

DECODING PROCESSING METHOD AND APPARATUS, COMPUTER DEVICE, AND STORAGE MEDIUM

RELATED APPLICATION

[0001] This application claims priority to Chinese Patent Application No. 202110965320.0, entitled "DECODING PROCESSING METHOD AND APPARATUS, COMPUTER DEVICE, AND STORAGE MEDIUM" filed with the Chinese Patent Office on August 20, 2021, which is incorporated by reference in its entirety.

FIELD OF THE TECHNOLOGY

[0002] This disclosure relates to the field of computer technologies, and in particular, to a decoding processing method, a decoding processing apparatus, a computer device, a computer-readable storage medium, and a computer program product.

BACKGROUND OF THE DISCLOSURE

[0003] In the field of multimedia services, it is necessary to perform decoding processing on the encoded multimedia data, and then perform page rendering on the multimedia data subjected to the decoding processing for users to view. However, in a process of performing decoding processing on the multimedia data, if the performance of a decoder is insufficient, the performance of the decoding process of the multimedia data in terms of frequency is not stable enough, resulting in video stuttering and affecting the video viewing effect.

[0004] In view of the above situation, a common practice is to call an application program interface (API) of a hardware decoder to use hardware to accelerate the video decoding process. However, after using the hardware for accelerated decoding processing, under the high requirements of ultra low latency application scenarios, inherent performance problems (e.g. the decoding speed is not fast enough and the stability is not high enough, etc.) of the decoder may occur, and these problems may reduce the efficiency of the decoding process by the hardware. Therefore, how to stabilize the performance of the decoder is a technical problem that needs to be urgently solved at present.

SUMMARY

[0005] Embodiments of this disclosure provide a decoding processing method and apparatus, a computer device, a storage medium, and a computer program product.

[0006] According to one aspect, the embodiments of this disclosure provide a decoding processing method, performed by a computer device. The method includes:

[0007] acquiring a running state of a decoder in a process of performing hardware decoding on first multimedia data, the running state including a working state or an idle state;

[0008] acquiring second multimedia data when the decoder is in an idle state, the second multimedia data being a copy of the first multimedia data; and

[0009] using the decoder to perform decoding processing on the second multimedia data in the idle state.

[0010] According to one aspect, the embodiments of this disclosure provide a decoding processing apparatus. The apparatus includes:

[0011] an acquisition unit, configured to acquire a running state of a decoder in a process of performing hardware decoding on first multimedia data, the running state including a working state or an idle state;

[0012] the acquisition unit being further configured to acquire second multimedia data when the decoder is in an idle state, and the second multimedia data being a copy of the first multimedia data; and

[0013] a decoding unit, configured to use the decoder to perform decoding processing on the second multimedia data in the idle state.

[0014] In one implementation, the decoding unit is further configured to: pause decoding processing of the second multimedia data when the running state of the decoder switches from the idle state to a working state.

[0015] In one implementation, the first multimedia data refers to a multimedia data stream. The multimedia data stream includes N data packets, and M media frames are encapsulated in each data packet, N and M being both positive integers. The decoding processing apparatus further includes a processing unit.

[0016] The processing unit is configured to copy an i-th data packet of the multimedia data stream when the decoder receives the i-th data packet to obtain a cloned packet of the i-th data packet, the second multimedia data including the cloned packet of the i-th data packet, i being a positive integer, and i being less than or equal to N.

[0017] The decoding unit is further configured to: perform decoding processing on media frames in the i -th data packet when the decoder is in a working state, and output the media frames subjected to the decoding processing in the i -th data packet.

[0018] In one possible implementation, the cloned packet of the i -th data packet includes M cloned frames, and the M cloned frames have one-to-one correspondence to the M media frames in the i -th data packet. The decoding unit is specifically configured to: use the decoder to perform decoding processing on the cloned frames in the cloned packet of the i -th data packet in the idle state.

[0019] In one possible implementation, the processing unit is further configured to: copy an $(i+1)$ -th data packet of the multimedia data stream when the decoder receives the $(i+1)$ -th data packet to obtain a cloned packet of the $(i+1)$ -th data packet.

[0020] The processing unit is further configured to delete the cloned packet of the i -th data packet, so that the second multimedia data includes the cloned packet of the $(i+1)$ -th data packet.

[0021] In one possible implementation, the acquisition unit is further configured to: acquire a performance parameter of the decoder in the process of performing hardware decoding on the first multimedia data; and trigger to perform the step of acquiring a running state of a decoder when the performance parameter of the decoder does not meet a performance indicator requirement.

[0022] In one possible implementation, the performance parameter of the decoder includes an actual decoding duration of the first multimedia data, and the performance indicator requirement includes a reference decoding frame rate. The processing unit is further configured to:

[0023] determine an actual decoding frame rate of the decoder according to the actual decoding duration; and

[0024] determine that the performance parameter of the decoder does not meet the performance indicator requirement when the actual decoding frame rate of the decoder is less than the reference decoding frame rate.

[0025] In one possible implementation, the first multimedia data includes a plurality of media frames, the performance parameter of the decoder includes an actual decoding duration of each media frame in the first multimedia data, and the performance indicator requirement includes a reference decoding time series range. The processing unit is further configured to:

[0026] calculate a decoding speed time series of the decoder according to the actual decoding duration of each media frame; and

[0027] determine that the performance parameter of the decoder does not meet the performance indicator requirement when the decoding speed time series of the decoder exceeds the reference decoding time series range.

[0028] In one possible implementation, the reference decoding time series range includes reference decoding duration ranges respectively corresponding to the media frames. The processing unit 603 is further configured to:

[0029] determine, according to an actual decoding duration and a reference decoding duration range corresponding to a same media frame, a target proportion of media frames of which actual decoding durations do not fall within the reference decoding duration ranges; and

[0030] determine that the decoding speed time series of the decoder exceeds the reference decoding time series range when the target proportion is greater than or equal to the proportion threshold.

[0031] In one possible implementation, the processing unit 603 is further configured to:

[0032] determine, according to an actual decoding duration of a first media frame in the first multimedia data, the reference decoding duration ranges respectively corresponding to the media frames.

[0033] In one possible implementation, a first thread, a second thread, a third thread, and a fourth thread run in parallel in the decoder.

[0034] The first thread is used for performing hardware decoding on the first multimedia data, and the second thread is used for sequentially outputting the first multimedia data subjected to the decoding processing.

[0035] The third thread is used for copy the first multimedia data to obtain the second multimedia data.

[0036] The fourth thread is used for acquiring the running state of the decoder, and acquiring and decoding the second multimedia data when the decoder is in an idle state.

[0037] In one possible implementation, the first multimedia data refers to a cloud gaming video stream. The processing unit is further configured to:

[0038] acquire a scenario requirement of a cloud gaming, the scenario requirement being used for indicating a decoding mode of the cloud gaming video stream; and

[0039] acquire the running state of the decoder in the process of performing hardware decoding on the cloud gaming video stream when the scenario requirement indicates that an accelerated decoding mode is used for the cloud gaming video stream.

[0040] According to one aspect, the embodiments of this disclosure provide a computer device. The computer device includes a memory and one or more processors, the memory storing a computer-readable instruction, and the computer-readable instruction, when executed by the one or more processors, causing the one or more processors to perform the decoding processing method.

