

一种音频处理方法及装置

本公开要求于 2023 年 08 月 29 日提交的中国专利申请 202311099735.X 的优先权，其全部内容通过援引加入本文。

5

技术领域

本公开涉及数据处理技术领域，尤其涉及一种音频处理方法及装置。

背景技术

在工作中，用户端需要自动收集音频或者通过插件或者 SDK（英文全
10 称:Software Development Kit，中文全称：软件开发工具包）的方式引入音
频进行音频压缩和可视化。

目前，进行音频压缩和可视化的方法是采用 FFmpeg 方法或实例方法
用于音频编码压缩，通过 Linux 平台或者服务端进行处理以文字形式可视
化展示；其中，所述 FFmpeg 方法是指在 Linux 平台下开发的可以用来记
15 录、转换数字音频并将其转化的开源计算机程序；所述实例方法是
JavaScript 方法中的一种对象方法，通过实例方法更改数据参数进行音频编
码压缩。

但是，上述处理方法都是采用 C/C++进行处理，且都是通过安卓/iOS
原生进行处理，因此，无法直接运行在浏览器、H5、小程序等用户端；并
20 且都是以文字形式可视化展示，没有可视化到音轨中，无法进行音频可视
化分析。

发明内容

鉴于上述问题，本公开提供一种音频处理方法及装置，主要目的是为了
实现直接运行于不同用户端，且可视化到音轨中进行音频可视化分析。

25

为解决上述技术问题，本公开提出以下方案：

第一方面，本公开提供一种音频处理方法，所述方法包括：

获取执行命令；

基于所述执行命令，利用预设实例方法从预设配置文件中获取所述执
行命令对应的配置数据，其中，所述预设配置文件至少包括音频输出模

式、音频编码方式和音频文件格式；

基于所述执行命令，利用预设获取规则获取音频数据；

基于所述执行命令对应的配置数据，利用预设处理规则对所述音频数据进行音频压缩和格式处理，得到目标音频数据；

5 基于所述目标音频数据，利用预设可视化规则进行可视化处理，得到所述目标音频数据对应的可视化显示，所述可视化显示包括音轨和文本标注。

第二方面，本公开提供一种音频处理装置，所述装置包括：

第一获取单元，用于获取执行命令；

10 第二获取单元，用于基于所述执行命令，利用预设实例方法从预设配置文件中获取所述执行命令对应的配置数据，其中，所述预设配置文件至少包括音频输出模式、音频编码方式和音频文件格式；

第三获取单元，用于基于所述执行命令，利用预设获取规则获取音频数据；

15 处理单元，用于基于所述执行命令对应的配置数据，利用预设处理规则对所述音频数据进行音频压缩和格式处理，得到目标音频数据；

可视化单元，用于基于所述目标音频数据，利用预设可视化规则进行可视化处理，得到所述目标音频数据对应的可视化显示，所述可视化显示包括音轨和文本标注。

20 为了实现上述目的，根据本公开的第三方面，提供了一种存储介质，所述存储介质包括存储的程序，其中，在所述程序运行时控制所述存储介质所在设备执行上述第一方面所述音频处理方法。

为了实现上述目的，根据本公开的第四方面，提供了一种电子设备，包括存储器、处理器及存储在所述存储器上并可在所述处理器上运行的计算机程序，所述处理器执行所述程序时实现如第二方面所述用于音频处理装置的全部或部分步骤。

25 借由上述技术方案，本公开提供的音频处理方法及装置，是由于采用FFmpeg方法或实例方法用于音频编码压缩，通过Linux平台或者服务端进行处理以文字形式可视化展示存在无法直接运行在浏览器、H5、小程序等

用户端；并且都是以文字形式可视化展示，没有可视化到音轨中，无法进行音频可视化分析的问题。为此，本公开通过获取执行命令；基于所述执行命令，利用预设实例方法从预设配置文件中获取所述执行命令对应的配置数据，其中，所述预设配置文件至少包括音频输出模式、音频编码方式和音频文件格式；基于所述执行命令，利用预设获取规则获取音频数据；
5 基于所述执行命令对应的配置数据，利用预设处理规则对所述音频数据进行音频压缩和格式处理，得到目标音频数据；基于所述目标音频数据，利用预设可视化规则进行可视化处理，得到所述目标音频数据对应的可视化显示，所述可视化显示包括音轨和文本标注。本公开可以实现降低音频可
10 视化的门槛，且可以可视化到音轨中进行音频可视化分析；通过音频压缩减少用户和企业流量以及储存成本；可兼容不同终端包括 PC、H5、小程序等；可选择其他自定义的配置选项，便于在不同场景下的应用。

