МЕТОДЫ ПРЕОБРАЗОВАНИЯ ГРАФИЧЕСКОЙ ИНФОРМАЦИИ В АУДИОФАЙЛЫ

Алефиренко Виктор Михайлович,
Белорусский государственный университет информатики и радиоэлектроники,
г. Минск, Республика Беларусь

E-mail: alefirenko@bsuir.by

Асиненко Алексей Михайлович,
Белорусский государственный университет информатики и радиоэлектроники,
г. Минск, Республика Беларусь

E-mail: asinenko2016@mail.ru

Аннотация. Рассмотрены современные методы преобразования графической информации в аудиофайлы. Представлен анализ существующих подходов, включая методы, основанные на преобразовании пиксельных значений в частотные характеристики аудиосигнала, а также подходы, использующие различные виды кодирования (амплитудное, частотное, фазовое) для представления информации о цвете, яркости и координатах пикселей.

Ключевые слова: графическая информация, преобразование изображений, аудиофайлы, защита информации, скрытие информации.

Прямое преобразование изображения, являющегося одним из видов графической информации, в «осмысленный» звук, который легко интерпретировать как информацию из изображения, очень сложно. Чаще всего результат будет скорее абстрактным звуковым представлением, чем прямым переводом.

Перечислим 10 подходов к преобразованию изображений в аудио:

1. Растеризация и высота тона/длительность. Этот подход использует значение каждого пикселя (например, яркость в градациях серого) для определения высоты тона или длительности звука. Изображение сканируется построчно или столбцами. Яркость пикселя преобразуется в частоту (высоту тона) или продолжительность ноты.

Принцип работы можно разбить на 7 частей:

– Растеризация. Изображение разбивается на сетку пикселей.

– Сопоставление. Яркость пикселя (или среднее значение для цветных изображений) нормализуется в диапазон (например, 0-1 или 0-127).

– Высота тона. Значение пикселя преобразуется в частоту. Например, 0 соответствует минимальной частоте, а 127 – максимальной.

– Длительность. Значение пикселя определяет длительность ноты. Более яркий пиксель – более длинная нота, и наоборот.

– Генерация звука. На каждый пиксель (или группу пикселей) генерируется звук соответствующей частоты и длительности.

– Вариации. Можно использовать другие цветовые каналы (красный, зеленый, синий) для управления разными параметрами звука, например, громкостью, панорамой или тембром.

– Сканирование. Может проходить горизонтальным, вертикальным, спиральным или любым другим способом.

В качестве примера можно привести следующее: изображение ярко освещенного объекта в темном фоне будет звучать как серия высоких тонов на фоне тишины [1].

Используемые для данного метода инструменты/библиотеки включают в себя: *Python* (*PIL*/*Pillow* для обработки изображений, *librosa* или *sounddevice* для генерации звука), *Pure Data*, *Max*/*MSP*.

2. Спектрограмма. Изображение напрямую используется как спектрограмма. Спектрограмма визуально отображает частоты звука во времени.

Принцип работы можно разбить на 2 части:

– Интерпретация изображения. Строки изображения интерпретируются как временные отрезки, а столбцы - как частоты. Яркость пикселя соответствует амплитуде (громкости) данной частоты в данный момент времени.

– Преобразование в звук. Используется обратное преобразование Фурье (*Inverse FFT - iFFT*) для создания звукового сигнала из спектрограммы.

Этот метод может создавать довольно сложные и интересные звуковые текстуры, особенно если изображение содержит четкие структуры, однако требует знания работы спектрограмм и *iFFT* и результат может быть довольно хаотичным и трудным для интерпретации.

Приведем пример. Изображение с горизонтальными линиями будет звучать как серия постоянных тонов, а изображение с вертикальными линиями - как быстро меняющиеся частоты.

Используемые для данного метода инструменты/библиотеки включают в себя: *Python* (*librosa*, *scipy.fft*), *MATLAB*, *Sonic Visualiser.*

3. Анализ краев (*Edge Detection*) и ритм/тембр. Сначала выделяются контуры изображения (например, с помощью фильтра Собеля или Канни). Затем, положение и ориентация этих контуров используются для управления ритмом и тембром звука.

Принцип работы можно разбить на 3 части:

– Детекция краев. Применяется алгоритм обнаружения краев для выделения границ объектов на изображении.

– Ритм. Плотность краев в определенной области изображения определяет ритм. Больше краев равняются более быстрому ритму.

– Тембр. Ориентация краев может влиять на тембр. Например, горизонтальные линии могут генерировать один тембр, а вертикальные - другой. Длина линий может влиять на длительность ноты или на параметры синтезатора (например, *cutoff* фильтра).

Приведем пример. Изображение с множеством острых углов и деталей может создать быстрый и резкий ритм, а изображение с плавными кривыми - более медленный и мягкий.

Используемые для данного метода инструменты/библиотеки включают в себя: *Python* (*OpenCV* для обработки изображений, *librosa*, *PySynth*, *Csound*, *SuperCollider* для генерации звука).

