

解码处理方法、装置、计算机设备及存储介质

本申请要求于 2021 年 08 月 20 日提交中国专利局，申请号为 202110965320.0，申请名称为“解码处理方法、装置、计算机设备及存储介质”的中国专利申请的优先权，其全部内容通过引用结合在本申请中。

技术领域

本申请涉及计算机技术领域，尤其涉及一种解码处理方法、一种解码处理装置、一种计算机设备、一种计算机可读存储介质、一种计算机程序产品。

背景技术

在多媒体业务领域中，需要对编码后得到的多媒体数据进行解码处理，然后将解码处理后的多媒体数据进行页面渲染以供用户观看。但在对多媒体数据进行解码处理的过程中，若解码器的性能不够，会导致对多媒体数据的解码过程在频率上的表现不够稳定，从而产生视频卡顿的现象，影响视频观看效果。

针对上述情况，目前常用的做法是调用硬件解码器的应用程序接口（Application Program Interface, API）来利用硬件对视频解码过程做加速处理。但是在利用硬件做解码加速处理之后，在超低延迟应用的场景的高要求下，可能会遇到解码器固有的性能问题（包括解码速度不够快及稳定性不够高等问题），这些问题会降低硬件解码过程的效率。因此，如何稳定解码器的性能问题是当前亟待解决的一个技术问题。

发明内容

根据本申请的各种实施例，提出了一种解码处理方法、装置、计算机设备、存储介质和计算机程序产品。

一方面，本申请实施例提供一种解码处理方法，由计算机设备执行，该方法包括：

在对第一多媒体数据进行硬解码的过程中，获取解码器的运行状态，运行状态包括工作状态或空闲状态；

若所解码器处于空闲状态，则获取第二多媒体数据，第二多媒体数据是对第一多媒体数据进行复制得到的；

在空闲状态下，利用解码器对第二多媒体数据进行解码处理。

一方面，本申请实施例提供一种解码处理装置，该装置包括：

获取单元，用于在对第一多媒体数据进行硬解码的过程中，获取解码器的运行状态，运行状态包括工作状态或空闲状态；

获取单元，还用于若解码器处于空闲状态，则获取第二多媒体数据，第二多媒体数据是对第一多媒体数据进行复制得到的；

解码单元，用于在空闲状态下，利用解码器对第二多媒体数据进行解码处理。

在一种实施方式中，解码单元还用于：当解码器的运行状态从空闲状态切换至工作状态时，暂停对第二多媒体数据的解码处理。

在一种实施方式中，第一多媒体数据是指多媒体数据流，多媒体数据流包括 N 个数据包，每个数据包中封装有 M 个媒体帧，N、M 均为正整数；解码处理装置还包括处理单元：

该处理单元，用于当解码器接收到多媒体数据流的第 i 个数据包时，对第 i 个数据包进行复制，得到第 i 个数据包的克隆包，第二多媒体数据包括第 i 个数据包的克隆包，i 为正整数且 $i \leq N$ ；

解码单元还用于：在解码器处于工作状态下，对第 i 个数据包中的各个媒体帧进行解码处理，并输出解码处理后的第 i 个数据包中的媒体帧。

在一种可能的实现方式中，第 i 个数据包的克隆包中包含 M 个克隆帧，M 个克隆帧与第 i 个数据包中的 M 个媒体帧一一对应；解码单元具体用于：在空闲状态下，利用解码器

对第 i 个数据包的克隆包中的各个克隆帧进行解码处理。

在一种可能的实现方式中，处理单元还用于：当解码器接收到多媒体数据流的第 $i+1$ 个数据包时，对第 $i+1$ 个数据包进行复制，得到第 $i+1$ 个数据包的克隆包；

处理单元，还用于删除第 i 个数据包的克隆包，则第二多媒体数据包含第 $i+1$ 个数据包的克隆包。

在一种可能的实现方式中，获取单元还用于：在对第一多媒体数据进行硬解码的过程中，获取解码器的性能参数；若解码器的性能参数不满足性能指标要求时，则触发执行获取解码器的运行状态的步骤。

在一种可能的实现方式中，解码器的性能参数包括第一多媒体数据的实际解码时长，性能指标要求包括参考解码帧率；处理单元还用于：

根据实际解码时长确定解码器的实际解码帧率；

若解码器的实际解码帧率小于参考解码帧率，则确定解码器的性能参数不满足性能指标要求。

在一种可能的实现方式中，第一多媒体数据包括多个媒体帧，解码器的性能参数包括第一多媒体数据中的每个媒体帧的实际解码时长，性能指标要求包括参考解码时序范围；处理单元还用于：

根据每个媒体帧的实际解码时长，计算解码器的解码速度时序；

若解码器的解码速度时序超出参考解码时序范围，则确定解码器的性能参数不满足性能指标要求。

在一种可能的实现方式中，参考解码时序范围包括每个媒体帧分别对应的参考解码时长范围；处理单元 603 还用于：

根据同一媒体帧对应的实际解码时长和参考解码时长范围，确定实际解码时长不在参考解码时长范围内的媒体帧所占的目标比例；

若目标比例大于或等于比例阈值，则确定解码器的解码速度时序超出参考解码时序范围。

在一种可能的实现方式中，处理单元 603 还用于：

根据第一多媒体数据中第一媒体帧的实际解码时长，确定每个媒体帧分别对应的参考解码时长范围。

在一种可能的实现方式中，解码器中运行有第一线程、第二线程、第三线程以及第四线程，第一线程、第二线程、第三线程和第四线程之间相互并行运行；

第一线程用于对第一多媒体数据进行硬解码，第二线程用于依次输出解码处理后的第一多媒体数据；

第三线程用于对第一多媒体数据进行复制，得到第二多媒体数据；

第四线程用于获取解码器的运行状态，并在解码器处于空闲状态时获取并解码第二多媒体数据。

在一种可能的实现方式中，第一多媒体数据是指云游戏视频流；处理单元还用于：

获取云游戏的场景需求，场景需求用于指示云游戏视频流的解码模式；

若场景需求指示云游戏视频游戏采用加速解码模式，则在对云游戏视频流进行硬解码的过程中，获取解码器的运行状态。

一方面，本申请实施例提供一种计算机设备，该计算机设备包括存储器和一个或多个处理器，存储器存储有计算机可读指令，计算机可读指令被一个或多个处理器执行时，使得一个或多个处理器执行上述的解码处理方法。

一方面，本申请实施例提供一个或多个非易失性计算机可读存储介质，其上存储有计算机可读指令，该计算机可读指令被一个或多个处理器执行时，使得一个或多个处理器执行上述的解码处理方法。

一方面，本申请实施例提供了一种计算机程序产品或计算机程序，该计算机程序产品或计算机程序包括计算机可读指令，该计算机可读指令存储在计算机可读存储介质中。计算机设备的处理器从计算机可读存储介质读取该计算机可读指令，处理器执行该计算机可读指令，使得该计算机设备执行上述的解码处理方法。

本申请的一个或多个实施例的细节在下面的附图和描述中提出。本申请的其它特征、目的和优点将从说明书、附图以及权利要求书变得明显。

附图说明

为了更清楚地说明本申请实施例中的技术方案，下面将对实施例描述中所需要使用的附图作简单地介绍，显而易见地，下面描述中的附图仅仅是本申请的一些实施例，对于本领域普通技术人员来讲，在不付出创造性劳动的前提下，还可以根据这些附图获得其他的附图。