上述说明仅是本公开技术方案的概述，为了能够更清楚了解本公开的技术手段，而可依照说明书的内容予以实施，并且为了让本公开的上述和
15 其它目的、特征和优点能够更明显易懂，以下特举本公开的具体实施方式。

附图说明

通过阅读下文优选实施方式的详细描述，各种其他的优点和益处对于本领域普通技术人员将变得清楚明了。附图仅用于示出优选实施方式的目的，而并不认为是对本公开的限制。而且在整个附图中，用相同的参考符
20 号表示相同的部件。在附图中：

图 1 示出了本公开实施例提供的一种音频处理方法流程图；

图 2 示出了本公开实施例提供的另一种音频处理方法流程图；

图 3 示出了本公开实施例提供的一种音频处理装置的组成框图；

25 图 4 示出了本公开实施例提供的另一种音频处理装置的组成框图。

具体实施方式

下面将参照附图更详细地描述本公开的示例性实施例。虽然附图中显示了本公开的示例性实施例，然而应当理解，可以以各种形式实现本公开而不应被这里阐述的实施例所限制。相反，提供这些实施例是为了能够更

透彻地理解本公开，并且能够将本公开的范围完整的传达给本领域的技术人员。

术语解释：

5 FFmpeg：是一套可以用来记录、转换数字音频、视频，并能将其转化为流的开源计算机程序。采用 LGPL 或 GPL 许可证。它提供了录制、转换以及流化音视频的完整解决方案。

Linux：全称 GNU/Linux，是一种免费使用和自由传播的类 UNIX 操作系统，其内核由林纳斯·本纳第克特·托瓦兹（Linus Benedict Torvalds）发布，它是一个基于 POSIX 的多用户、多任务、支持多线程和多 CPU 的操作系统。它支持 32 位和 64 位硬件，能运行主要的 Unix 工具软件、应用程序和网络协议。

Webassembly：简称 wasm，是一个虚拟指令集体系架构（virtual ISA），整体架构包括核心的 ISA 定义、二进制编码、程序语义的定义与执行，以及面向不同的嵌入环境（如 Web）的应用编程接口（WebAssembly API）。

H5：HTML5 是构建 Web 内容的一种语言描述方式。HTML5 是互联网的下一代标准，是构建以及呈现互联网内容的一种语言方式，被认为是互联网的核心技术之一。

20 PCM：PCM（Pulse Code Modulation）脉冲编码调制是数字通信的编码方式之一。主要过程是将话音、图像等模拟信号每隔一定时间进行取样，使其离散化，同时将抽样值按分层单位四舍五入取整量化，同时将抽样值按一组二进制码来表示抽样脉冲的幅值。

25 WAV：是最常见的声音文件格式之一，是微软公司专门为 Windows 开发的一种标准数字音频文件，该文件能记录各种单声道或立体声的声音信息，并能保证声音不失真。

MP3：是一种音频压缩技术，其全称是动态影像专家压缩标准音频层面 3（Moving Picture Experts Group Audio Layer III），简称为 MP3。它被设计用来大幅度地降低音频数据量。利用 MPEG Audio Layer 3 的技术，将音乐以 1:10 甚至 1:12 的压缩率，压缩成容量较小的文件，而对于大多数

用户来说重放的音质与最初的不压缩音频相比没有明显的下降。

Ogg: 全称是 OGGVobis(oggVorbis)是一种音频压缩格式, 类似于 MP3 等的音乐格式。Ogg 是完全免费、开放和没有专利限制的。OggVorbis 文件的扩展名是".ogg"。Ogg 文件格式可以不断地进行大小和音质的改良, 而不影响旧有的编码器或播放器。

Canvas: Canvas API (画布) 是在 HTML5 中新增的标签用于在网页实时生成图像, 并且可以操作图像内容, 基本上它是一个可以用 JavaScript 操作的位图(bitmap)。

JavaScript: 简称“JS”, 是一种具有函数优先的轻量级, 解释型或即时编译型的编程语言。虽然它是作为开发 Web 页面的脚本语言而出名, 但是它也被用到了很多非浏览器环境中, JavaScript 基于原型编程、多范式的动态脚本语言, 并且支持面向对象、命令式、声明式、函数式编程范式。