4. Разделение на области и инструменты/звуки. Изображение разбивается на отдельные области (например, с помощью сегментации или кластеризации). Каждой области назначается определенный инструмент, звук или патч синтезатора. Размер и цвет области управляют параметрами этого звука (громкость, высота тона, панорама).

Принцип работы можно разбить на 3 части:

– Сегментация. Изображение разбивается на сегменты с похожими характеристиками (например, цвет, текстура).

– Назначение звуков. Каждому сегменту назначается определенный звук (например, пианино, гитара, барабан, синтезированный тон).

– Параметры управления. Размер сегмента может управлять громкостью, средний цвет сегмента – высотой тона, положение сегмента – панорамой.

Приведем пример. Изображение с большим синим пятном может звучать как продолжительный звук синтезатора с низкой частотой, а маленькое красное пятно - как короткий и высокий звук скрипки.

Используемые для данного метода инструменты/библиотеки включают в себя: *Python* (*OpenCV*, *scikit-image* для сегментации изображений, *MIDIUtil*, *PySynth*, *Csound*, *SuperCollider* для генерации звука), *DAW* с поддержкой скриптов (*Ableton Live*, *Max*/*MSP*).

5. Использование алгоритмов машинного обучения (*Object Detection*/*Classification*). Используются модели машинного обучения для распознавания объектов на изображении. Распознанные объекты и их атрибуты (размер, положение, цвет) используются для управления параметрами звука [2].

Принцип работы можно разбить на 3 части:

– *Object Detection*/*Classification*. Модель машинного обучения, обученная на распознавание объектов (например, *YOLO*, *SSD*, *ResNet*), анализирует изображение и определяет, какие объекты на нем находятся.

– Сопоставление с звуками. Каждому типу объекта назначается определенный звук или инструмент. Например, обнаружение «кошки» может запустить звук «мяу», а обнаружение «автомобиля» - звук двигателя.

– Параметры управления: Размер объекта может влиять на громкость, положение – на панораму, цвет – на тембр.

Приведем пример. Изображение с кошкой и собакой может сгенерировать одновременно звуки «мяу» и «гав». Чем больше кошка на изображении, тем громче будет «мяу».

Используемые для данного метода инструменты/библиотеки включают в себя: *Python* (*TensorFlow*, *PyTorch* для машинного обучения, *librosa*, *PyAudio* для генерации звука), облачные *API* для распознавания изображений (*Google Cloud Vision API*, *Amazon Rekognition*).

6. Преобразование текстуры в звук. Анализируется текстура изображения (например, с помощью фильтров Габора или анализа текстуры на основе матриц совместной встречаемости (*GLCM*)). Параметры текстуры (например, контраст, энергия, однородность) используются для управления параметрами звукового синтезатора.

Принцип работы можно разбить на 2 части:

– Анализ текстуры. Используются алгоритмы анализа текстуры для извлечения характеристик текстуры изображения.

– Анализ параметров синтезатора. Параметры текстуры сопоставляются с параметрами звукового синтезатора (например, частота, амплитуда, фильтры, модуляция). Например, высокий контраст может соответствовать высоким частотам, а низкая энергия - низким частотам.

– Приведем пример. Изображение с грубой текстурой (например, кора дерева) может генерировать шумный и резкий звук, а изображение с гладкой текстурой (например, шелк) - мягкий и гармоничный.

Используемые для данного метода инструменты/библиотеки включают в себя: *Python* (*scikit-image* для анализа текстуры, *librosa*, *PySynth*, *Csound*, *SuperCollider* для генерации звука), *MATLAB*.

7. Использование фракталов. Изображение используется для создания фрактальной структуры, которая затем сонифицируется.

Принцип работы можно разбить на 2 части:

– Фрактальная генерация. Пиксели изображения используются в качестве входных данных для генерации фрактальной структуры. Например, яркость пикселей может влиять на параметры итераций множества Мандельброта.

– Сонификация фрактала. Фрактальная структура сонифицируется. Например, можно сканировать фрактал и использовать его структуру для управления высотой тона, длительностью и тембром звука.

Приведем пример. Изображение с простыми геометрическими формами может создать сложные и повторяющиеся звуковые паттерны.

Используемые для данного метода инструменты/библиотеки включают в себя: *Python* (*NumPy*, *matplotlib* для визуализации фракталов, *librosa*, *Csound*, *SuperCollider* для генерации звука), специализированные инструменты для генерации фракталов [3].

8. Использование алгоритмов обработки естественного языка (*Natural Language Processing – NLP*). Изображение сначала описывается с использованием алгоритмов компьютерного зрения и *NLP* (например, *image*

captioning). Затем, текстовое описание изображения преобразуется в звук с помощью *text-to-speech* (*TTS*) или других методов звукового синтеза.

Принцип работы можно разбить на 2 части:

– *Image Captioning*. Используется модель *NLP* для генерации текстового описания изображения. Например, изображение кошки, сидящей на столе, может быть описано как «*a cat sitting on a table*».