图1是本申请实施例提供了一种解码处理方法的流程示意图；

图2是本申请实施例提供了一种解码器的解码速度时序的示意图；

图3是本申请实施例提供了一种参考解码时序范围的示意图；

图4是本申请实施例提供的另一种参考解码时序范围的示意图；

图5是本申请实施例提供的另一种解码处理方法的流程示意图；

图6是本申请实施例提供了一种解码处理装置的结构示意图；

图7是本申请实施例提供了一种计算机设备的结构示意图。

具体实施方式

这里将详细地对示例性实施例进行说明，其示例表示在附图中。下面的描述涉及附图时，除非另有表示，不同附图中的相同数字表示相同或相似的要素。以下示例性实施例中所描述的实施方式并不代表与本申请相一致的所有实施方式。相反，它们仅是与如所附权利要求书中所详述的、本申请的一些方面相一致的装置和方法的例子。

本申请实施例提出一种解码处理方案，能够稳定解码器的性能，具体可以使解码器保持恒定的解码帧率，及保持解码速度时序的稳定性；可适用于需要硬件解码加速处理的业务场景，例如适用于超高帧率实时流媒体传输播放场景，以及适用于要求超低延时云游戏解码渲染场景等等。接下来，对该解码处理方案的原理进行相应说明：

首先获取到需要硬解码的第一多媒体数据，该第一多媒体数据可以为视频数据、音频数据或者云游戏数据等等。并且，该第一多媒体数据可以为一段多媒体数据流，也可以为一段多媒体数据流中包括的任一数据包。接下来，可以对获取到的第一多媒体数据进行复制得到第二多媒体数据，可以理解的是，该第一多媒体数据包含多个媒体帧，每个媒体帧均会复制得到相应的克隆帧，该第二多媒体数据则包含这些克隆帧；进一步，在对第一多媒体数据进行硬解码的过程中，可以获取解码器的运行状态，解码器的运行状态可以包括工作状态或者空闲状态；若获取到的解码器的运行状态为空闲状态，则可以在空闲状态下，利用解码器对第二多媒体数据进行解码处理。这样就能保证解码器一直处于恒定的工作频率，从而提升解码器的性能，使得解码器能够输出恒定的解码帧率，及保持解码速度时序的稳定性，进而有效提升解码效率。

在一种可能的实现方式中，本申请的解码处理方案可与区块链技术相结合。例如可以将第一多媒体数据、第二多媒体数据等数据上传至区块链中进行保存，可以保证区块链上的这些数据不易被篡改。基于本申请实施例提供的方案，在对第一多媒体数据进行硬解码的过程中，可以将第一多媒体数据以及第二多媒体数据均发送至区块链网络中并存储至区块链网络中，以使得若解码器处于空闲状态，则获取第二多媒体数据。并且，可以在空闲状态下，利用解码器对第二多媒体数据进行解码处理。

下面对本申请实施例涉及的技术术语进行介绍：

云技术（Cloud technology）基于云计算商业模式应用的网络技术、信息技术、整合技

术、管理平台技术、应用技术等的总称，可以组成资源池，按需所用，灵活便利。云计算技术将变成重要支撑。技术网络系统的后台服务需要大量的计算、存储资源，如视频网站、图片类网站和更多的门户网站。伴随着互联网行业的高度发展和应用，将来每个物品都有可能存在自己的识别标志，都需要传输到后台系统进行逻辑处理，不同程度级别的数据将会分开处理，各类行业数据皆需要强大的系统后盾支撑，只能通过云计算来实现。

目前，云技术主要分为云基础技术类以及云应用类；云基础技术类可以进一步细分为：云计算、云储存、数据库以及大数据等；云应用类可以进一步细分为：医疗云、云物联、云安全、云呼叫、私有云、公有云、混合云、云游戏、云教育、云会议、云社交以及人工智能云服务等。

从基础技术角度来说，本申请的解码处理方法涉及云技术下属的云计算；从应用角度来说，本申请的解码处理方法涉及云技术下属的云游戏。

云计算(cloud computing)指 IT 基础设施的交付和使用模式，指通过网络以按需、易扩展的方式获得所需资源；广义云计算指服务的交付和使用模式，指通过网络以按需、易扩展的方式获得所需服务。这种服务可以是 IT 和软件、互联网相关，也可能是其他服务。云计算是网格计算(Grid Computing)、分布式计算(Distributed Computing)、并行计算(Parallel Computing)、效用计算(Utility Computing)、网络存储(Network Storage Technologies)、虚拟化(Virtualization)、负载均衡(Load Balance)等传统计算机和网络技术发展融合的产物。

云游戏(Cloud gaming)又可称为游戏点播(gaming on demand)，是一种以云计算技术为基础的在线游戏技术。云游戏技术使图形处理与数据运算能力相对有限的轻端设备(thin client)能运行高品质游戏。在云游戏场景下，游戏并不在玩家游戏终端，而是在云端服务器中运行，并由云端服务器将游戏场景渲染为视频音频流，通过网络传输给玩家游戏终端。玩家游戏终端无需拥有强大的图形运算与数据处理能力，仅需拥有基本的流媒体播放能力与获取玩家输入指令并发送给云端服务器的能力即可。

本申请提供的解码处理方案中涉及较大规模计算，需要较大的算力和存储空间，因此在本申请中的一种可行的实施方式中，可以由计算机设备通过云计算技术获取足够算力和存储空间。

进一步地，本申请提供的解码处理方案可以适用于云游戏场景，可以将云游戏场景中的云游戏数据(例如游戏图像数据、游戏音频数据等)作为第一多媒体数据，然后对云游戏数据执行硬解码加速处理，从而可以较大程度的稳定游戏客户端的解码器的解码性能，从而提升用户的游戏体验感。

针对上述所提及的解码处理方案的相关描述，需要说明以下几点：

①上述所提及的解码处理方案可由一个计算机设备执行，该计算机设备可以是终端设备或服务器。其中，终端设备可以是手机、平板电脑、笔记本电脑、掌上电脑、移动互联网设备(MID, mobile internet device)、车辆、车载设备、路边设备、飞行器、智能可穿戴设备(例如智能手表、智能手环、计步器等)、智能电视等具有解码处理功能的设备，等等；服务器可以是独立的物理服务器，也可以是多个物理服务器构成的服务器集群或者分布式系统，还可以是提供云服务、云数据库、云计算、云函数、云存储、网络服务、云通信、中间件服务、域名服务、安全服务、CDN(Content Delivery Network, 内容分发网络)、以及大数据和人工智能平台等基础云计算服务的云服务器，等等。

②上述所提及的计算机设备需要配置有 Intel CPU(Intel Central Processing Unit, 英特尔中央处理器)集成显卡，且在业务场景中使用的 Intel CPU 集成显卡的视频解码模块进行视频硬件解码操作。

③上述所提及的解码处理方案可与区块链技术相结合，进一步地，本申请中利用解码器对第二多媒体数据进行解码处理流程可以在区块链上执行，这样既可以保证解码处理流程的公平公正化，同时可以使得解码处理流程具备可追溯性，从而提升解码处理流程的安

全性。

④上述所提及的的解码处理方案可以应用于云游戏场景中的客户端的解码处理模块，也可运用于任意需要超低延迟硬件视频解码的业务场景。通过本申请提供的解码处理方案的高性能解码模式，提高硬件解码速度，由于在空闲状态下利用解码器对克隆后的多媒体数据进行解码处理，可以保障解码速度时序稳定性，为产品侧的极致用户体验奠定坚实基础。

基于以上分析，下面结合图 1 对本申请的解码处理方法进行描述。请参见图 1，图 1 是本申请实施例提供的一种解码处理方法的流程示意图。该解码处理方法可以由计算机设备执行，计算机设备例如可以是车载设备、智能手机、平板电脑、智能可穿戴设备等等安装有解码器的终端设备。如图 1 所示，该解码处理方法可包括步骤 S110~S130。其中：