[0041] According to one aspect, the embodiments of this disclosure provide one or more non-volatile computer-readable storage media, having a computer-readable instruction stored thereon, and the computer-readable instruction, when executed by one or more processors, causing the one or more processors to implement the decoding processing method.

[0042] According to one aspect, the embodiments of this disclosure provide a computer program product or a computer program, the computer program product or the computer program including a computer-readable instruction stored in a computer-readable storage medium. A processor of a computer device reads the computer-readable instruction from the computer-readable storage medium, and the processor executes the computer-readable instruction to cause the computer device to perform the decoding processing method.

[0043] Details of one or more embodiments of this disclosure are provided in the accompanying drawings and descriptions below. Other features, objectives, and advantages of this disclosure become apparent from the description, the drawings, and the claims.

BRIEF DESCRIPTION OF THE DRAWINGS

[0044] To describe the technical solutions of embodiments of this disclosure more clearly, the following briefly introduces the accompanying drawings required for describing the embodiments. Apparently, the accompanying drawings in the following description show only some embodiments of this disclosure, and a person of ordinary skill in the art may still derive other drawings from these accompanying drawings without creative efforts.

[0045] FIG. 1 is a schematic flowchart of a decoding processing method according to embodiments of this disclosure.

[0046] FIG. 2 is a diagram of a decoding speed time series of a decoder according to embodiments of this disclosure.

[0047] FIG. 3 is a diagram of a reference decoding time series range according to embodiments of this disclosure.

[0048] FIG. 4 is a diagram of another reference decoding time series range according to embodiments of this disclosure.

[0049] FIG. 5 is a schematic flowchart of another decoding processing method according to embodiments of this disclosure.

[0050] FIG. 6 is a schematic structural diagram of a decoding processing apparatus according to embodiments of this disclosure.

[0051] FIG. 7 is a schematic structural diagram of a computer device according to embodiments of this disclosure.

DESCRIPTION OF EMBODIMENTS

[0052] Exemplary embodiments are described in detail herein, and examples of the exemplary embodiments are shown in the accompanying drawings. When the following description involves the accompanying drawings, unless otherwise indicated, the same numerals in different accompanying drawings represent the same or similar elements. The implementations described in the following exemplary embodiments do not represent all implementations consistent with this disclosure. On the contrary, the implementations are merely examples of apparatuses and methods that are described in detail in the appended claims and that are consistent with some aspects of this disclosure.

[0053] Embodiments of this disclosure provide a decoding processing scheme, which can stabilize the performance of a decoder, and can enable the decoder to maintain a constant decoding frame rate and maintain the stability of a decoding speed time series. The decoding processing scheme is applicable to service scenarios that require hardware-accelerated decoding processing, such as ultra-high frame rate real-time streaming media transmission and playback scenarios, and cloud gaming decoding and rendering scenarios that require ultra low latency. Next, the principle of the decoding processing scheme is correspondingly explained:

[0054] Firstly, first multimedia data that needs to be subjected to hardware decoding is acquired. The first multimedia data may be video data, audio data, or cloud gaming data, etc. Moreover, the first multimedia data may be a multimedia data stream or any data packet included in the multimedia data stream. Next, the acquired first multimedia data may be copied to obtain second multimedia data. It is to be understood that the first multimedia data includes a plurality of media frames. The media frames are copied to obtain corresponding cloned frames. The second

multimedia data includes these cloned frames. Furthermore, a running state of a decoder may be acquired in a process of performing hardware decoding on the first multimedia data, the running state of the decoder including a working state or an idle state. If the acquired running state of the decoder is an idle state, the decoder may be used to perform decoding processing on the second multimedia data in the idle state. In this way, it is ensured that the decoder is always at a constant working frequency so as to improve the performance of the decoder, enable the decoder to output a constant decoding frame rate, and maintain the stability of a decoding speed time series, thereby effectively improving the decoding efficiency.

[0055] In one possible implementation, the decoding processing scheme in this disclosure may be combined with the blockchain technology. For example, data such as the first multimedia data and the second multimedia data is uploaded into a blockchain for storing, which can ensure that the data on the blockchain is not prone to be tampered. Based on the scheme provided in the embodiments of this disclosure, in a process of performing hardware decoding on the first multimedia data, both the first multimedia data and the second multimedia data are sent and stored into a blockchain network, so that the second multimedia data is acquired when the decoder is in an idle state. Moreover, the decoder is used to perform decoding processing on the second multimedia data in the idle state.

[0056] The following describes the terminologies involved the embodiments of this disclosure:

[0057] Cloud technology can include a general term of network technologies, information technologies, integration technologies, management platform technologies, application technologies, etc. based on an application of cloud computing as a business model, can form a resource pool, can be used on demand, and is flexible and convenient. Cloud computing technology becomes an important support. A large variety of computing and storage resources are required for background services in a technical network system, such as video websites, picture websites, and more web portals. With the rapid development and application of the Internet industry, each item is likely to have a recognition mark in the future, which needs to be transmitted to a background system for logical processing. Data of different levels is processed separately. Therefore, data of various industries requires a powerful system to support, which is implemented only through cloud computing.

[0058] At present, cloud technology mainly includes cloud basic technology and cloud application. The cloud basic technology may be further divided into: cloud computing, cloud storage, database, big data, etc. The cloud application may be further divided into: medical cloud,

cloud Internet of Things, cloud security, cloud calling, private cloud, public cloud, hybrid cloud, cloud gaming, cloud education, cloud conference, cloud social networking, an artificial intelligence cloud service, etc.

[0059] From the perspective of the basic technology, the decoding processing method of this disclosure involves cloud computing in the cloud technology. From the perspective of the application, the decoding processing method of this disclosure involves cloud gaming in the cloud technology.

[0060] Cloud computing refers to the delivery and usage mode of IT infrastructure, and refers to obtaining required resources through the network in an on-demand and easy-to-expand manner. Cloud computing in a broad sense refers to the delivery and usage mode of services, and refers to obtaining required services through the network in an on-demand and easy-to-expand manner. Such services may be IT and software, Internet-related, or other services. Cloud computing is the product of the development and fusion of traditional computer and network technologies such as grid computing, distributed computing, parallel computing, utility computing, network storage technology, virtualization, and load balance.

[0061] Cloud gaming, also called gaming on demand, is an online gaming technology based on cloud computing technology. Cloud gaming technology enables thin clients with relatively limited graphics processing and data computing capabilities to run high-quality games. In a cloud gaming scenario, the games do not run on a player's game terminal, but run on a cloud server, and the cloud server renders the gaming scenario into a video and audio stream, which is transmitted to the player's game terminal through the network. The player's game terminal does not need to have powerful graphics computing and data processing capabilities, but only needs to have the basic streaming media playback capability and the capabilities to acquire player input commands and send same to the cloud server.

[0062] The decoding processing scheme provided in this disclosure involves larger-scale computing and requires larger computing power and storage space. Therefore, in one feasible implementation of this disclosure, a computer device may acquire sufficient computing power and storage space through the cloud gaming technology.

[0063] Furthermore, the decoding processing scheme provided in this disclosure may be applied to the cloud gaming scenario, and cloud gaming data (such as game image data and game audio data) in the cloud gaming scenario may be used as the first multimedia data, and then hardware-accelerated decoding processing is performed on the cloud gaming data, which can

stabilize the decoding performance of a decoder of a game client to a greater extent, thereby improving the user's game experience.