ASR: 语音识别技术, 也被称为自动语音识别 (Automatic Speech Recognition, ASR), 其目标是将人类的语音中的词汇内容转换为计算机可读的输入, 例如按键、二进制编码或者字符序列。与说话人识别及说话人确认不同, 后者尝试识别或确认发出语音的说话人而非其中所包含的词汇内容。

Opus: 是一个有损声音编码的格式, 由 Xiph.Org 基金会开发, 之后由 IETF (互联网工程任务组) 进行标准化, 目标是希望用单一格式包含声音和语音, 取代 Speex 和 Vorbis, 且适用于网络上低延迟的即时声音传输, 标准格式定义于 RFC 6716 文件。Opus 格式是一个开放格式, 使用上没有任何专利或限制。

CDN: 全称是 Content Delivery Network, 即内容分发网络。通过在网络各处放置节点服务器所构成的在现有的互联网基础之上的一层智能虚拟网络, CDN 系统能够实时地根据网络流量和各节点的连接、负载状况以及到用户的距离和响应时间等综合信息将用户的请求重新导向离用户最近的服务节点上。其目的是使用户可就近取得所需内容, 解决 Internet 网络拥挤的状况, 提高用户访问网站的响应速度。

Raw：通常称为“源码”。意义是把光盘上的音频格式不加处理地从同轴和光纤输出。

对于目前采用 FFmpeg 方法或实例方法用于音频编码压缩，通过 Linux 平台或者服务端进行处理以文字形式可视化展示；但是由于上述是采用 C/C++ 进行处理，且都是通过安卓/IOS 原生进行处理，因此，无法直接运行在浏览器、H5、小程序等用户端；并且都是以文字形式可视化展示，没有可视化到音轨中，无法进行音频可视化分析。针对此问题，发明人想到利用预设实例方法和预设配置文件配合二次压缩音频处理音频进行音轨结合文本标注进而可视化分析。

为此，本公开实施例提供了一种音频处理方法，通过该方法实现直接运行于不同用户端，且可视化到音轨中进行音频可视化分析，其具体执行步骤如图 1 所示，包括：

101、获取执行命令。

当用户端操作需要完成的音频处理项目时，用户端需要先通过预先设置的调用地址调用预先设置的实例方法，再通过这个实例方法选择要完成的音频处理项目，例如：音频压缩、音频可视化等，本实施例不做具体限定；当选定所要处理的音频处理项目时，生成执行命令。

102、基于执行命令，利用预设实例方法从预设配置文件中获取执行命令对应的配置数据。

其中，所述预设配置文件至少包括音频输出模式、音频编码方式和音频文件格式。其中，所述音频输出模式可以为“Raw”和“PCM”，所述音频编码方式可以为“波形编码”和“混合编码”，所述音频文件格式可以为 WAV、MP3、ogg 等格式，本实施例不做具体限定。

从步骤 101 可得所述执行命令，解析所述执行命令可以得到完成选定的音频处理项目所需的配置项目，例如：音频输出模式、音频编码方式等，本实施例不做具体限定；根据所需的配置项目从所述预设配置文件中获取对应的配置数据，用于后续对采集到的音频数据进行处理。

103、基于执行命令，利用预设获取规则获取音频数据。

从步骤 101 得到生成的所述执行命令，需要争取用户端的授权，只有

当用户端同意采集音频数据才能去获取所述音频数据，因此，可以设置预设获取规则，该预设获取规则可以是判断用户端是否授权同意获取音频数据，当用户端授权时，可以进行采集音频数据，用于完成音频处理项目

104、基于执行命令对应的配置数据，利用预设处理规则对音频数据
5 进行音频压缩和格式处理，得到目标音频数据。

从步骤 103 可得所述执行命令对应的配置数据，该配置数据中包括完成音频压缩和格式处理所需的数据，例如：采样率、音频文件格式等，本实施例不做具体限定。

根据配置数据，可以利用 Webassembly 封装 FFmpeg 的方法对所述音
10 频数据直接进行压缩得到目标音频数据，也可以先将所述音频数据基于压缩率进行初次压缩后，再利用 Webassembly 封装 FFmpeg 的方法进行二次压缩得到目标音频数据，本实施例不做具体限定。所述目标音频数据是经过音频压缩和格式转化后的音频数据，可用于可视化操作。