S110：在对第一多媒体数据进行硬解码的过程中，获取解码器的运行状态，运行状态包括工作状态或空闲状态。

本申请实施例中，第一多媒体数据可以是一段多媒体数据流中的任一数据包，第一多媒体数据也可以是一个单独的数据。其中，第一多媒体数据可以包括但不限于：视频数据、音频数据或者云游戏数据等。并且，第一多媒体数据是利用编码器编码后的数据，第一多媒体数据的数据格式可以包括但不限于：H.264 格式（一种由国际电信联盟所指定的新一代的视频压缩格式）、MPEG-4 格式（Moving Pictures Experts Group-4，一种国际标准的动态图像格式）等，本申请实施例对第一多媒体数据的数据格式不进行任何限制。

其中，所谓硬解码是指硬件解码，硬件解码就是通过硬件进行视频的解码工作，硬件解码是由图像处理器（Graphics Processing Unit，GPU）来进行的，使用 GPU 解码能够降低 CPU 的工作负荷，降低功耗。需要说明的是，本申请实施例所涉及的解码处理场景主要是硬件解码（以下简称硬解码）场景。因此，本申请实施例中的解码器可以是指计算机设备中的 Intel CPU 集成显卡中的解码模块，解码模块可以包括视频解码模块和音频解码模块等等。

在一种可能的实现方式中，在获取到第一多媒体数据之后，获取解码器的性能参数。然后，若根据解码器的性能参数确定解码器的性能不满足场景需求时，则触发执行获取解码器的运行状态的步骤。另外，若根据解码器的性能参数确定解码器的性能满足场景需求时，则可以通过常规的解码处理方法直接对第一多媒体数据进行解码处理。其中，解码器的性能参数可以包括但不限于：解码时间、解码速度以及解码速度时序稳定性等参数。意思是说，本申请实施例中，在获取到第一多媒体数据之后，可以预先确定解码器的性能是否满足场景需求，若解码器的性能不能满足场景需求，则可以通过执行本申请实施例提供的解码处理方法，从而在特定场景下实现稳定解码器的性能，其中，特定场景包括对解码器性能要求较高的业务场景，例如超高帧率实时流媒体传输播放场景，以及超低延时云游戏解码渲染场景等等。

本申请实施例中，获取到的解码器的运行状态可以包括工作状态或空闲状态。在一种可能的实现方式中，当解码器的运行状态从空闲状态切换至工作状态时，则暂停对第二多媒体数据的解码处理。其中，所谓工作状态是指解码器对接收到的多媒体数据正在执行解码处理的状态；那么，所谓空闲状态可以是指解码器等待接收多媒体数据或未对接收到的多媒体数据执行解码处理的状态；例如，假设解码器的解码帧率为 60 fps (Frames Per Second，即每一秒可以解码 60 帧数据)，即解码器每隔 16ms 接收一个多媒体数据，那么，解码器对接收到的多媒体数据进行解码处理的时间段处于工作状态；而在等待接收每个多媒体数据的时间段（如 16ms）则处于空闲状态。可以理解的是，解码器正在执行的解码处理过程出现被暂停、中断等情况时，解码器则可以由工作状态进入至空闲状态；同理，解码器被暂停、被中断等情况得以解决或恢复时，解码器则可以由空闲状态进入至工作状态。当解

码器的运行状态从空闲状态切换至工作状态时，暂停对第二多媒体数据的解码处理，解码器优先对接收到的多媒体数据进行解码处理，有助于提高解码器的解码效率，保障多媒体数据的有序解码。

在一种可能的实现方式中，第一多媒体数据可以是指多媒体数据流，多媒体数据流可以包括 N 个数据包，每个数据包中封装有 M 个媒体帧， N 、 M 均为正整数。那么，当解码器接收到多媒体数据流的第 i 个数据包时，可以对第 i 个数据包进行复制，得到第 i 个数据包的克隆包。其中，第 i 个数据包可以为多媒体数据流中的任一数据包， i 为正整数且 $i \leq N$ 。并且，第二多媒体数据包括第 i 个数据包的克隆包。此时，在解码器处于工作状态下，对第 i 个数据包中的各个媒体帧进行解码处理，并输出解码处理后的第 i 个数据包中的媒体帧。

具体实现时，假设多媒体数据流为视频数据流，媒体帧可以为图像帧、音频帧、或者云游戏数据帧等等。那么，在解码器需要对一段视频数据流进行硬解码的过程中，在获取到视频数据流中的任一数据包时，可以对该获取到的数据包进行复制，即解码器执行 cloneVideoPkt 操作，得到该数据包的克隆包 (cloned new packet)。那么，第二多媒体数据包括复制得到的 cloned new packet。在解码器处于工作状态下，可以对获取到的该数据包中包括的各个媒体帧依次进行解码处理，然后再依次将解码处理后每个媒体帧逐帧输出。这样，以数据包为单位进行解码处理，以数据包为单元进行复制处理，可以有效保障数据处理的有序性，避免数据混乱。在一种可能的实现方式中，数据包中的各个媒体帧可以携带媒体标识，每个媒体帧的媒体标识用于指示该媒体帧在数据包中的位置和顺序。解码器可以根据各个媒体帧所携带的媒体标识，依次对各个媒体帧进行解码处理，并按照各个媒体帧所携带的媒体标识，依次输出解码处理后的各个媒体帧。其中，媒体标识包括但不限于：字符、数字或者字符串等等，本申请实施例对此不做限定。

S120：若解码器处于空闲状态，则获取第二多媒体数据，第二多媒体数据是对第一多媒体数据进行复制得到的。

在一种可能的实现方式中，第一多媒体数据可以是指多媒体数据流，多媒体数据流可以包括 N 个数据包，每个数据包中封装有 M 个媒体帧。那么，在解码器获取到多媒体数据流中的第 i 个数据包时，可以对第 i 个数据包进行复制，得到第 i 个数据包的克隆包，那么，第二多媒体数据可以包括第 i 个数据包的克隆包。

在另一种可能的实现方式中，当解码器接收到多媒体数据流的第 $i+1$ 个数据包时，对第 $i+1$ 个数据包进行复制，得到第 $i+1$ 个数据包的克隆包。另外，可以删除第 i 个数据包的克隆包，则第二多媒体数据包含第 $i+1$ 个数据包的克隆包。意思是说，在解码器每获取到一个新的数据包时，可以将对该数据包进行复制后得到的克隆包替换掉上一个数据包的克隆包。通过这种方式，可以保证解码器的存储中存储的是最新的一个克隆包，从而减少了存储负担，减轻存储器的缓存压力，进而可以进一步提高解码器的性能。

S130：在空闲状态下，利用解码器对第二多媒体数据进行解码处理。

在一种可能的实现方式中，第二多媒体数据可以包括第 i 个数据包的克隆包，第 i 个数据包的克隆包中包含 M 个克隆帧， M 个克隆帧与第 i 个数据包中的 M 个媒体帧一一对应。在空闲状态下，利用解码器对第二多媒体数据进行解码处理，可以是指：在空闲状态下，利用解码器对第 i 个数据包的克隆包中的各个克隆帧进行解码处理。这样，第 i 个数据包的克隆包中的数据与第 i 个数据包中的数据一一对应，在空闲状态下，解码器无需进行复杂的解码处理，只需便捷地对克隆包中的各个克隆帧进行解码处理，减轻了空闲状态下解码器的解码压力。

本申请实施例中，在对第一多媒体数据进行硬解码的过程中，可以获取解码器的性能参数。若解码器的性能参数不满足性能指标要求时，则触发执行获取解码器的运行状态的步骤。

在一种可能的实现方式中，解码器的性能参数包括第一多媒体数据的实际解码时长，性能指标要求包括参考解码帧率。那么，可以根据第一多媒体数据的实际解码时长确定解码器的实际解码帧率。若解码器的实际解码帧率小于参考解码帧率，则确定解码器的性能参数不满足性能指标要求。