[0064] For the descriptions of the decoding processing scheme mentioned above, the following items need to be explained:

[0065] (1) The decoding processing scheme mentioned above may be executed by a computer device. The computer device may be a terminal device or a server. The terminal device may be a mobile phone, a tablet computer, a notebook computer, a handheld computer, a mobile Internet device (MID), a vehicle, a vehicle-mounted device, a roadside device, an aircraft, a smart wearable device (such as a smart watch, a smart bracelet, and a pedometer), a smart TV and other devices with a decoding processing function, etc. The server may be an independent physical server, may also be a server cluster or a distributed system composed of a plurality of physical servers, and may further be a cloud server that provides basic cloud computing services such as cloud services, cloud databases, cloud computing, cloud functions, cloud storage, network services, cloud communication, middleware services, domain name services, security services, content delivery networks (CDNs), big data, and artificial intelligence platforms.

[0066] (2) The computer device mentioned above needs to be provided with an Intel central processing unit (CPU) integrated graphics card, and a video hardware decoding operation is performed by a video decoding module of the Intel CPU integrated graphics card used in a service scenario.

[0067] (3) The decoding processing scheme mentioned above may be combined with the blockchain technology. Furthermore, in this disclosure, a process of decoding the second multimedia data by using the decoder may be executed on the blockchain, which can not only ensure the fairness of the decoding process, but also make the decoding process traceable, thereby improving the security of the decoding process.

[0068] (4) The decoding processing scheme mentioned above may be applied to a decoding processing module of a client in the cloud gaming scenario, and may also be applied to any service scenario that requires ultra low latency hardware video decoding. Through the high-performance decoding mode of the decoding processing scheme provided in this disclosure, the hardware decoding speed is improved. The decoder is used to perform decoding processing on the cloned multimedia data in the idle state, so that the stability of the decoding speed time series can be guaranteed, and a solid foundation is laid for the ultimate user experience at the product side.

[0069] Based on the analyses above, the following describes the decoding processing method in this disclosure with reference to FIG. 1. FIG. 1 is a schematic flowchart of a decoding processing method according to embodiments of this disclosure. The decoding processing method may be performed by a computer device. The computer device is, for example, a vehicle-mounted device, a smartphone, a tablet computer, a smart wearable device, and other terminal devices having a decoder installed. As shown in FIG. 1, the decoding processing method includes steps S110 to S130.

[0070] In S110, a running state of a decoder is acquired in a process of performing hardware decoding on first multimedia data, the running state including a working state or an idle state.

[0071] In the embodiments of this disclosure, the first multimedia data may be any data packet in a multimedia data stream, and may also be one single piece of data. The first multimedia data may include, but is not limited to: video data, audio data, or cloud gaming data, etc. Moreover, the first multimedia data is data encoded by an encoder. A data format of the first multimedia data may include, but is not limited to: H.264 format (which is a new generation of video compression format specified by the international telecommunication union), a moving picture experts group-4 (MPEG-4) format, which is an international standard dynamic image format, etc. The data format of the first multimedia data is not limited in the embodiments of this disclosure.

[0072] The hardware decoding refers to decoding a video through hardware, and is performed by a graphics processing unit (GPU). Decoding by the GPU can reduce the workload of the CPU and reduce power consumption. A decoding processing scenario involved in the embodiments of this disclosure is mainly a hardware decoding scenario. Therefore, the decoder in the embodiments of this disclosure may refer to a decoding module in an Intel CPU integrated graphics card in a computer device. The decoding module may include a video decoding module, an audio decoding module, etc.

[0073] In one possible implementation, after the first multimedia data is acquired, a performance parameter of the decoder is acquired. If it is determined, according to the performance parameter of the decoder, that the performance of the decoder does not meet a scenario requirement, the step of acquiring a running state of a decoder is then triggered to be performed. In addition, if it is determined, according to the performance parameter of the decoder, that the performance of the decoder meets the scenario requirement, decoding processing is directly performed on the first multimedia data through a related decoding processing method. The performance parameter of the decoder may include, but is not limited to: decoding time, decoding speed, stability of the decoding speed time series, and other parameters. That is, in the embodiments of this disclosure, after the first

multimedia data is acquired, it is pre-determined whether the performance of the decoder meets the scenario requirement; if the performance of the decoder does not meet the scenario requirement, the performance of the decoder may be stabilized in a specific scenario by performing the decoding processing method provided in the embodiments of this disclosure. The specific scenario includes a service scenario that requires high performance of the decoder, such as an ultra-high frame rate real-time streaming media transmission and playback scenario and an ultra low latency cloud gaming decoding and rendering scenario.

[0074] In the embodiments of this disclosure, the acquired running state of the decoder may include a working state or an idle state. In one possible implementation, if the running state of the decoder switches from an idle state to a working state, the decoding processing of the second multimedia data is paused. The working state refers to a state where the decoder is performing decoding processing on the received multimedia data. Moreover, the idle state refers to a state where the decoder is awaiting to receive multimedia data or is not performing decoding processing on the received multimedia data. For example, assume that a decoding rate of the decoder is 60 frames per second (fps, that is, the decoder decodes 60 frames of data per second), the decoder receives one piece of multimedia data every 16 ms. Therefore, the decoder is in a working state in a time period when the decoder performs decoding processing on the received multimedia data. Moreover, the decoder is in an idle state at a time period (e.g. 16 ms) when the decoder is awaiting to receive each piece of multimedia data. It is to be understood that, when the decoding processing that is being performed by the decoder is paused, interrupted, etc., the decoder may enter an idle state from a working state. Similarly, pausing, interruption, and other situations of the decoder are handled or restored, the decoder may enter a working state from an idle state. If the running state of the decoder switches from an idle state to a working state, the decoding processing of the second multimedia data is paused. The decoder performs decoding processing on the received multimedia data in priority, which can improve the decoding efficiency of the decoder and ensure the ordered decoding of the multimedia data.

[0075] In one possible implementation, the first multimedia data may refer to a multimedia data stream. The multimedia data stream may include N data packets, and M media frames are encapsulated in each data packet, N and M being both positive integers. Therefore, when receiving an i-th data packet of the multimedia data stream, the decoder may copy the i-th data packet to obtain a cloned packet of the i-th data packet. The i-th data packet may be any data packet in the multimedia data stream, i being a positive integer, and i being less than or equal to N. Moreover, the second multimedia data includes the cloned packet of the i-th data packet. In this case, the decoder

is in a working state, performs decoding processing on media frames in the i -th data packet, and outputs the media frames subjected to the decoding processing in the i -th data packet.

[0076] During specific implementation, assume that the multimedia data stream is a video data stream, and the media frame is an image frame, an audio frame, or a cloud gaming data frame, etc. Therefore, in a process of performing hardware decoding on a video data stream, when the decoder acquires any data packet in the video data stream, the decoder may copy the acquired data packet. That is, the decoder performs a cloneVideoPkt operation to obtain a cloned new packet of the data packet. Therefore, the second multimedia data includes the cloned new packet obtained by replication. When the decoder is in a working state, the decoder may sequentially perform decoding processing on media frames included in the acquired data packet, and then sequentially outputs the media frames subjected to the decoding processing frame by frame. In this way, the decoding processing is performed in units of data packets, and the replication processing is performed in units of data packets, which can effectively guarantee the orderliness of data processing and avoid data confusion. In one possible implementation, each media frame in the data packet may carry a media identifier. The media identifier of each media frame is used for indicating a position and sequence of the media frame in the data packet. The decoder may sequentially perform decoding processing on the media frames according to the media identifiers carried in the media frames, and sequentially outputs, according to the media identifiers carried in the media frames, the media frames subjected to the decoding processing. The media identifier includes, but is not limited to: a character, a number or a string, etc., which is not limited in the embodiments of this embodiment.