105、基于目标音频数据，利用预设可视化规则进行可视化处理，得
15 到目标音频数据对应的可视化显示。

其中，所述可视化显示包括音轨和文本标注。所述预设可视化规则可以是利用 Canvas 和 JavaScript 将所述目标音频数据形成音轨可视化显示。

从步骤 104 可得所述目标音频数据，利用 Canvas 和 JavaScript 将所述
20 目标音频数据形成音轨可视化显示，同时基于音频与文本的对应关系进行标注文本，例如：关键词、解释等，本实施例不做具体限定。并且在该可视化显示上可以通过划词或拖拽音轨来进行音轨分析等操作。

基于上述图 1 实施例的实现方式可以看出，本公开提供一种音频处理方法，本公开通过获取执行命令；基于所述执行命令，利用预设实例方法从预设配置文件中获取所述执行命令对应的配置数据，其中，所述预设配置
25 文件至少包括音频输出模式、音频编码方式和音频文件格式；基于所述执行命令，利用预设获取规则获取音频数据；基于所述执行命令对应的配置数据，利用预设处理规则对所述音频数据进行音频压缩和格式处理，得到目标音频数据；基于所述目标音频数据，利用预设可视化规则进行可视化处理，得到所述目标音频数据对应的可视化显示，所述可视化显示包括

音轨和文本标注。本公开可以实现降低音频可视化的门槛，且可以可视化到音轨中进行音频可视化分析；通过音频压缩减少用户和企业流量以及储存成本；可兼容不同终端包括 PC、H5、小程序等；可选择其他自定义的配置选项，便于在不同场景下的应用。

5 进一步的，作为对图 1 所示实施例的细化及扩展，本公开实施例还提供了另一种音频处理方法，如图 2 所示，其具体步骤如下：

201、利用预设调用规则调用预设实例方法。

开发人员会在音频处理项目前先将所述预设实例方法设计好，保存在预设的储存地址中，用户端通过该预设的储存地址可以调用所述预设实例方法，或者，也可以利用移动储存设备上传到用户端进行使用。

例如：通过设置命令下载或者进行 CDN 方式引入，调用预设实例化方法。

202、利用预设实例方法创建预设配置文件。

通过步骤 201 调用预先设置的预设实例方法后，可以通过所述预设实例方法创建预设配置文件，创建的过程是通过所述预设实例方法设置配置项目和配置项目对应的数据，例如：配置项目可选择音频文件格式，然后对应的数据设置为“WAV”；本实施例不做具体限定。

203、获取执行命令。

本步骤结合上述方法中 101 步骤的描述，在此相同的内容不赘述。

20 204、基于执行命令，利用预设实例方法从预设配置文件中获取执行命令对应的配置数据。

本步骤结合上述方法中 102 步骤的描述，在此相同的内容不赘述。

从步骤 203 可得所述执行命令，然后解析所述执行命令得到完成所述执行命令所需的配置项目，例如：音频输出模式、音频编码方式和音频文件格式等，本实施例不做具体限定；基于所述配置项目，利用预设实例方法从预设配置文件中获取所述配置项目对应的配置数据，得到所述执行命令对应的配置数据，例如：音频输出模式为“PCM”、音频编码方式为“波形编码”和音频文件格式“MP3”等。

205、基于执行命令，利用预设获取规则获取音频数据。

本步骤结合上述方法中 103 步骤的描述，在此相同的内容不赘述。

从步骤 203 可得所述执行命令，然后基于所述执行命令，判断用户端是否授权获取所述音频数据；若是，则获取所述音频数据；若否，则回到上一步执行动作。

5 206、基于执行命令对应的配置数据，利用预设处理规则对音频数据进行音频压缩和格式处理，得到目标音频数据。

本步骤结合上述方法中 104 步骤的描述，在此相同的内容不赘述。

所述配置数据可以包括收集采样率和输出采样率，本实施例不做具体限定；从步骤 205 得到所述执行命令对应的配置数据后，首先，计算所述
10 收集采样率和所述输出采样率的比值，得到压缩率，即该比值为所述压缩率；基于所述压缩率，利用预设压缩规则压缩所述音频数据，得到初级压缩音频数据；利用 Webassembly 封装 FFmpeg 的方法对所述初级压缩音频数据集进行二次压缩（即进行 opus 压缩），得到目标音频数据，进一步提高压缩能力。