其中，第一多媒体数据的实际解码时长是指第一多媒体数据的真实解码时长。解码帧率是指一秒钟可以解码的媒体帧的数量。实际解码帧率是指解码器的真实解码帧率，参考解码帧率是指解码器的期望解码帧率，可接受的解码帧率。

例如，假设解码器对第一多媒体数据的实际解码时长为 20ms，参考解码帧率为 60fps。其中，解码帧率是指一秒钟可以解码的媒体帧的数量，那么，解码时长和解码帧率之间的关系可以为：解码时长 $t=1000/\text{fps}$ (ms)。那么，根据解码器对第一多媒体数据的实际解码时长 20ms，可以确定得到解码器的实际解码帧率为 $1000/20=50\text{fps}$ ，由于解码器的实际解码帧率 (50fps) 小于参考解码帧率 (60fps)，则可以确定解码器的性能参数不满足性能指标要求。又如，假设解码器对第一多媒体数据的实际解码时长为 10ms，参考解码帧率为 60fps。那么，根据解码器对第一多媒体数据的实际解码时长 10ms，可以确定得到解码器的实际解码帧率为 $1000/10=100\text{fps}$ ，由于解码器的实际解码帧率 (100fps) 大于参考解码帧率 (60fps)，则可以确定解码器的性能参数满足性能指标要求。这样，若解码器的实际解码帧率小于参考解码帧率，确定解码器的性能参数不满足性能指标要求，此时可以触发执行获取解码器的运行状态的步骤执行，使得解码器保持持续稳定的工作频率，从而提高解码器的解码帧率，稳定解码器的性能和提高解码效率。

在另一种可能的实现方式中，第一多媒体数据包括多个媒体帧，解码器的性能参数包括第一多媒体数据中的每个媒体帧的实际解码时长，性能指标要求包括参考解码时序范围。那么，可以根据每个媒体帧的实际解码时长，计算解码器的解码速度时序。若解码器的解码速度时序超出参考解码时序范围，则确定解码器的性能参数不满足性能指标要求。

其中，媒体帧的实际解码时长是指媒体帧的真实解码时长。解码速度时序是由第一多媒体数据中各个媒体帧的实际解码时长构成的。参考解码时序范围是指期望的解码时序范围，解码时序范围是指可接受的解码时长区间。

举例来说，假设第一媒体数据包括 n 个媒体帧， n 为正整数。然后，基于每个媒体帧的实际解码时长，可以得到解码器的解码速度时序。如图 2 所示，图 2 是本申请实施例提供的一种解码器的解码速度时序的示意图。如图 2 所示，解码器的解码速度时序可以由该第一多媒体数据包括的各个媒体帧的实际解码时长构成，如第 1 帧的实际解码时长 $t_1=16\text{ms}$ ，第 2 帧的实际解码时长 $t_2=20\text{ms}$ ，第 3 帧的实际解码时长 $t_3=32\text{ms}$ ，第 4 帧的实际解码时长 $t_4=16\text{ms}$ ，第 5 帧的实际解码时长 $t_5=44\text{ms}$ ，第 6 帧的实际解码时长 $t_6=24\text{ms}$ 。那么，由各个媒体帧的实际解码时长可以构成解码器的解码速度时序。另外，计算机设备可以获取参考解码时序范围。如图 3 所示，图 3 是本申请实施例提供的一种参考解码时序范围的示意图。在一种可能的实现方式中，参考解码时序范围包括各个媒体帧分别对应的参考解码时长范围。可以预先设置各个媒体帧分别对应的参考解码时长范围。各个媒体帧分别对应的参考解码时长范围可以相同可以不同。在一种可能的实现方式中，参考解码时序范围可以根据第一多媒体数据中第 1 帧的实际解码时长确定。根据第一多媒体数据中第一媒体帧的实际解码时长，确定每个媒体帧分别对应的参考解码时长范围。其中，第一媒体帧是指第一多媒体数据中的第 1 帧。例如假设解码器解码第一多媒体数据中的第一媒体帧的实际解码时长为 t_1 ，那么，该第一多媒体数据中所有媒体帧对应的参考解码时长范围可以为： $0.5t_1\sim 3t_1$ 。意思是说，所有媒体帧都对应相同的参考解码时长范围 ($0.5t_1\sim 3t_1$)。这样，根据一个媒体帧的实际解码时长可以快速每个媒体帧分别对应的参考解码时长范围，快速确定参考解码时序范围。例如，剩下 $n-1$ 个媒体帧的实际解码时长均需要满足在该参考解码时长范围内，则解码器的解码速度时序未超出参考解码时序范围，即可确定解码器

的性能参数满足性能指标要求；若剩下 $n-1$ 个媒体帧的实际解码时长中存在一个媒体帧的实际解码时长未满足在该参考解码时长范围内，则解码器的解码速度时序超出参考解码时序范围，即可确定解码器的性能参数不满足性能指标要求。

这样，若解码器的解码速度时序超出参考解码时序范围，确定解码器的性能参数不满足性能指标要求，此时可以触发执行获取解码器的运行状态的步骤执行，使得解码器保持持续稳定的工作频率，从而缩短解码器的解码速度时序，稳定解码器的性能和提高解码效率。

在另一种可能的实现方式中，进一步地，也可以基于比例阈值来判断解码器的解码速度时序是否超出参考解码时序范围。参考解码时序范围包括第一多媒体数据中的每个媒体帧分别对应的参考解码时长范围。根据第一多媒体数据中同一媒体帧对应的实际解码时长和参考解码时长范围，确定实际解码时长不在参考解码时长范围内的媒体帧所占的目标比例；若目标比例大于或等于比例阈值，则确定解码器的解码速度时序超出参考解码时序范围。

在一种可能的实现方式中，实际解码时长不在参考解码时长范围内的媒体帧所占的目标比例，是指针对第一多媒体数据，实际解码时长不在参考解码时长范围内的媒体帧占有所有媒体帧的比例。例如，第一多媒体数据包括 10 个媒体帧，实际解码时长不在参考解码时长范围内的媒体帧的数量为 5，则目标比例为 $5/10=0.5$ 。

在一种可能的实现方式中，若参考解码时序范围是根据第一多媒体数据中第一媒体帧的实际解码时长确定的，实际解码时长不在参考解码时长范围内的媒体帧所占的目标比例，可以是指针对第一多媒体数据，实际解码时长不在参考解码时长范围内的媒体帧占除第一媒体帧之外的所有媒体帧的比例，也可以是指实际解码时长不在参考解码时长范围内的媒体帧占有所有媒体帧的比例。

例如，假设解码器解码第一多媒体数据中的第一媒体帧的实际解码时长为 t_1 ，那么，该第一多媒体数据中所有媒体帧对应的参考解码时长范围可以为： $0.5t_1 \sim 3t_1$ 。可以预先设置一个比例阈值，首先可以根据 $n-1$ 个媒体帧的实际解码时长和参考解码时长范围（ $0.5t_1 \sim 3t_1$ ），确定 $n-1$ 个媒体帧中未能满足在参考解码时长范围内的媒体帧的数量。假设有 k 个， k 为正整数且 $k \leq n-1$ 。然后计算不满足参考时长范围的媒体帧的比例 $k/n-1$ ，根据 $k/n-1$ 和比例阈值，判断解码器的解码速度时序是否超出参考解码时序范围。若 $k/n-1$ 小于比例阈值，则可以确定解码器的性能参数满足性能指标要求；若 $k/n-1$ 大于或者等于比例阈值，则可以确定解码器的性能参数不满足性能指标要求。