[0077] In S120, second multimedia data is acquired when the decoder is in an idle state, the second multimedia data being obtained by copying the first multimedia data.

[0078] In one possible implementation, the first multimedia data may refer to a multimedia data stream. The multimedia data stream may include N data packets. M media frames are encapsulated in each data packet. Therefore, when acquiring an i -th data packet of the multimedia data stream, the decoder may copy the i -th data packet to obtain a cloned packet of the i -th data packet, so that the second multimedia data may include the cloned packet of the i -th data packet.

[0079] In another possible implementation, when receiving an $(i+1)$ -th data packet of the multimedia data stream, the decoder may copy the $(i+1)$ -th data packet to obtain a cloned packet of the $(i+1)$ -th data packet. In addition, the cloned packet of the i -th data packet may be deleted, so that the second multimedia data includes the cloned packet of the $(i+1)$ -th data packet. That is, when the decoder acquires a new data packet, the decoder may replace, with a cloned packet obtained by copying the data packet, a cloned packet of a previous data packet. In this way, it is ensured that the

latest cloned packet is stored in a memory of the decoder, so that the storage burden is reduced, and the cache pressure of the memory is reduced, thereby further improving the performance of the decoder.

[0080] In S130, the decoder is used to perform decoding processing on the second multimedia data in the idle state.

[0081] In one possible implementation, the second multimedia data may include the cloned packet of the i-th data packet. The cloned packet of the i-th data packet includes M cloned frames. The M cloned frames have one-to-one correspondence to the M media frames in the i-th data packet. The step of using the decoder to perform decoding processing on the second multimedia data in the idle state may refer to: using the decoder to perform decoding processing on the cloned frames in the cloned packet of the i-th data packet in the idle state. In this way, data in the cloned packet of the i-th data packet have one-to-one correspondence to data in the i-th data packet. In the idle state, the decoder does not need to perform complex decoding processing, but only needs to conveniently perform decoding processing on the cloned frames in the cloned packet, which reduces the decoding pressure of the decoder in the idle state.

[0082] In the embodiments of this disclosure, in the process of performing hardware decoding on the first multimedia data, a performance parameter of the decoder may be acquired. If the performance parameter of the decoder does not meet a performance indicator requirement, the step of acquiring a running state of a decoder is triggered to perform.

[0083] In one possible implementation, the performance parameter of the decoder includes an actual decoding duration of the first multimedia data, and the performance indicator requirement includes a reference decoding frame rate. Therefore, an actual decoding frame rate of the decoder may be determined according to the actual decoding duration of the first multimedia data. If the actual decoding frame rate of the decoder is less than the reference decoding frame rate, it is determined that the performance parameter of the decoder does not meet the performance indicator requirement.

[0084] The actual decoding duration of the first multimedia data refers to a real decoding duration of the first multimedia data. The decoding frame rate refers to the number of media frames that can be decoded per second. The actual decoding frame rate refers to a real decoding frame rate of the decoder. The reference decoding frame rate refers to an expected decoding frame rate of the decoder, that is, an acceptable decoding frame rate.

[0085] For example, assume that the decoder has an actual decoding duration of 20 ms, and a reference decoding frame rate of 60 fps. The decoding frame rate refers to the number of media frames that can be decoded per second. Therefore, a relation between the decoding duration and the decoding frame rate may be expressed as: Decoding duration $t=1,000/\text{fps}$ (ms). Therefore, according to the actual decoding duration of 20 ms of the decoder for the first multimedia data, it can be determined that the actual decoding frame rate of the decoder is $1,000/20=50$ fps. The actual decoding frame rate (50 fps) of the decoder is less than the reference decoding frame rate (60 fps); therefore, it is determined that the performance parameter of the decoder does not meet the performance indicator requirement. For another example, assume that the decoder has an actual decoding duration of 10 ms for the first multimedia data, and a reference decoding frame rate is 60 fps. Therefore, according to the actual decoding duration of 10 ms of the decoder for the first multimedia data, it is determined that the actual decoding frame rate of the decoder is $1,000/10=100$ fps. The actual decoding frame rate (100 fps) of the decoder is greater than the reference decoding frame rate (60 fps); therefore, it is determined that the performance parameter of the decoder meets the performance indicator requirement. In this way, if the actual decoding frame rate of the decoder is less than the reference decoding frame rate, it is determined that the performance parameter of the decoder does not meet the performance indicator requirement. In this case, the step of acquiring a running state of a decoder is triggered to perform, so that the decoder maintains a continuous and stable working frequency, thereby increasing the decoding frame rate of the decoder, stabilizing the performance of the decoder, and improving the decoding efficiency.

[0086] In another possible implementation, the first multimedia data includes a plurality of media frames. The performance parameter of the decoder includes an actual decoding duration of each media frame in the first multimedia data. The performance indicator requirement includes a reference decoding time series range. Therefore, a decoding speed time series of the decoder may be calculated according to the actual decoding duration of each media frame. If the decoding speed time series of the decoder exceeds the reference decoding time series range, it is determined that the performance parameter of the decoder does not meet the performance indicator requirement.

[0087] The actual decoding duration of the media frame refers to a real decoding duration of the media frame. The decoding speed time series is composed of actual decoding durations of the media frames in the first multimedia data. The reference decoding time series range refers to an expected decoding time series range. The decoding time series range refers to an acceptable decoding duration interval.

[0088] For example, assume that the first media data includes n media frames, n being a positive integer. Then, the decoding speed time series of the decoder is obtained based on the actual decoding duration of each media frame. As shown in FIG. 2, FIG. 2 is a diagram of a decoding speed time series of a decoder according to embodiments of this disclosure. As shown in FIG. 2, the decoding speed time series of the decoder is composed of actual decoding durations of the media frames included in the first multimedia data. For example, an actual decoding duration t_1 of the first frame is 16 ms, an actual decoding duration t_2 of the second frame is 20 ms, an actual decoding duration t_3 of the third frame is 32 ms, an actual decoding duration t_4 of the fourth frame is 16 ms, an actual decoding duration t_5 of the fifth frame is 44 ms, and an actual decoding duration t_6 of the sixth frame is 24 ms. Therefore, the decoding speed time series of the decoder is composed of the actual decoding durations of the media frames. In addition, the computer device may acquire the reference decoding time series range. FIG. 3 is a diagram of a reference decoding time series range according to embodiments of this disclosure. In one possible implementation, the reference decoding time series range includes reference decoding duration ranges respectively corresponding to the media frames. The reference decoding duration ranges respectively corresponding to the media frames may be set in advance. The reference decoding duration ranges respectively corresponding to the media frames may be the same or different. In one possible implementation, the reference decoding time series range may be determined according to the actual decoding duration of the first frame. The reference decoding duration ranges respectively corresponding to the media frames are determined according to an actual decoding duration of a first media frame in the first multimedia data. The first media frame refers to the first frame in the first multimedia data. For example, assume that the actual decoding duration of the first media frame in the first multimedia data is t_1 , reference decoding duration ranges corresponding to all media frames in the first multimedia data are: $0.5 t_1 - 3 t_1$. That is, all the media frames correspond to the same reference decoding duration ranges ($0.5 t_1 - 3 t_1$). In this way, the reference decoding duration ranges respectively corresponding to the media frames are quickly determined according to an actual decoding duration of one media frame so as to quickly determine the reference decoding time series ranges. For example, if actual decoding durations of the remaining $(n-1)$ media frames fall within the reference decoding duration range, the decoding speed time series of the decoder does not exceed the reference decoding time series range; therefore, it is determined that the performance parameter of the decoder meets the performance indicator requirement. If an actual decoding duration of one media frame among the remaining $(n-1)$ media frames does not fall within the reference decoding duration range, the decoding speed time series of the decoder exceeds the

reference decoding time series range; therefore, it is determined that the performance parameter of the decoder does not meet the performance indicator requirement.