15 其中，所述基于所述压缩率，利用预设压缩规则压缩所述音频数据，得到初级压缩音频数据，包括：识别所述音频数据中存在的噪音数据和空数据，且进行删除处理，得到待压缩音频数据，此处理可以提高压缩能力；根据所述压缩率对所述待压缩音频数据进行压缩，得到所述初级压缩音频数据。

20 举例说明：

客户端收集音频进行采样率处理 48000/44100Hz 转化为 8000/16000Hz, 然后进行编码压缩把 pcm 流压缩为 opus 数据流，并且支持把 pcm 流转化成 WAV、mp3、ogg 等音频格式，用于音频流传输。

25 207、基于目标音频数据，利用预设可视化规则进行可视化处理，得到目标音频数据对应的可视化显示。

本步骤结合上述方法中 105 步骤的描述，在此相同的内容不赘述。

从步骤 206 可得所述目标音频数据，然后利用 ASR 识别所述目标音频数据获取音频和文本对应关系，可以储存在服务端；基于所述音频和文本对应关系以及所述目标音频数据（可以从服务端获取得到），利用 Canvas

和 JavaScript 形成音轨可视化显示，即 JavaScript 控制 Canvas 画音轨，所述音轨可视化显示包括文本，例如：关键词放在音轨中；基于所述音轨可视化显示，利用预设移动命令进行音频分析和标注。所述预设移动命令可以为划词或者拖拽音轨，即通过划词和拖拽音轨的方式实现音频分析和标注。

基于上述图 2 的实现方式可以看出，本公开提供一种音频处理方法，采用本公开的二次压缩处理方法可以使音频压缩后占用内存更小，可以克服现有技术中数据包太大对用户端内存占用太大，采用 C/C++进行封装导致包大并且 C/C++运行时消耗内存大的问题，同时可以在不同终端中更方便的进行音频收集和传输；本公开由于采用预设实例方法和预设配置文件的方法适用于不同用户端进行使用音频压缩，克服了现有技术中在使用时不能直接应用于用户端的问题；本公开可以进行音轨结合文本进行音轨可视化显示，解决音频在企业数据分析中，进行可视化内容分析的问题，扩展音频资产的价值挖掘方式和效率。

进一步的，作为对上述图 1 所示方法的实现，本公开实施例还提供了一种音频处理装置，用于对上述图 1 所示的方法进行实现。该装置实施例与前述方法实施例对应，为便于阅读，本装置实施例不再对前述方法实施例中的细节内容进行逐一赘述，但应当明确，本实施例中的装置能够对应实现前述方法实施例中的全部内容。如图 3 所示，该装置包括：

- 第一获取单元 31，用于获取执行命令；
- 第二获取单元 32，用于基于从所述第一获取单元 31 得到的所述执行命令，利用预设实例方法从预设配置文件中获取所述执行命令对应的配置数据，其中，所述预设配置文件至少包括音频输出模式、音频编码方式和音频文件格式；
- 第三获取单元 33，用于基于从所述第一获取单元 31 得到的所述执行命令，利用预设获取规则获取音频数据；
- 处理单元 34，用于基于从所述第二获取单元 32 得到的所述执行命令对应的配置数据，利用预设处理规则对从所述第三获取单元 33 得到的所述音频数据进行音频压缩和格式处理，得到目标音频数据；

可视化单元 35，用于基于从所述处理单元 34 得到的所述目标音频数据，利用预设可视化规则进行可视化处理，得到所述目标音频数据对应的可视化显示，所述可视化显示包括音轨和文本标注。

进一步的，作为对上述图 2 所示方法的实现，本公开实施例还提供了
5 另一种音频处理装置，用于对上述图 2 所示的方法进行实现。该装置实施例与前述方法实施例对应，为便于阅读，本装置实施例不再对前述方法实施例中的细节内容进行逐一赘述，但应当明确，本实施例中的装置能够实现前述方法实施例中的全部内容。如图 4 所示，该装置包括：

第一获取单元 31，用于获取执行命令；

10 第二获取单元 32，用于基于从所述第一获取单元 31 得到的所述执行命令，利用预设实例方法从所述创建单元 37 得到的预设配置文件中获取所述执行命令对应的配置数据，其中，所述预设配置文件至少包括音频输出模式、音频编码方式和音频文件格式；