其中，比例阈值可以根据场景需求自适应调整，比例阈值也可以根据经验人工设置，本申请实施例对此不做具体限定。

这样，根据目标比例和比例阈值来判断解码器的解码速度时序是否超出参考解码时序范围，可以在第一多媒体数据中大多数媒体帧的实际解码时长均超时的情况下，才确定解码器的解码速度时序超出参考解码时序范围，有效保障判断的准确性。

在另一种可能的实现方式中，参考解码时序范围包括相邻两个媒体帧之间对应的参考解码时长范围。任意相邻两个媒体帧对应的参考解码时长范围可以相同可以不同。若任意相邻两个媒体帧之间对应的参考解码时长范围可以不同，粗略相当于，每个媒体帧对应的参考解码时长范围可以不相同。假设第 1 帧的实际解码时长为 t_1 ，第 2 帧的实际解码时长为 t_2 ，第 3 帧的实际解码时长为 t_3 ，... ..，第 n 帧的实际解码时长为 t_n 。假设 t_1 与 t_2 之间的参考解码时长范围（绝对值）可以为： $0 \leq t_2 - t_1 \leq \max_1$ ， t_2 与 t_3 之间的参考解码时长范围可以为： $0 \leq t_3 - t_2 \leq \max_2$ ， t_3 与 t_4 之间的参考解码时长范围可以为： $0 \leq t_4 - t_3 \leq \max_3$ ，... ..， t_{n-1} 与 t_n 之间的参考解码时长范围可以为： $0 \leq t_n - t_{n-1} \leq \max_{n-1}$ 。其中， $\max_1$ 、 $\max_2$ 、 $\max_3$ 、... ..、 $\max_{n-1}$ 之间可以互不相同，也可以存在至少两个相同的数据。

例如，请参见图 4，图 4 是本申请实施例提供的另一种参考解码时序范围的示意图。

首先根据图 2 所示的解码器的解码速度时序图,可以确定第一帧与第二帧之间的解码时长差值为: $20\text{ms}-16\text{ms}=4\text{ms}$, 第二帧与第三帧之间的解码时长差值为: $32\text{ms}-20\text{ms}=12\text{ms}$, 以此类推, 依次得到解码时长差值 16ms , 28ms , 20ms 等等。然后, 假设两个相邻的媒体帧之间的解码时长范围如图 4 所示, 即第一帧与第二帧之间的参考解码时长范围为: $0\sim 10\text{ms}$, 第二帧与第三帧之间的参考解码时长范围为: $0\sim 15\text{ms}$, 以此类推, 第五帧与第六帧之间的参考解码时长范围为: $0\sim 30\text{ms}$ 。其中, 一个参考解码时长范围对应一个解码时长差值 (即任意相邻两个媒体帧的实际解码时长之间的差值)。然后, 基于前述所确定的任意相邻两个媒体帧之间的解码时长差值, 依次判断任意相邻两个媒体帧之间的解码时长差值是否满足对应的参考解码时长范围。例如, 第一帧与第二帧之间的解码时长差值 4ms , 满足对应的参考解码时长范围 ($0\sim 10\text{ms}$), 第二帧与第三帧之间的解码时长差值 12ms , 也满足对应的参考解码时长范围 ($0\sim 15\text{ms}$)。又如, 第四帧与第五帧之间的解码时长差值 28ms , 不满足对应的参考解码时长范围 ($0\sim 25\text{ms}$)。最终, 若任意相邻两个媒体帧之间的解码时长差值, 均满足对应的参考解码时长范围, 则可以确定解码器的性能参数满足性能指标要求; 若存在相邻两个媒体帧之间的解码时长差值, 未满足对应的参考解码时长范围, 则确定解码器的解码速度时序超出参考解码时序范围, 即确定解码器的性能参数不满足性能指标要求。

通过上述方式, 可以预先根据解码器的性能参数, 确定解码器是否满足性能指标要求, 当然, 在不同的业务场景下, 其性能指标要求可以不相同。在解码器不满足性能指标要求的情况下, 即可通过本申请实施例提供的解码处理方法, 来优化解码器的性能, 并提高解码器的工作效率。

综上所述, 本申请实施例可以应用于云游戏场景。即第一多媒体数据是指云游戏视频流, 那么, 可以获取云游戏的场景需求, 其中, 场景需求用于指示云游戏视频流的解码模式。若场景需求指示云游戏视频游戏采用加速解码模式, 则在对云游戏视频流进行硬解码的过程中, 获取解码器的运行状态。

在一种可能的实现方式中, 游戏开发者在开发云游戏时, 可以预先设置该游戏的游戏模式为加速解码模式, 那么, 在该云游戏场景中, 即可采用本申请实施例提供的解码处理方案对云游戏视频流进行硬解码的加速处理, 从而可以较大程度的稳定游戏客户端的解码器的解码性能, 从而提升用户的游戏体验感。

在另一种可能的实现方式中, 云游戏的场景需求也可以根据运行该游戏的游戏客户端的网络环境确定。若用户 A 所在的游戏客户端的网络环境为弱网环境, 则可以确定云游戏的场景需求指示云游戏视频流需要采用加速解码模式; 若用户 B 所在的游戏客户端的网络环境为强网环境, 则可以确定云游戏的场景需求指示云游戏视频流可以不需要采用加速解码模式。

在超高帧率实时流媒体传输播放场景, 以及超低延时云游戏解码渲染等场景中, 本申请实施例可以解决硬件解码超低延迟要求场景中可能出现的解码时间偏长和时序稳定性差的问题, 用以保障流畅稳定的视频解码性能, 给云游戏等超低延时要求的业务带来极致的用户体验。

本申请实施例中, 在对第一多媒体数据进行硬解码的过程中, 可以获取解码器的运行状态, 运行状态可以包括空闲状态或者工作状态; 若解码器的运行状态为空闲状态, 则可以获取对第一多媒体数据进行复制后得到的第二多媒体数据, 并在空闲状态下, 利用解码器对第二多媒体数据进行解码处理。本申请可以在解码器对第一多媒体数据进行硬解码处理的过程中, 利用解码器的空闲状态解码克隆的第一多媒体数据, 这样使得解码器保持持续稳定的工作频率, 从而稳定解码器的性能和提高解码效率。

请参见图 5, 图 5 是本申请实施例提供的另一种解码处理方法的流程示意图。如图 5

所示,该解码处理方法由解码器执行,解码器中运行有第一线程、第二线程、第三线程以及第四线程,并且第一线程、第二线程、第三线程和第四线程之间相互并行运行。需要说明的是,结合图1实施例所述的解码处理方法,图5所示的第一线程(线程1)用于:对第一多媒体数据进行硬解码,第二线程(线程2)用于:依次输出解码处理后的多媒体数据,第三线程(线程3)用于:对第一多媒体数据进行复制,得到第二多媒体数据,第四线程(线程4)用于:获取解码器的运行状态,并在解码器处于空闲状态时获取并解码第二多媒体数据。

本申请实施例中,以解码器对多媒体数据流进行硬解码处理为例进行详细说明。假设多媒体数据流包括多个数据包。接下来,结合图5对本申请实施例提供的解码处理方法进行进一步地详细描述,图5所示的解码处理方法包括步骤S501~S509:

S501:等待一个新的数据包。

在一种可能的实现方式中,多媒体数据流是基于流式传输的,所谓流式传输是指多媒体数据流中的每个数据包是一个接着一个传输的,即解码器每当获取到多媒体数据流中的一个数据包,对该数据包完成解码处理后,再等待获取下一个数据包,再进行对下一个数据包的解码处理,以此类推。其中,可以根据解码器的解码帧率确定解码器需要等待一个新的数据包的时长,解码时长 $t=1000/\text{fps}$ (ms)。例如,解码器的解码帧率为60fps,则解码器在获取到第一个数据包之后,需要等待 $t=1000/60=16\text{ms}$ 才可以接收到下一个数据包。