[0089] In this way, if the decoding speed time series of the decoder exceeds the reference decoding time series range, it is determined that the performance parameter of the decoder does not meet the performance indicator requirement. In this case, the step of acquiring a running state of a decoder is triggered to perform, so that the decoder maintains a continuous and stable working frequency, thereby shortening the decoding speed time series of the decoder, stabilizing the performance of the decoder, and improving the decoding efficiency.

[0090] In another possible implementation, furthermore, it may also be determined, based on a proportion threshold, whether the decoding speed time series of the decoder exceeds the reference decoding time series range. The reference decoding time series range includes reference decoding duration ranges respectively corresponding to the media frames in the first multimedia data. A target proportion of media frames of which actual decoding durations do not fall within the reference decoding duration ranges is determined according to an actual decoding duration and a reference decoding duration range corresponding to a same media frame in the first multimedia data. If the target proportion is greater than or equal to the proportion threshold, it is determined that the decoding speed time series of the decoder exceeds the reference decoding time series range.

[0091] In one possible implementation, the target proportion of media frames of which actual decoding durations do not fall within the reference decoding duration ranges refers to a proportion of media frames among all the media frames in the first multimedia data of which actual decoding durations do not fall within the reference decoding duration range. For example, the first multimedia data includes 10 media frames, and the number of media frames of which actual decoding durations do not fall within the reference decoding duration ranges is 5; therefore, a target proportion of the media frames is $5/10=0.5$.

[0092] In one possible implementation, if the reference decoding time series range is determined according to an actual decoding duration of a first media frame in the first multimedia data, the target proportion of media frames of which actual decoding durations do not fall within the reference decoding duration ranges may refer to a proportion of media frames among all the media frames other than the first media frame in the first multimedia data of which actual decoding durations do not fall within the reference decoding duration range, or may refer to a proportion of media frames among all the media frames in the first multimedia data of which actual decoding durations do not fall within the reference decoding duration range.

[0093] For example, assume that the actual decoding duration of the first media frame in the first multimedia data is t_1 , reference decoding duration ranges corresponding to all media frames in the first multimedia data are: $0.5 t_1 - 3 t_1$. A proportion threshold may be set in advance. The number of media frames among $(n-1)$ media frames of which reference decoding duration ranges do not fall within a reference decoding duration range is determined according to an actual decoding duration and a reference decoding duration range ($0.5 t_1 - 3 t_1$) of the $(n-1)$ media frames. Assume that the number is k , k being a positive integer and k being less than or equal to $(n-1)$. A proportion $k/n-1$ of the media frames which do not fall within the reference duration range is then calculated. According to $k/n-1$ and the proportion threshold, it is determined whether the decoding speed time series of the decoder exceeds the reference decoding time series range. If $k/n-1$ is less than the proportion threshold, it is determined that the performance parameter of the decoder meets the performance indicator requirement. If $k/n-1$ is greater than or equal to the proportion threshold, it is determined that the performance parameter of the decoder does not meet the performance indicator requirement.

[0094] The proportion threshold may be adaptively adjusted according to a scenario requirement, or manually set according to experience, which is not specifically limited in the embodiments of this disclosure.

[0095] In this way, according to the target proportion and the proportion threshold, it is determined whether the decoding speed time series of the decoder exceeds the reference decoding time series range, so that when actual decoding durations of most media frames in the first multimedia data are expired, it is determined that the decoding speed time series of the decoder exceeds the reference decoding time series range, which effectively guarantees the determination accuracy.

[0096] In another possible implementation, the reference decoding time series range includes reference decoding duration ranges corresponding to two adjacent media frames. The reference decoding duration ranges corresponding to any two adjacent media frames may be the same or different. If the reference decoding duration ranges corresponding to any two adjacent media frames are different, it is roughly equivalent to reference decoding duration ranges corresponding to the media frames being different. Assume that the actual decoding duration of the first frame is t_1 , the actual decoding duration of the second frame is t_2 , the actual decoding duration of the third frame is t_3 , ..., and the actual decoding duration of an n -th frame is t_n . Assume that a reference decoding duration range (an absolute value) between t_1 and t_2 may be: $0 \leq t_2 - t_1 \leq \max 1$, a reference decoding duration range between t_2 and t_3 may be: $0 \leq t_3 - t_2 \leq \max 2$, a reference decoding

duration range between t_3 and t_4 may be: $0 \leq t_4 - t_3 \leq \max_3$, ..., and a reference decoding duration range between t_{n-1} and t_n may be: $0 \leq t_n - t_{n-1} \leq \max_{n-1}$, $\max_1$, $\max_2$, $\max_3$, ..., and $\max_{n-1}$ being different. At least two pieces of identical data may also exist.

[0097] For example, referring to FIG. 4, FIG. 4 is a diagram of another reference decoding time series range according to embodiments of this disclosure. Firstly, according to the diagram of the decoding speed time series of the decoder shown in FIG. 2, it is determined that a decoding duration difference between the first frame and the second frame is: $20 \text{ ms} - 16 \text{ ms} = 4 \text{ ms}$, a decoding duration difference between the second frame and the third frame is: $32 \text{ ms} - 20 \text{ ms} = 12 \text{ ms}$, and so on, and decoding duration differences of 16 ms, 28 ms, 20 ms, etc. are sequentially obtained. Then, assume that a decoding duration range between two adjacent media frames is shown in FIG. 4, that is, a reference decoding duration range between the first frame and the second frame is: 0-10 ms, a reference decoding duration range between the second frame and the third frame is: 0-15 ms, and so on, and a reference decoding duration range between the fifth frame and the sixth frame is: 0-30 ms. One reference decoding duration range corresponds to one decoding duration difference (that is, a difference between actual decoding durations of any two adjacent media frames). Then, based on the determined decoding duration difference between the any two adjacent media frames, it is sequentially determined whether decoding duration differences between any two adjacent media frames fall within corresponding reference decoding duration ranges. For example, the decoding duration difference between the first frame and the second frame is 4 ms, which falls within the corresponding reference decoding duration range (0-10 ms), and the decoding duration difference between the second frame and the third frame is 12 ms, which also falls within the corresponding reference decoding duration range (0-15 ms). For another example, the decoding duration difference between the fourth frame and the fifth frame is 28 ms, which does not fall within the corresponding reference decoding duration range (0-25 ms). Finally, if the decoding duration differences between any two adjacent media frames all fall within the corresponding reference decoding duration ranges, it is determined that the performance parameter of the decoder meets the performance indicator requirement. If the decoding duration differences between two adjacent media frames do not fall within the corresponding reference decoding duration ranges, it is determined that the decoding speed time series of the decoder exceeds the reference decoding time series range, that is, it is determined that the performance parameter of the decoder does not meet the performance indicator requirement.