15 第三获取单元 33，用于基于从所述第一获取单元 31 得到的所述执行命令，利用预设获取规则获取音频数据；

处理单元 34，用于基于从所述第二获取单元 32 得到的所述执行命令对应的配置数据，利用预设处理规则对从所述第三获取单元 33 得到的所述音频数据进行音频压缩和格式处理，得到目标音频数据；

20 可视化单元 35，用于基于从所述处理单元 34 得到的所述目标音频数据，利用预设可视化规则进行可视化处理，得到所述目标音频数据对应的可视化显示，所述可视化显示包括音轨和文本标注；

调用单元 36，用于利用预设调用规则调用所述预设实例方法；

创建单元 37，用于利用从所述调用单元 36 得到的所述预设实例方法创建所述预设配置文件。

25 进一步的，所述配置数据至少包括收集采样率和输出采样率；所述处理单元 34，包括：

计算模块 341，用于计算所述收集采样率和所述输出采样率的比值，得到压缩率；

第一压缩模块 342，用于基于从所述计算模块 341 得到的所述压缩

率，利用预设压缩规则压缩所述音频数据，得到初级压缩音频数据；

第二压缩模块 343，用于利用 Webassembly 封装 FFmpeg 的方法对从所述第一压缩模块 342 得到的所述初级压缩音频数据集进行二次压缩，得到目标音频数据。

5 进一步的，所述第一压缩模块 342，包括：

识别子模块 3421，用于识别所述音频数据中存在的噪音数据和空数据，且进行删除处理，得到待压缩音频数据；

压缩子模块 3422，用于根据所述压缩率对从所述识别子模块 3421 得到的所述待压缩音频数据进行压缩，得到所述初级压缩音频数据。

10 进一步的，所述可视化单元 35，包括：

识别模块 351，用于利用 ASR 识别所述目标音频数据获取音频和文本对应关系；

15 可视化模块 352，用于基于从所述识别模块 351 得到的所述音频和文本对应关系以及所述目标音频数据，利用 Canvas 和 JavaScript 形成音轨可视化显示，所述音轨可视化显示包括文本；

分析模块 353，用于基于从所述可视化模块 352 得到的所述音轨可视化显示，利用预设移动命令进行音频分析和标注。

进一步的，所述第二获取单元 32，包括：

20 解析模块 321，用于解析所述执行命令得到完成所述执行命令所需的配置项目；

获取模块 322，用于基于从所述解析模块 321 得到的所述配置项目，利用预设实例方法从预设配置文件中获取所述配置项目对应的配置数据，得到所述执行命令对应的配置数据。

进一步的，所述第三获取单元 33，包括：

25 判断模块 331，用于基于所述执行命令，判断用户端是否授权获取所述音频数据；

获取模块 332，用于若从所述判断模块 331 得到的用户端授权获取所述音频数据，则获取所述音频数据；

返回模块 333，用于若从所述判断模块 331 得到的用户端不授权获取

所述音频数据，则回到上一步执行动作。

进一步的，本公开实施例还提供一种处理器，所述处理器用于运行程序，其中，所述程序运行时执行上述图 1-2 中所述的音频处理方法。

进一步的，本公开实施例还提供一种存储介质，所述存储介质用于存储计算机程序，其中，所述计算机程序运行时控制所述存储介质所在设备
5 执行上述图 1-2 中所述的音频处理方法。

在上述实施例中，对各个实施例的描述都各有侧重，某个实施例中没
有详述的部分，可以参见其他实施例的相关描述。

可以理解的是，上述方法及装置中的相关特征可以相互参考。另外，
10 上述实施例中的“第一”、“第二”等是用于区分各实施例，而并不代表
各实施例的优劣。

所属领域的技术人员可以清楚地了解到，为描述的方便和简洁，上述
描述的系统，装置和单元的具体工作过程，可以参考前述方法实施例中的
对应过程，在此不再赘述。

15 在此提供的算法和显示不与任何特定计算机、虚拟系统或者其它设备
固有相关。各种通用系统也可以与基于在此的示教一起使用。根据上面的
描述，构造这类系统所要求的结构是显而易见的。此外，本公开也不针对
任何特定编程语言。应当明白，可以利用各种编程语言实现在此描述的本
公开的内容，并且上面对特定语言所做的描述是为了披露本公开的最佳实
20 施方式。

