунды 10 (gte = greater or equal). Параметр t говорит о том, что далее будет указано время в секундах. Фильтр
`-filter_complex overlay=enable='between(t,15,4*60)'`
выполнит наложение в левый верхний угол в течение четырёх минут начиная с секунды 15.

f. **Создание файла с несколькими выходным потоками и получение потоков из файла.**
Ключ `-map` позволяет указывать потоки, которые будут направлены в выходной файл. Потоки указываются номером файла (тогда в выходной файл дополнительно будут включены все дорожки этого файла), номером потока или его буквенным обозначением. Также значением ключа `map` может быть пользовательское имя потока, заданное на выходе из фильтра. (Примеры: `-map 0, -map 0:v, -map [outputFromFilter]`)
Все параметры перекодирования, указанные после ключа, будут применяться к указанному в ключе потоку.

g. **Задание 1. Видеографическое оформление записи.**
Вам предложено видео. Это фрагмент из промо-диска МИЭМ. Необходимо оформить этот ролик. Наложите на него логотип и плашку и подпишите выступающую девушку её именем и фамилией (Екатерина Титова), *используя только одну команду*.
Параметры наложения: логотип и плашка подобраны по размерам так, что при наложении в левый верхний угол сразу займут свои места в кадре. Текст должен быть расположен в координатах (10; 640), написан кеглем 24 чёрного цвета, шрифтом arialbd.ttf (считается, что он уже лежит в той папке, в которой вы работаете). Плашка должна появиться на секунде 2, текст на секунде 3, оба должны исчезнуть на секунде 10. Логотип должен быть виден в течение всего видео.
Имена файлов: входной файл input1.avi, файлы с логотипом и плашкой -- logo.png и plate.png соответственно. Выходной файл output1.avi.
Обратите внимание на наличие неперекодированной звуковой дорожки в выходном файле.

h. **Задание 2. Работа с аудиодорожками.**

Часть 1.

Возьмём то же видео, что использовали в первом задании, и дополните его двумя новыми звуковыми дорожками. Если новые звуковые дорожки оказались длиннее вашего видео, то обрежьте их точно по длине имеющегося файла. Видео и звуковые дорожки должны остаться неперекодированными.
Имена файлов: входной файл input1.avi, две входные аудиодорожки audio1.mp3 и audio2.aac. Выходной файл output2.avi.

Часть 2.

Возьмите файл, получившийся в первой части задания, и извлеките из него исходную аудиодорожку. Она должна быть извлечена без перекодирования.
Имена файлов: входным файлом берётся выходной файл из первой части этого задания. Имя выходного файла -- output2.ext, где ext -- расширение, которое необходимо выбрать, исходя из параметров потока.

2. Установки аудио- и видеокодеков.

a. Для понимания данного раздела вам следует ознакомиться с рекомендованной книгой B. Waggoner “Compression for Great Video and Audio.

b. **Настройки кодека H.264.**
Установки кодека h.264 позволяют настроить профиль и уровень, с которым будет кодироваться видео. Профиль устанавливается ключом `-profile:v` далее соответственно одно из трех значений: `baseline`, `main`, `high`. Для установки уровня используется ключ `-level:v`. По умолчанию он принимает значение 4.0.
Для установки CABAC ключу `-coder` нужно задать значение 1, для отключения — значение 0. Использование CABAC недоступно для профиля Baseline.

c. **Настройки кодека MPEG-4.**
Для кодека MPEG-4 Part 2 существует два наиболее часто используемых на практике профиля: Simple и Advanced Simple, которые задаются в ffmpeg значениями 16 и 15 для соответствующего ключа.

d. **Общие настройки видеокодеков.**
Длина GOP в ffmpeg задается ключом `-g`, за которым следует целое число, равное желаемой длине. Количество B-кадров задается ключом `-bf`.
Цветовая субдискретизация относится к параметрам пикселя и выставляется ключом `-pix_fmt`. Есть несколько наиболее используемых значений этого параметра: `yuv420p`, `yuv422p`, `yuv400p`, `rgb8`, `rgb4`, `bgr8`. Здесь первые три буквы задают цветовое пространство, далее, для цветового пространства YUV следующие четыре цифры — параметры цветовой субдискретизации. Для RGB и BGR — количество памяти, выделяемой на канал цвета. Буква p для форматов yuv говорит о том, что такие форматы используются для записи прогрессивного видео (не интерлейс).

e. **Работа с видеобитрейтом.**
В лабораторной работе №1 уже упоминались ключи `-minrate` и `-maxrate`, позволяющие управлять битрейтом видео. Рассмотрим их более подробно. Эти ключи позволяют задать верхнюю и нижнюю границы битрейта при кодировании с переменным битрейтом. Однако, они не будут работать без еще одного дополнительного ключа `-bufsize`. Этот ключ задает размер буфера в битах, после заполнения которого ffmpeg будет сверять, насколько корректно проходит кодирование с точки зрения получающегося битрейта. Задание большого значения приведет к более быстрому кодированию, однако к менее точным результатам, задание маленького значения приведет к долгому кодированию и вырождению качества видео. Таким образом, лучший размер буфера подбирается экспериментально.
Для задания кодирования с постоянным битрейтом следует установить все три ключа `-b:v`, `-minrate` и `-maxrate` равными желаемому значению и указать размер буфера.

f. **Дискретизация аудио** задается ключом `-ar`, далее — значение в Гц.

g. **Задание 3. Кодирование для устройств и сервисов.**
Возьмите файл, который получился в первой части предыдущего задания. Затем закодируйте его для одного устройства и соответствующего сервиса, согласно варианту из этой [варианты будут чуть позже] таблички.
Имена файлов: входным файлом берётся выходной файл из первой части задания 2. Выходные файлы называются output3_1.ext и output3_2.ext соответственно для двух команд. ext заменяется расширением, которое указано в задании.