[0098] Through the manner, according to the performance parameter of the decoder, it is pre-determined whether the decoder meets the performance indicator requirement. The performance indicator requirements in different service scenarios may be certainly different. If the decoder does

not meet the performance indicator requirement, the performance of the decoder may be optimized and the working efficiency of the decoder may be improved through the decoding processing method provided in the embodiments of this disclosure.

[0099] In conclusion, the embodiments of this disclosure may be applied to a cloud gaming scenario. That is, the first multimedia data refers to a cloud gaming video stream; therefore, a scenario requirement of a cloud gaming may be acquired, the scenario requirement being used for indicating a decoding mode of the cloud gaming video stream. If the scenario requirement indicates that an accelerated decoding mode is used for the cloud gaming video stream, the running state of the decoder is acquired in the process of performing hardware decoding on the cloud gaming video stream.

[00100] In one possible implementation, when developing the cloud gaming, game developers may preset a game mode of the game as an accelerated decoding mode. Therefore, in the cloud gaming scenario, the decoding processing scheme provided in the embodiments of this disclosure may be used to perform hardware-accelerated decoding processing on the cloud gaming video stream, which can stabilize the decoding performance of a decoder of a game client to a greater extent, thereby improving the user's game experience.

[00101] In another possible implementation, the scenario requirement of the cloud gaming may be determined according to a network environment of the game client where the game runs. If a network environment of a game client where user A is located is a weak network environment, it is determined that the scenario requirement of the cloud gaming indicates that an accelerated decoding mode needs to be used for the cloud gaming video stream. If a network environment of a game client where user B is located is a strong network environment, it is determined that the scenario requirement of the cloud gaming indicates that an accelerated decoding mode does not need to be used for the cloud gaming video stream.

[00102] In the ultra-high frame rate real-time streaming media transmission and playback scenario, the ultra low latency cloud gaming decoding and rendering scenario, etc., the embodiments of this disclosure can solve the problems of long decoding time and poor stability of the time series which may occur in the scenario that requires ultra low latency hardware decoding so as to guarantee the smooth and stable video decoding performance, thereby providing the ultimate user experience for services that require ultra low latency such as cloud gaming.

[00103] In the embodiments of this disclosure, a running state of a decoder is acquired in a process of performing hardware decoding on the first multimedia data, the running state including an idle state or a working state. If the running state of the decoder is an idle state, second

multimedia data obtained by copying the first multimedia data is acquired, and the decoder is used to perform decoding processing on the second multimedia data in the idle state. According to this disclosure, in the process of performing, by the decoder, hardware decoding on the first multimedia data, the cloned first multimedia data is decoded in the idle state of the decoder, so that the decoder maintains a continuous and stable working frequency, thereby stabilizing the performance of the decoder and improving the decoding efficiency thereof.

[00104] Referring to FIG. 5, FIG. 5 is a schematic flowchart of another decoding processing method according to embodiments of this disclosure. As shown in FIG. 5, the decoding processing method is performed by the decoder. A first thread, a second thread, a third thread, and a fourth thread run in parallel in the decoder. With reference to the decoding processing method in the embodiments in FIG. 1, the first thread (thread 1) shown in FIG. 5 is used for: performing hardware decoding on the first multimedia data. The second thread (thread 2) is used for: sequentially outputting the first multimedia data subjected to the decoding processing. The third thread (thread 3) is used for: copying the first multimedia data to obtain the second multimedia data. The fourth thread (thread 4) is used for: acquiring the running state of the decoder, and acquiring and decoding the second multimedia data when the decoder is in an idle state.

[00105] In the embodiments of this disclosure, the following describes the example of performing, by the decoder, hardware decoding processing on the multimedia data stream in detail. Assume that the multimedia data stream includes a plurality of data packets. Next, the decoding processing method provided in the embodiments of this disclosure is further described in detail with reference to FIG. 5. The decoding processing method shown in FIG. 5 includes steps S501 to S509.

[00106] In S501, a new data packet is awaited.

[00107] In one possible implementation, the multimedia data stream is based on streaming. Streaming means that data packets in the multimedia data stream are transmitted one after another. That is, when acquiring a data packet in the multimedia data stream, the decoder completes performing decoding processing on the data packet, and then awaits to acquire the next data packet, and performs decoding processing on the next data packet, and so on. A duration when a new data packet needs to be awaited may be determined according to a decoding frame rate of the decoder. A decoding duration $t=1,000/\text{fps}$ (ms). For example, the decoding frame rate of the decoder is 60 fps, and after acquiring a first data packet, the decoder needs to await for $t=1,000/60=16$ ms before receiving the next data packet.

[00108] In S502, first multimedia data is acquired.

[00109] In the embodiments of this disclosure, the first multimedia data may be any data packet among N data packets included in the multimedia data stream, and M media frames are encapsulated in each data packet, N and M being both positive integers. That is, M media frames are encapsulated in the first multimedia data acquired by thread 1 running in the decoder. Each media frame includes, but is not limited to: an image frame, an audio frame, a cloud gaming video frame, etc.

[00110] In S503, a replication operation is performed.

[00111] In one possible implementation, when acquiring a new data packet, the decoder may simultaneously start thread 3 to perform a cloneVideoPkt operation (a replication operation) on the data packet to obtain a cloned new packet (second multimedia data).

[00112] In S504, whether a decoder is in an idle state is determined.

[00113] In one possible implementation, thread 4 acquires a running state (including an idle state or a working state) of the decoder, and determines whether the running state of the decoder is an idle state. If the running state of the decoder is an idle state, the operation of acquiring the second multimedia data by thread 4 is triggered to perform. The idle state refers to whether the decoder is in a state where the decoder is awaiting to receive a new data packet; if yes, it is determined that the running state of the decoder is an idle state.

[00114] In S505, second multimedia data is acquired.

[00115] In one possible implementation, if the running state of the decoder is an idle state, thread 4 may acquire the second multimedia data. The second multimedia data may also include a cloned packet of any data packet. For example, the second multimedia data acquired by thread 4 is a cloned packet obtained by cloning a current data packet acquired by thread 1, or the second multimedia data is a cloned packet obtained by cloning a previous data packet acquired by thread 1. Moreover, the cloned packet of the previous data packet is replaced with a cloned packet of the latest data packet, which can reduce the storage burden of the decoder.

[00116] In S506, the second multimedia data is decoded.

[00117] In one possible implementation, thread 4 may perform a decodeClonePacket operation in the idle state, that is, thread 4 running in the decoder performs decoding processing on the second multimedia data. The second multimedia data includes M cloned frames. The M cloned frames have one-to-one correspondence to the M media frames in the first multimedia data. Therefore, the decoder performs decoding processing on the second multimedia data, which means that the decoder sequentially performs decoding processing on the M cloned frames.

[00118] In one possible implementation, if the running state of the decoder switches from the idle state to a working state, thread 4 pauses decoding processing of the second multimedia data.

[00119] In S507, the first multimedia data is decoded.