此外，存储器可能包括计算机可读介质中的非永久性存储器，随机存
取存储器(RAM)和/或非易失性内存等形式，如只读存储器(ROM)或闪存
(flash RAM)，存储器包括至少一个存储芯片。

本领域内的技术人员应明白，本公开的实施例可提供为方法、系统、
25 或计算机程序产品。因此，本公开可采用完全硬件实施例、完全软件实
施例、或结合软件和硬件方面的实施例的形式。而且，本公开可采用在一个
或多个其中包含有计算机可用程序代码的计算机可用存储介质（包括但不
限于磁盘存储器、CD-ROM、光学存储器等）上实施的计算机程序产品的
形式。

本公开是参照根据本公开实施例的方法、设备（系统）、和计算机程序产品的流程图和 / 或方框图来描述的。应理解可由计算机程序指令实现流程图和 / 或方框图中的每一流程和 / 或方框、以及流程图和 / 或方框图中的流程和 / 或方框的结合。可提供这些计算机程序指令到通用计算机、
 5 专用计算机、嵌入式处理机或其他可编程数据处理设备的处理器以产生一个机器，使得通过计算机或其他可编程数据处理设备的处理器执行的指令产生用于实现在流程图一个流程或多个流程和 / 或方框图一个方框或多个方框中指定的功能的装置。

这些计算机程序指令也可存储在能引导计算机或其他可编程数据处理
 10 设备以特定方式工作的计算机可读存储器中，使得存储在该计算机可读存储器中的指令产生包括指令装置的制造品，该指令装置实现在流程图一个流程或多个流程和 / 或方框图一个方框或多个方框中指定的功能。

这些计算机程序指令也可装载到计算机或其他可编程数据处理设备上，使得在计算机或其他可编程设备上执行一系列操作步骤以产生计算机
 15 实现的处理，从而在计算机或其他可编程设备上执行的指令提供用于实现在流程图一个流程或多个流程和 / 或方框图一个方框或多个方框中指定的功能的步骤。

在一个典型的配置中，计算设备包括一个或多个处理器 (CPU)、输入/输出接口、网络接口和内存。

20 存储器可能包括计算机可读介质中的非永久性存储器，随机存取存储器 (RAM) 和 / 或非易失性内存等形式，如只读存储器 (ROM) 或闪存 (flash RAM)。存储器是计算机可读介质的示例。

计算机可读介质包括永久性和非永久性、可移动和非可移动媒体可以由任何方法或技术来实现信息存储。信息可以是计算机可读指令、数据结
 25 构、程序的模块或其他数据。计算机的存储介质的例子包括，但不限于相变内存 (PRAM)、静态随机存取存储器 (SRAM)、动态随机存取存储器 (DRAM)、其他类型的随机存取存储器 (RAM)、只读存储器 (ROM)、电可擦除可编程只读存储器 (EEPROM)、快闪记忆体或其他内存技术、只读光盘只读存储器 (CD-ROM)、数字多功能光盘 (DVD) 或其他光学存储、磁盒

式磁带，磁带磁磁盘存储或其他磁性存储设备或任何其他非传输介质，可用于存储可以被计算设备访问的信息。按照本文中的界定，计算机可读介质不包括暂存电脑可读媒体 (transitory media)，如调制的数据信号和载波。

5 还需要说明的是，术语“包括”、“包含”或者其任何其他变体意在涵盖非排他性的包含，从而使得包括一系列要素的过程、方法、商品或者设备不仅包括那些要素，而且还包括没有明确列出的其他要素，或者是还包括为这种过程、方法、商品或者设备所固有的要素。在没有更多限制的情况下，由语句“包括一个……”限定的要素，并不排除在包括要素的过程、方法、商品或者设备中还存在另外的相同要素。

10 本领域技术人员应明白，本公开的实施例可提供为方法、系统或计算机程序产品。因此，本公开可采用完全硬件实施例、完全软件实施例或结合软件和硬件方面的实施例的形式。而且，本公开可采用在一个或多个其中包含有计算机可用程序代码的计算机可用存储介质（包括但不限于磁盘存储器、CD-ROM、光学存储器等）上实施的计算机程序产品的形式。

15 以上仅为本公开的实施例而已，并不用于限制本公开。对于本领域技术人员来说，本公开可以有各种更改和变化。凡在本公开的精神和原理之内所作的任何修改、等同替换、改进等，均应包含在本公开的权利要求范围之内。