S502:获取第一多媒体数据。

本申请实施例中,第一多媒体数据可以为多媒体数据流包括的N个数据包中的任意一个数据包,每个数据包封装有M个媒体帧,N、M均为正整数。即解码器中运行的线程1所获取到的第一多媒体数据封装有M个媒体帧。其中,媒体帧可以包括但不限于:图像帧、音频帧、云游戏视频帧等等。

S503:复制操作。

在一种可能的实现方式中,解码器每获取到一个新的数据包之后,即可同时启动线程3对该数据包进行cloneVideoPkt操作(复制操作),从而得到cloned new packet(第二多媒体数据)。

S504:判断是否为空闲状态。

在一种可能的实现方式中,线程4获取解码器的运行状态(包括空闲状态或工作状态),并判断解码器的运行状态是否为空闲状态。若解码器的运行状态为空闲状态,则触发执行线程4获取第二多媒体数据的操作。其中,所谓空闲状态可以是指解码器是否处于等待接收一个新的数据包的状态,若是,则可以确定解码器的运行状态为空闲状态。

S505:获取第二多媒体数据。

在一种可能的实现方式中,若解码器的运行状态为空闲状态,则线程4可以获取第二多媒体数据。其中,第二多媒体数据也可以包括任一数据包的克隆包。例如,线程4所获取到的第二多媒体数据可以是对线程1当前获取到的数据包进行克隆后得到的克隆包,第二多媒体数据也可以是对线程1获取到的上一个数据包进行克隆后得到的克隆包。并且,可以将最新数据包的克隆包替换掉上一个数据包的克隆包,以此来减小解码器的存储负担。

S506:解码第二多媒体数据。

在一种可能的实现方式中,在空闲状态下,线程4可以执行decodeClonePacket,即解码器中的运行的线程4对第二多媒体数据进行解码处理。其中,第二多媒体数据包括M个克隆帧,M个克隆帧与第一多媒体数据中的M个媒体帧一一对应。那么,解码器对第二多媒体数据进行解码处理可以是指解码器对M个克隆帧依次进行解码处理。

在一种可能的实现方式中,若解码器的运行状态从空闲状态切换为工作状态,则线程4暂停对第二多媒体数据的解码处理。

S507:解码第一多媒体数据。

在一种可能的实现方式中，线程 1 获取到第一多媒体数据之后，可以对第一多媒体数据进行解码处理。其中，解码器执行 `decodeOneFrame` 操作，即解码器对第一多媒体数据进行解码处理可以是指解码器对 M 个媒体帧依次进行解码处理。

S508：解码回传。

在一种可能的实现方式中，所谓解码回传（`decode callback`）可以是指：线程 2 依次输出解码处理后的第一多媒体数据包括的 M 个媒体帧。

S509：判断是否读取结束。

在一种可能的实现方式中，在每完成对一个数据包的解码处理之后，线程 1 可以判断当前多媒体数据流是否读取结束，若是，则确定完成对多媒体数据流的解码处理；若否，则继续等待接收下一个新的数据包，并重复执行上述步骤，直至多媒体数据流读取结束。其中，若检测到当前数据包携带有结束标识符，则可以确定多媒体数据流读取结束。

需要说明的是，上述步骤 S501-S509 中，步骤标号并不对这些步骤之间的执行顺序进行限定，例如，步骤 S502 和步骤 S503 可以是同时执行。又如，步骤 S506 可以是先于步骤 S507 执行，步骤 S506 也可以是后于步骤 S507 执行。

本申请实施例中，在对第一多媒体数据进行硬解码的过程中，可以获取解码器的运行状态，运行状态可以包括空闲状态或者工作状态；若解码器的运行状态为空闲状态，则可以获取对第一多媒体数据进行复制后得到的第二多媒体数据，并在空闲状态下，利用解码器对第二多媒体数据进行解码处理。本申请可以在解码器对第一多媒体数据进行硬解码处理的过程中，利用解码器的空闲状态解码克隆的第一多媒体数据，这样使得解码器保持持续稳定的工作频率，从而稳定解码器的性能和提高解码效率。

应该理解的是，虽然上述各实施例的流程图中的各个步骤按照箭头的指示依次显示，但是这些步骤并不是必然按照箭头指示的顺序依次执行。除非本文中有明确的说明，这些步骤的执行并没有严格的顺序限制，这些步骤可以以其它的顺序执行。而且，上述各实施例的至少一部分步骤可以包括多个步骤或者多个阶段，这些步骤或者阶段并不必然是在同一时刻执行完成，而是可以在不同的时刻执行，这些步骤或者阶段的执行顺序也不必然是依次进行，而是可以与其它步骤或者其它步骤中的步骤或者阶段的至少一部分轮流或者交替地执行。

请参见图 6，图 6 是本申请实施例提供的一种解码处理装置的结构示意图。该解码处理装置 600 可应用于图 1 和图 5 对应的方法实施例中的计算机设备。解码处理装置 600 可以是运行于计算机设备中的一个计算机程序（包括程序代码），例如该解码处理装置 600 为一个应用软件；该装置可以用于执行本申请实施例提供的方法中的相应步骤。该解码处理装置 600 可以包括：

获取单元 601，用于在对第一多媒体数据进行硬解码的过程中，获取解码器的运行状态，运行状态包括工作状态或空闲状态；

获取单元 601，还用于若解码器处于空闲状态，则获取第二多媒体数据，第二多媒体数据是对第一多媒体数据进行复制得到的；

解码单元 602，用于在空闲状态下，利用解码器对第二多媒体数据进行解码处理。

在一种可能的实现方式中，解码单元 602 还用于：当解码器的运行状态从空闲状态切换至工作状态时，暂停对第二多媒体数据的解码处理。

在一种可能的实现方式中，第一多媒体数据是指多媒体数据流，多媒体数据流包括 N 个数据包，每个数据包中封装有 M 个媒体帧，N、M 均为正整数；解码处理装置 600 还可以包括处理单元 603：

该处理单元 603，用于当解码器接收到多媒体数据流的第 i 个数据包时，对第 i 个数据包进行复制，得到第 i 个数据包的克隆包，第二多媒体数据包括第 i 个数据包的克隆包，

i 为正整数且 $i \leq N$;

解码单元 602 还用于: 在解码器处于工作状态下, 对第 i 个数据包中的各个媒体帧进行解码处理, 并输出解码处理后的第 i 个数据包中的媒体帧。

在一种可能的实现方式中, 第 i 个数据包的克隆包中包含 M 个克隆帧, M 个克隆帧与第 i 个数据包中的 M 个媒体帧一一对应; 解码单元 602 具体用于: 在空闲状态下, 利用解码器对第 i 个数据包的克隆包中的各个克隆帧进行解码处理。

在一种可能的实现方式中, 处理单元 603 还用于: 当解码器接收到多媒体数据流的第 $i+1$ 个数据包时, 对第 $i+1$ 个数据包进行复制, 得到第 $i+1$ 个数据包的克隆包;

处理单元 603, 还用于删除第 i 个数据包的克隆包, 则第二多媒体数据包含第 $i+1$ 个数据包的克隆包。

在一种可能的实现方式中, 获取单元 601 还用于: 在对第一多媒体数据进行硬解码的过程中, 获取解码器的性能参数; 若解码器的性能参数不满足性能指标要求时, 则触发执行获取解码器的运行状态的步骤。

在一种可能的实现方式中, 解码器的性能参数包括第一多媒体数据的实际解码时长, 性能指标要求包括参考解码帧率; 处理单元 603 还用于:

根据实际解码时长确定解码器的实际解码帧率;