[00120] In one possible implementation, after acquiring the first multimedia data, thread 1 may perform decoding processing on the first multimedia data. The decoder performs a decodeOneFrame operation, that is, the decoder performs decoding processing on the first multimedia data, which means that the decoder sequentially performs decoding processing on the M media frames.

[00121] In S508, a decode callback is performed.

[00122] In one possible implementation, decode callback may refer to: sequentially outputting, by thread 2, the M media frames subjected to the decoding processing included in the first multimedia data.

[00123] In S509, whether reading is ended is determined.

[00124] In one possible implementation, after completing decoding processing of a data packet, thread 1 may determine whether reading of a current multimedia data stream is ended; if yes, it is determined that the decoding processing of the multimedia data stream is completed; if not, the next new data packet is continued to be awaited, and the step above is repeated performing until reading of the multimedia data stream is ended. If it is detected that a current data packet carries an end identifier, it is determined that reading of the multimedia data stream is ended.

[00125] The step numbers in steps S501 to S509 are not used to limit the execution orders of the steps. For example, step S502 and step S503 are simultaneously performed. For another example, step S506 is performed prior to step S507, or step S506 is performed after step S507.

[00126] In the embodiments of this disclosure, a running state of a decoder is acquired in a process of performing hardware decoding on the first multimedia data, the running state including an idle state or a working state. If the running state of the decoder is an idle state, second multimedia data obtained by copying the first multimedia data is acquired, and the decoder is used to perform decoding processing on the second multimedia data in the idle state. According to this disclosure, in the process of performing, by the decoder, hardware decoding on the first multimedia data, the cloned first multimedia data is decoded in the idle state of the decoder, so that the decoder maintains a continuous and stable working frequency, thereby stabilizing the performance of the decoder and improving the decoding efficiency thereof.

[00127] It is to be understood that although the steps are displayed sequentially according to the instructions of the arrows in the flowcharts of the embodiments, these steps are not necessarily performed sequentially according to the sequence instructed by the arrows. Unless otherwise explicitly specified in this disclosure, execution of the steps is not strictly limited, and the steps may be performed in other sequences. Moreover, at least some of the steps in each embodiment may include a plurality of steps or a plurality of stages. The steps or stages are not necessarily performed at the same moment but may be performed at different moments. Execution of the steps or stages is not necessarily sequentially performed, but may be performed alternately with other steps or at least some of steps or stages of other steps.

[00128] Referring to FIG. 6, FIG. 6 is a schematic structural diagram of a decoding processing apparatus according to embodiments of this disclosure. The decoding processing apparatus 600 may be applied to the computer device in the method embodiments corresponding to FIG. 1 and FIG. 5. The decoding processing apparatus 600 may be a computer program (including a program code) running in the computer device. For example, the decoding processing apparatus 600 is application software. The apparatus may be configured to perform the corresponding steps of the method provided in the embodiments of this disclosure. The decoding processing apparatus 600 may include: an acquisition unit 601 and a decoding unit 602.

[00129] The acquisition unit 601 is configured to acquire a running state of a decoder in a process of performing hardware decoding on first multimedia data, the running state including a working state or an idle state;

[00130] The acquisition unit 601 is further configured to acquire second multimedia data when the decoder is in an idle state, and the second multimedia data being obtained by copying the first multimedia data; and

[00131] The decoding unit 602 is configured to use the decoder to perform decoding processing on the second multimedia data in the idle state.

[00132] In one possible implementation, the decoding unit 602 is further configured to: pause decoding processing of the second multimedia data if the running state of the decoder switches from the idle state to a working state.

[00133] In one possible implementation, the first multimedia data refers to a multimedia data stream. The multimedia data stream includes N data packets, and M media frames are encapsulated in each data packet, N and M being both positive integers. The decoding processing apparatus 600 may further include a processing unit 603.

[00134] The processing unit 603 is configured to copy an i -th data packet of the multimedia data stream when the decoder receives the i -th data packet to obtain a cloned packet of the i -th data packet, the second multimedia data including the cloned packet of the i -th data packet, i being a positive integer, and i being less than or equal to N .

[00135] The decoding unit 602 is further configured to: perform decoding processing on media frames in the i -th data packet when the decoder is in a working state, and output the media frames subjected to the decoding processing in the i -th data packet.

[00136] In one possible implementation, the cloned packet of the i -th data packet includes M cloned frames, and the M cloned frames have one-to-one correspondence to the M media frames in the i -th data packet. The decoding unit 602 is specifically configured to: use the decoder to perform decoding processing on the cloned frames in the cloned packet of the i -th data packet in the idle state.

[00137] In one possible implementation, the processing unit 603 is further configured to: copy an $(i+1)$ -th data packet of the multimedia data stream when the decoder receives the $(i+1)$ -th data packet to obtain a cloned packet of the $(i+1)$ -th data packet.

[00138] The processing unit 603 is further configured to delete the cloned packet of the i -th data packet, so that the second multimedia data includes the cloned packet of the $(i+1)$ -th data packet.

[00139] In one possible implementation, the acquisition unit 601 is further configured to: acquire a performance parameter of the decoder in the process of performing hardware decoding on the first multimedia data; and trigger to perform the step of acquiring a running state of a decoder if the performance parameter of the decoder does not meet a performance indicator requirement.

[00140] In one possible implementation, the performance parameter of the decoder includes an actual decoding duration of the first multimedia data, and the performance indicator requirement includes a reference decoding frame rate. The processing unit 603 is further configured to:

[00141] determine an actual decoding frame rate of the decoder according to the actual decoding duration; and

[00142] determine that the performance parameter of the decoder does not meet the performance indicator requirement when the actual decoding frame rate of the decoder is less than the reference decoding frame rate.

[00143] In one possible implementation, the first multimedia data includes a plurality of media frames. The performance parameter of the decoder includes an actual decoding duration of

each media frame in the first multimedia data. The performance indicator requirement includes a reference decoding time series range. The processing unit 603 is further configured to:

[00144] calculate a decoding speed time series of the decoder according to the actual decoding duration of each media frame; and

[00145] determine that the performance parameter of the decoder does not meet the performance indicator requirement when the decoding speed time series of the decoder exceeds the reference decoding time series range.

[00146] In one possible implementation, the reference decoding time series range includes reference decoding duration ranges respectively corresponding to the media frames. The processing unit 603 is further configured to:

[00147] determine, according to an actual decoding duration and a reference decoding duration range corresponding to a same media frame, a target proportion of media frames of which actual decoding durations do not fall within the reference decoding duration ranges; and

[00148] determine that the decoding speed time series of the decoder exceeds the reference decoding time series range when the target proportion is greater than or equal to the proportion threshold.

[00149] In one possible implementation, the processing unit 603 is further configured to:

[00150] determine, according to an actual decoding duration of a first media frame in the first multimedia data, the reference decoding duration ranges respectively corresponding to the media frames.

[00151] In one possible implementation, a first thread, a second thread, a third thread, and a fourth thread run in parallel in the decoder.

[00152] The first thread is used for performing hardware decoding on the first multimedia data. The second thread is used for sequentially outputting the first multimedia data subjected to the decoding processing.

[00153] The third thread is used for copying the first multimedia data to obtain the second multimedia data.

[00154] The fourth thread is used for acquiring the running state of the decoder, and acquiring and decoding the second multimedia data when the decoder is in an idle state.