– *Text-to-Speech* (*TTS*) или *Sonic Synthesis from Text*. Текстовое описание преобразуется в звук. Можно использовать стандартные движки *TTS* или более продвинутые методы, которые позволяют контролировать тембр, ритм и другие параметры звука на основе смысла текста.

Приведем пример. Изображение с описанием «*a smiling woman wearing a red hat*» будет звучать как женский голос, говорящий эту фразу.

Используемые для данного метода инструменты/библиотеки включают в себя: *Python* (*TensorFlow*, *PyTorch*, *Google Cloud Text-to-Speech API*, *Festival*, *Espeak* для *TTS*, библиотеки для генерации звука на основе текста).

9. *Data Bending*. Этот метод заключается в прямом изменении двоичного кода изображения и его интерпретации как аудиоданных.

Принцип работы можно разбить на 3 части:

– Открытие файла изображения как бинарного. Изображение открывается как обычный бинарный файл.

– Манипуляции с байтами. В бинарном коде изображения случайно или систематически изменяются байты.

– Интерпретация как аудио. Измененные бинарные данные затем интерпретируются как *RAW* аудиоданные (например, 8-*bit*, 16-*bit PCM*).

Это очень простой в реализации метод, который может дать неожиданные звуки, однако звук часто неприятный и неконтролируемый, а также довольно трудно получить осмысленные результаты.

Используемые для данного метода инструменты/библиотеки включают в себя: *Hex*-редакторы, аудиоредакторы (*Audacity*, *Logic Pro X*), *Python*.

10. Использование игровых движков (*Game Engines*). Изображение используется как текстура в 3*D* сцене игрового движка. Звук генерируется в реальном времени на основе взаимодействия с этой 3*D* сценой.

Принцип работы можно разбить на 3 части:

– Импорт изображения. Изображение импортируется в игровой движок (*Unity*, *Unreal Engine*) как текстура.

– Создание 3*D* сцены. Текстура накладывается на 3*D* объект (например, плоскость, куб).

– Взаимодействие. Игрок (или алгоритм) перемещается по 3*D* сцене, и звук генерируется на основе цвета и положения пикселей текстуры, с которыми происходит взаимодействие. Приведем пример. Можно создать 3*D* лабиринт на основе изображения, и звук будет меняться в зависимости от того, где находится игрок в лабиринте.

Используемые для данного метода инструменты/библиотеки включают в себя: *Unity*, *Unreal Engine*, *FMOD*, *Wwise* (для звукового дизайна) [4].

Однако перед применением вышеперечисленных методов следует помнить несколько рекомендаций:

1. Перед преобразованием важно масштабировать и нормализовать значения пикселей, чтобы они соответствовали диапазону значений, ожидаемых алгоритмом генерации звука.

2. Чтобы избежать резких переходов в звуке, можно использовать фильтры сглаживания для значений пикселей.

3. После генерации звука можно применять различные аудиоэффекты (например, реверберация, задержка, дисторшн) для улучшения звучания.

Важно понимать, что преобразование изображения в аудиофайл - во многом субъективный процесс. Нет «правильного» или «неправильного» способа сделать это. Необходимо экспериментировать с разными подходами и параметрами, чтобы найти то, что требуется.

Литература:

1. Звук – теория [Электронный ресурс]. – Режим доступа: https://soundmain.ru/articles/zvuk-teoriya-chast-1.41/?ysclid=mb7ou9tq3g967883497.

2. Алгоритмы распознавания объектов [Электронный ресурс]. – Режим доступа: https://moluch.ru/conf/tech/archive/166/10825/?ysclid=mb8gpe0thj784541601.

3. Анализ методов сегментации текстурных областей изображений [Электронный ресурс]. – Режим доступа: https://cyberleninka.ru/article/n/analiz-metodov-segmentatsii-teksturnyh-oblastey-izobrazheniy-v-sistemah-obrabotki-izobrazheniy/viewer.

4. Генерация музыки из изображений [Электронный ресурс]. – Режим доступа: https://habr.com/ru/companies/ruvds/articles/708890/.

а

б

в

г

Рис. 1 Типовые модели АПК: а – «СПРУТ-11», б – «СПРУТ-7М», в – «Шепот-М1», г – «СМАРТ-АВ»

Таблица 1

Технические характеристики АПК

Характеристика	**Изделие**			
	СПРУТ-11	**СПРУТ-7М**	**Шепот**	**СМАРТ-АВ**
Диапазон частот уровней звукового давления, Гц	20–20000	20–20000	80–11200	20–11500
Диапазон частот измерений виброускорения, Гц	5–12500	5–12500	20–11200	20–11200
Диапазон уровней звукового давления, дБ	25–124	25–140	24–132	25–120
Диапазон уровней виброускорения, м/с^2 (дБ)	80–174	80–174	80–150	55–200
Габариты прибора, мм	551×407×439	662×448×337	340×486×220	260×560×260
Наработка на отказ, час	2000	2000	1000	2000
Гарантия, мес	12	12	12	12