若解码器的实际解码帧率小于参考解码帧率, 则确定解码器的性能参数不满足性能指标要求。

在一种可能的实现方式中, 第一多媒体数据包括多个媒体帧, 解码器的性能参数包括第一多媒体数据中的每个媒体帧的实际解码时长, 性能指标要求包括参考解码时序范围; 处理单元 603 还用于:

根据每个媒体帧的实际解码时长, 计算解码器的解码速度时序;

若解码器的解码速度时序超出参考解码时序范围, 则确定解码器的性能参数不满足性能指标要求。

在一种可能的实现方式中, 参考解码时序范围包括每个媒体帧分别对应的参考解码时长范围; 处理单元 603 还用于:

根据同一媒体帧对应的实际解码时长和参考解码时长范围, 确定实际解码时长不在参考解码时长范围内的媒体帧所占的目标比例;

若目标比例大于或等于比例阈值, 则确定解码器的解码速度时序超出参考解码时序范围。

在一种可能的实现方式中, 处理单元 603 还用于:

根据第一多媒体数据中第一媒体帧的实际解码时长, 确定每个媒体帧分别对应的参考解码时长范围。

在一种可能的实现方式中, 解码器中运行有第一线程、第二线程、第三线程以及第四线程, 第一线程、第二线程、第三线程和第四线程之间相互并行运行;

第一线程用于对第一多媒体数据进行硬解码, 第二线程用于依次输出解码处理后的第一多媒体数据;

第三线程用于对第一多媒体数据进行复制, 得到第二多媒体数据;

第四线程用于获取解码器的运行状态, 并在解码器处于空闲状态时获取并解码第二多媒体数据。

在一种可能的实现方式中, 第一多媒体数据是指云游戏视频流; 处理单元 603 还用于: 获取云游戏的场景需求, 场景需求用于指示云游戏视频流的解码模式;

若场景需求指示云游戏视频游戏采用加速解码模式, 则在对云游戏视频流进行硬解码的过程中, 获取解码器的运行状态。

本申请实施例中, 在对第一多媒体数据进行硬解码的过程中, 可以获取解码器的运行

状态，运行状态可以包括空闲状态或者工作状态；若解码器的运行状态为空闲状态，则可以获取对第一多媒体数据进行复制后得到的第二多媒体数据，并在空闲状态下，利用解码器对第二多媒体数据进行解码处理。本申请可以在解码器对第一多媒体数据进行硬解码处理的过程中，利用解码器的空闲状态解码克隆的第一多媒体数据，这样使得解码器保持持续稳定的工作频率，从而稳定解码器的性能和提高解码效率。

请参见图 7，图 7 是本申请实施例提供的一种计算机设备的结构示意图。该计算机设备 700 用于执行图 1 和图 5 对应的方法实施例中计算机设备所执行的步骤，该计算机设备 700 包括：一个或多个处理器 710；一个或多个输入设备 720，一个或多个输出设备 730 和存储器 740。上述处理器 710、输入设备 720、输出设备 730 和存储器 740 通过总线 750 连接。存储器 740 用于存储计算机程序，所述计算机程序包括计算机可读指令，处理器 710 用于调用存储器 740 存储的计算机可读指令，执行以下操作：

在对第一多媒体数据进行硬解码的过程中，获取解码器的运行状态，运行状态包括工作状态或空闲状态；

若解码器处于空闲状态，则获取第二多媒体数据，第二多媒体数据是对第一多媒体数据进行复制得到的；

在空闲状态下，利用解码器对第二多媒体数据进行解码处理。

在一种可能的实现方式中，当解码器的运行状态从空闲状态切换至工作状态时，则处理器 710 暂停对第二多媒体数据的解码处理。

在一种可能的实现方式中，第一多媒体数据是指多媒体数据流，多媒体数据流包括 N 个数据包，每个数据包中封装有 M 个媒体帧，N、M 均为正整数；处理器 710 还用于执行以下操作：

当解码器接收到多媒体数据流的第 i 个数据包时，对第 i 个数据包进行复制，得到第 i 个数据包的克隆包，第二多媒体数据包括第 i 个数据包的克隆包，i 为正整数且 $i \leq N$ ；

在解码器处于工作状态下，对第 i 个数据包中的各个媒体帧进行解码处理，并输出解码处理后的第 i 个数据包中的媒体帧。

在一种可能的实现方式中，第 i 个数据包的克隆包中包含 M 个克隆帧，M 个克隆帧与第 i 个数据包中的 M 个媒体帧一一对应；在空闲状态下，处理器 710 利用解码器对第二多媒体数据进行解码处理，包括：

在空闲状态下，利用解码器对第 i 个数据包的克隆包中的各个克隆帧进行解码处理。

在一种可能的实现方式中，当解码器接收到多媒体数据流的第 i+1 个数据包时，处理器 710 对第 i+1 个数据包进行复制，得到第 i+1 个数据包的克隆包；

处理器 710 删除第 i 个数据包的克隆包，则第二多媒体数据包含第 i+1 个数据包的克隆包。

在一种可能的实现方式中，在处理器 710 对第一多媒体数据进行硬解码的过程中，获取解码器的性能参数；

若解码器的性能参数不满足性能指标要求时，则触发执行获取解码器的运行状态的步骤。

在一种可能的实现方式中，解码器的性能参数包括第一多媒体数据的实际解码时长，性能指标要求包括参考解码帧率；处理器 710 还用于执行以下操作：

根据实际解码时长确定解码器的实际解码帧率；

若解码器的实际解码帧率小于参考解码帧率，则确定解码器的性能参数不满足性能指标要求。

在一种可能的实现方式中，第一多媒体数据包括多个媒体帧，解码器的性能参数包括第一多媒体数据中的每个媒体帧的实际解码时长，性能指标要求包括参考解码时序范围；

处理器 710 还用于执行以下操作：

根据每个媒体帧的实际解码时长，计算解码器的解码速度时序；

若解码器的解码速度时序超出参考解码时序范围，则确定解码器的性能参数不满足性能指标要求。

在一种可能的实现方式中，参考解码时序范围包括每个媒体帧分别对应的参考解码时长范围；处理器 710 还用于执行以下操作：

根据同一媒体帧对应的实际解码时长和参考解码时长范围，确定实际解码时长不在参考解码时长范围内的媒体帧所占的目标比例；

若目标比例大于或等于比例阈值，则确定解码器的解码速度时序超出参考解码时序范围。

在一种可能的实现方式中，处理器 710 还用于执行以下操作：

根据第一多媒体数据中第一媒体帧的实际解码时长，确定每个媒体帧分别对应的参考解码时长范围。

在一种可能的实现方式中，解码器中运行有第一线程、第二线程、第三线程以及第四线程，第一线程、第二线程、第三线程和第四线程之间相互并行运行；

第一线程用于对第一多媒体数据进行硬解码，第二线程用于依次输出解码处理后的第一多媒体数据；

第三线程用于对第一多媒体数据进行复制，得到第二多媒体数据；

第四线程用于获取解码器的运行状态，并在解码器处于空闲状态时获取并解码第二多媒体数据。

在一种可能的实现方式中，第一多媒体数据是指云游戏视频流；处理器 710 还用于执行以下操作：

获取云游戏的场景需求，场景需求用于指示云游戏视频流的解码模式；

若场景需求指示云游戏视频游戏采用加速解码模式，则在对云游戏视频流进行硬解码的过程中，获取解码器的运行状态。

本申请实施例中，在对第一多媒体数据进行硬解码的过程中，可以获取解码器的运行状态，运行状态可以包括空闲状态或者工作状态；若解码器的运行状态为空闲状态，则可以获取对第一多媒体数据进行复制后得到的第二多媒体数据，并在空闲状态下，利用解码器对第二多媒体数据进行解码处理。本申请可以在解码器对第一多媒体数据进行硬解码处理的过程中，利用解码器的空闲状态解码克隆的第一多媒体数据，这样使得解码器保持持续稳定的工作频率，从而稳定解码器的性能和提高解码效率。