20

25

权 利 要 求 书

- 1、一种音频处理方法，其特征在于，所述方法包括：
获取执行命令；
基于所述执行命令，利用预设实例方法从预设配置文件中获取所述执
5 行命令对应的配置数据，其中，所述预设配置文件至少包括音频输出模
式、音频编码方式和音频文件格式；
基于所述执行命令，利用预设获取规则获取音频数据；
基于所述执行命令对应的配置数据，利用预设处理规则对所述音频数
据进行音频压缩和格式处理，得到目标音频数据；
10 基于所述目标音频数据，利用预设可视化规则进行可视化处理，得到
所述目标音频数据对应的可视化显示，所述可视化显示包括音轨和文本标
注。
- 2、根据权利要求 1 所述的方法，其特征在于，在所述获取执行命令
之前，所述方法还包括：
15 利用预设调用规则调用所述预设实例方法；
利用所述预设实例方法创建所述预设配置文件。
- 3、根据权利要求 1 所述的方法，其特征在于，所述配置数据至少包
括收集采样率和输出采样率；
所述基于所述执行命令对应的配置数据，利用预设处理规则对所述音
20 频数据进行音频压缩和格式处理，得到目标音频数据，包括：
计算所述收集采样率和所述输出采样率的比值，得到压缩率；
基于所述压缩率，利用预设压缩规则压缩所述音频数据，得到初级压
缩音频数据；
利用 Webassembly 封装 FFmpeg 的方法对所述初级压缩音频数据集进
25 行二次压缩，得到目标音频数据。
- 4、根据权利要求 3 所述的方法，其特征在于，所述基于所述压缩
率，利用预设压缩规则压缩所述音频数据，得到初级压缩音频数据，包
括：
识别所述音频数据中存在的噪音数据和空数据，且进行删除处理，得

到待压缩音频数据；

根据所述压缩率对所述待压缩音频数据进行压缩，得到所述初级压缩音频数据。

5 5、根据权利要求 1-4 任一项所述的方法，其特征在于，所述基于所述目标音频数据，利用预设可视化规则进行可视化处理，得到所述目标音频数据对应的可视化显示，包括：

利用 ASR 识别所述目标音频数据获取音频和文本对应关系；

基于所述音频和文本对应关系以及所述目标音频数据，利用 Canvas 和 JavaScript 形成音轨可视化显示，所述音轨可视化显示包括文本；

10 基于所述音轨可视化显示，利用预设移动命令进行音频分析和标注。

6、根据权利要求 5 所述的方法，其特征在于，所述基于所述执行命令，利用预设实例方法从预设配置文件中获取所述执行命令对应的配置数据，包括：

解析所述执行命令得到完成所述执行命令所需的配置项目；

15 基于所述配置项目，利用预设实例方法从预设配置文件中获取所述配置项目对应的配置数据，得到所述执行命令对应的配置数据。

7、根据权利要求 5 所述的方法，其特征在于，所述基于所述执行命令，利用预设获取规则获取音频数据，包括：

基于所述执行命令，判断用户端是否授权获取所述音频数据；

20 若是，则获取所述音频数据；

若否，则回到上一步执行动作。

8、一种音频处理装置，其特征在于，包括：

第一获取单元，用于获取执行命令；

25 第二获取单元，用于基于所述执行命令，利用预设实例方法从预设配置文件中获取所述执行命令对应的配置数据，其中，所述预设配置文件至少包括音频输出模式、音频编码方式和音频文件格式；

第三获取单元，用于基于所述执行命令，利用预设获取规则获取音频数据；

处理单元，用于基于所述执行命令对应的配置数据，利用预设处理规

则对所述音频数据进行音频压缩和格式处理，得到目标音频数据；

可视化单元，用于基于所述目标音频数据，利用预设可视化规则进行可视化处理，得到所述目标音频数据对应的可视化显示，所述可视化显示包括音轨和文本标注。

5 9、一种存储介质，所述存储介质包括存储的程序，其特征在于，在所述程序运行时控制所述存储介质所在设备执行权利要求 1 至权利要求 7 中任一项所述音频处理方法。

10 10、一种电子设备，包括存储器、处理器及存储在所述存储器上并可在所述处理器上运行的计算机程序，其特征在于，所述处理器执行所述程序时实现如权利要求 1 至权利要求 7 中任一项所述音频处理方法。

15

20

25