应当理解，本申请实施例中所描述的计算机设备 700 可执行前文图 1 和图 5 所对应实施例中对解码处理方法的描述，也可执行前文图 6 所对应实施例中对解码处理装置 600 的描述，在此不再赘述。

此外，这里需要指出的是：本申请实施例还提供了一种计算机存储介质，且计算机存储介质中存储有计算机程序，且该计算机程序包括计算机可读指令，当处理器执行上述程序指令时，能够执行前文图 1 和图 5 所对应实施例中的方法，因此，这里将不再进行赘述。对于本申请所涉及的计算机存储介质实施例中未披露的技术细节，请参照本申请方法实施例的描述。作为示例，程序指令可以被部署在一个计算机设备上，或者在位于一个地点的多个计算机设备上执行，又或者，在分布在多个地点且通过通信网络互连的多个计算机设备上执行。

根据本申请的一个方面，提供了一种计算机程序产品或计算机程序，该计算机程序产品或计算机程序包括计算机可读指令，该计算机可读指令存储在计算机可读存储介质中。计算机设备的处理器从计算机可读存储介质读取该计算机可读指令，处理器执行该计算机可读指令，使得该计算机设备可以执行前文图 1 和图 5 所对应实施例中的方法，因此，这

里将不再进行赘述。

本领域普通技术人员可以理解实现上述实施例方法中的全部或部分流程，是可以通过计算机可读指令来指令相关的硬件来完成，上述程序可存储于一计算机可读取存储介质中，该程序在执行时，可包括如上述各方法的实施例的流程。其中，上述存储介质可为磁碟、光盘、只读存储记忆体（Read-Only Memory，ROM）或随机存储记忆体（Random Access Memory，RAM）等。

以上所揭露的仅为本申请较佳实施例而已，当然不能以此来限定本申请之权利范围，因此依本申请权利要求所作的等同变化，仍属本申请所涵盖的范围。

权利要求书

1、一种解码处理方法，由计算机设备执行，包括：

在对第一多媒体数据进行硬解码的过程中，获取解码器的运行状态，所述运行状态包括工作状态或空闲状态；

若所述解码器处于空闲状态，则获取第二多媒体数据，所述第二多媒体数据是对所述第一多媒体数据进行复制得到的；及

在所述空闲状态下，利用所述解码器对所述第二多媒体数据进行解码处理。

2、如权利要求1所述的方法，所述方法还包括：

当所述解码器的运行状态从所述空闲状态切换至工作状态时，则暂停对所述第二多媒体数据的解码处理。

3、如权利要求1所述的方法，所述第一多媒体数据是指多媒体数据流，所述多媒体数据流包括N个数据包，每个数据包中封装有M个媒体帧，N、M均为正整数；所述方法还包括：

当所述解码器接收到所述多媒体数据流的第i个数据包时，对所述第i个数据包进行复制，得到所述第i个数据包的克隆包，所述第二多媒体数据包括所述第i个数据包的克隆包，i为正整数且 $i \leq N$ ；及

在所述解码器处于工作状态下，对所述第i个数据包中的各个媒体帧进行解码处理，并输出解码处理后的所述第i个数据包中的媒体帧。

4、如权利要求3所述的方法，所述第i个数据包的克隆包中包含M个克隆帧，所述M个克隆帧与所述第i个数据包中的M个媒体帧一一对应；所述在所述空闲状态下，利用所述解码器对所述第二多媒体数据进行解码处理，包括：

在所述空闲状态下，利用所述解码器对所述第i个数据包的克隆包中的各个克隆帧进行解码处理。

5、如权利要求3所述的方法，所述方法还包括：

当所述解码器接收到所述多媒体数据流的第i+1个数据包时，对所述第i+1个数据包进行复制，得到所述第i+1个数据包的克隆包；及

删除所述第i个数据包的克隆包，则所述第二多媒体数据包含所述第i+1个数据包的克隆包。

6、如权利要求1所述的方法，所述方法还包括：

在对所述第一多媒体数据进行硬解码的过程中，获取所述解码器的性能参数；及

若所述解码器的性能参数不满足性能指标要求时，则触发执行所述获取解码器的运行状态的步骤。

7、如权利要求6所述的方法，所述解码器的性能参数包括所述第一多媒体数据的实际解码时长，所述性能指标要求包括参考解码帧率；所述方法还包括：

根据所述实际解码时长确定所述解码器的实际解码帧率；及

若所述解码器的实际解码帧率小于所述参考解码帧率，则确定所述解码器的性能参数不满足所述性能指标要求。

8、如权利要求6所述的方法，所述第一多媒体数据包括多个媒体帧，所述解码器的

性能参数包括所述第一多媒体数据中的每个媒体帧的实际解码时长，所述性能指标要求包括参考解码时序范围；所述方法还包括：

根据所述每个媒体帧的实际解码时长，计算所述解码器的解码速度时序；及

若所述解码器的解码速度时序超出所述参考解码时序范围，则确定所述解码器的性能参数不满足所述性能指标要求。

9、如权利要求 8 所述的方法，所述参考解码时序范围包括所述每个媒体帧分别对应的参考解码时长范围，所述方法还包括：

根据同一媒体帧对应的实际解码时长和参考解码时长范围，确定实际解码时长不在参考解码时长范围内的媒体帧所占的目标比例；及

若所述目标比例大于或等于比例阈值，则确定所述解码器的解码速度时序超出所述参考解码时序范围。

10、如权利要求 9 所述的方法，所述方法还包括：

根据所述第一多媒体数据中第一媒体帧的实际解码时长，确定所述每个媒体帧分别对应的参考解码时长范围。

11、如权利要求 1 所述的方法，所述解码器中运行有第一线程、第二线程、第三线程以及第四线程，所述第一线程、第二线程、第三线程和第四线程之间相互并行运行；

所述第一线程用于对所述第一多媒体数据进行硬解码，所述第二线程用于依次输出解码处理后的所述第一多媒体数据；

所述第三线程用于对第一多媒体数据进行复制，得到所述第二多媒体数据；及

所述第四线程用于获取解码器的运行状态，并在所述解码器处于空闲状态时获取并解码所述第二多媒体数据。

12、如权利要求 1 至 11 中任一项所述的方法，所述第一多媒体数据是指云游戏视频流；所述方法还包括：

获取云游戏的场景需求，所述场景需求用于指示所述云游戏视频流的解码模式；及

若所述场景需求指示所述云游戏视频流采用加速解码模式，则在对云游戏视频流进行硬解码的过程中，获取所述解码器的运行状态。

13、一种解码处理装置，包括：

获取单元，用于在对第一多媒体数据进行硬解码的过程中，获取解码器的运行状态，所述运行状态包括工作状态或空闲状态；

所述获取单元，还用于若所述解码器处于空闲状态，则获取第二多媒体数据，所述第二多媒体数据是对所述第一多媒体数据进行复制得到的；及

解码单元，用于在所述空闲状态下，利用所述解码器对所述第二多媒体数据进行解码处理。

14、一种计算机设备，包括存储器和一个或多个处理器，所述存储器存储有计算机可读指令，所述一个或多个处理器执行所述计算机可读指令时实现如权利要求 1-12 中任一项所述的解码处理方法。

15、一个或多个非易失性计算机可读存储介质，存储有计算机可读指令，所述计算机可读指令被一个或多个处理器执行时实现如权利要求 1-12 中任一项所述的解码处理方法。

16、一种计算机程序产品，包括计算机可读指令，所述计算机可读指令被一个或多个处理器执行时实现权利要求 1-12 中任一项所述的方法的步骤。