[00155] In one possible implementation, the first multimedia data refers to a cloud gaming video stream. The processing unit 603 is further configured to:

[00156] acquire a scenario requirement of a cloud gaming, the scenario requirement being used for indicating a decoding mode of the cloud gaming video stream; and

[00157] acquire the running state of the decoder in the process of performing hardware decoding on the cloud gaming video stream when the scenario requirement indicates that an accelerated decoding mode is used for the cloud gaming video stream.

[00158] In the embodiments of this disclosure, a running state of a decoder is acquired in a process of performing hardware decoding on the first multimedia data, the running state including an idle state or a working state. If the running state of the decoder is an idle state, second multimedia data obtained by copying the first multimedia data is acquired, and the decoder is used to perform decoding processing on the second multimedia data in the idle state. According to this disclosure, in the process of performing hardware decoding on the first multimedia data by the decoder, the cloned first multimedia data is decoded in the idle state of the decoder, so that the decoder maintains a continuous and stable working frequency, thereby stabilizing the performance of the decoder and improving the decoding efficiency thereof.

[00159] Referring to FIG. 7, FIG. 7 is a schematic structural diagram of a computer device according to embodiments of this disclosure. The computer device 700 is configured to perform the steps performed by the computer device in the method embodiments corresponding to FIG. 1 and FIG. 5. The computer device 700 includes: one or more processors 710; and one or more input devices 720, one or more output devices 730, and a memory 740. The processor 710, the input device 720, the output device 730, and the memory 740 are connected through a bus 750. The memory 740 is configured to store a computer program, the computer program including a computer-readable instruction. The processor 710 is configured to call the computer-readable instruction stored in the memory 740 to perform the following operations:

[00160] acquiring a running state of a decoder in a process of performing hardware decoding on first multimedia data, the running state including a working state or an idle state;

[00161] acquiring second multimedia data when the decoder is in an idle state, the second multimedia data being a copy of the first multimedia data; and

[00162] using the decoder to perform decoding processing on the second multimedia data in the idle state.

[00163] In one possible implementation, if the running state of the decoder switches from an idle state to a working state, the processor 710 pauses the decoding processing of the second multimedia data.

[00164] In one possible implementation, the first multimedia data refers to a multimedia data stream. The multimedia data stream includes N data packets, and M media frames are encapsulated in each data packet, N and M being both positive integers. The processor 710 is further configured to perform the following operations:

[00165] copying an i-th data packet of the multimedia data stream when the decoder receives the i-th data packet to obtain a cloned packet of the i-th data packet, the second multimedia data including the cloned packet of the i-th data packet, i being a positive integer, and i being less than or equal to N; and

[00166] performing decoding processing on media frames in the i-th data packet when the decoder is in a working state, and outputting the media frames subjected to the decoding processing in the i-th data packet.

[00167] In one possible implementation, the cloned packet of the i-th data packet includes M cloned frames, and the M cloned frames have one-to-one correspondence to the M media frames in the i-th data packet. The using, by the processor 710, the decoder to perform decoding processing on the second multimedia data in the idle state includes:

[00168] using the decoder to perform decoding processing on the cloned frames in the cloned packet of the i-th data packet in the idle state.

[00169] In one possible implementation, the processor 710 copies an (i+1)-th data packet of the multimedia data stream when the decoder receives the (i+1)-th data packet to obtain a cloned packet of the (i+1)-th data packet.

[00170] The processor 710 deletes the cloned packet of the i-th data packet, so that the second multimedia data includes the cloned packet of the (i+1)-th data packet.

[00171] In one possible implementation, a performance parameter of the decoder is acquired in the process of performing, by the processor 710, hardware decoding on the first multimedia data.

[00172] If the performance parameter of the decoder does not meet a performance indicator requirement, the step of acquiring a running state of a decoder is triggered to be performed.

[00173] In one possible implementation, the performance parameter of the decoder includes an actual decoding duration of the first multimedia data, and the performance indicator requirement includes a reference decoding frame rate. The processor 710 is further configured to perform the following operations:

[00174] determining an actual decoding frame rate of the decoder according to the actual decoding duration; and

[00175] determining that the performance parameter of the decoder does not meet the performance indicator requirement when the actual decoding frame rate of the decoder is less than the reference decoding frame rate.

[00176] In one possible implementation, the first multimedia data includes a plurality of media frames. The performance parameter of the decoder includes an actual decoding duration of each media frame in the first multimedia data. The performance indicator requirement includes a reference decoding time series range. The processor 710 is further configured to perform the following operations:

[00177] calculating a decoding speed time series of the decoder according to the actual decoding duration of each media frame; and

[00178] determining that the performance parameter of the decoder does not meet the performance indicator requirement when the decoding speed time series of the decoder exceeds the reference decoding time series range.

[00179] In one possible implementation, the reference decoding time series range includes reference decoding duration ranges respectively corresponding to the media frames. The processor 710 is further configured to perform the following operations:

[00180] determining, according to an actual decoding duration and a reference decoding duration range corresponding to a same media frame, a target proportion of media frames of which actual decoding durations do not fall within the reference decoding duration ranges; and

[00181] determining that the decoding speed time series of the decoder exceeds the reference decoding time series range when the target proportion is greater than or equal to the proportion threshold.

[00182] In one possible implementation, the processor 710 is further configured to perform the following operation:

[00183] determining, according to an actual decoding duration of a first media frame in the first multimedia data, the reference decoding duration ranges respectively corresponding to the media frames.

[00184] In one possible implementation, a first thread, a second thread, a third thread, and a fourth thread run in parallel in the decoder.

[00185] The first thread is used for performing hardware decoding on the first multimedia data. The second thread is used for sequentially outputting the first multimedia data subjected to the decoding processing.

[00186] The third thread is used for copying the first multimedia data to obtain the second multimedia data.

[00187] The fourth thread is used for acquiring the running state of the decoder, and acquiring and decoding the second multimedia data when the decoder is in an idle state.

[00188] In one possible implementation, the first multimedia data refers to a cloud gaming video stream. The processor 710 is further configured to perform the following operations:

[00189] acquiring a scenario requirement of a cloud gaming, the scenario requirement being used for indicating a decoding mode of the cloud gaming video stream; and

[00190] acquiring the running state of the decoder in the process of performing hardware decoding on the cloud gaming video stream when the scenario requirement indicates that an accelerated decoding mode is used for the cloud gaming video stream.

[00191] In the embodiments of this disclosure, a running state of a decoder is acquired in a process of performing hardware decoding on the first multimedia data, the running state including an idle state or a working state. If the running state of the decoder is an idle state, second multimedia data obtained by copying the first multimedia data is acquired, and the decoder is used to perform decoding processing on the second multimedia data in the idle state. According to this disclosure, in the process of performing, by the decoder, hardware decoding on the first multimedia data, the cloned first multimedia data is decoded in the idle state of the decoder, so that the decoder maintains a continuous and stable working frequency, thereby stabilizing the performance of the decoder and improving the decoding efficiency thereof.

[00192] It is to be understood that the computer device 700 described in the embodiments of this disclosure may execute the description of the decoding processing method in the embodiments corresponding to FIG. 1 and FIG. 5, and may also execute the description of the decoding processing apparatus 600 in the embodiments corresponding to FIG. 6, which will not be repeated here.

[00193] In addition, the embodiments of this disclosure further provide a computer storage medium. A computer program is stored in the computer storage medium. The computer program includes a computer-readable instruction. A processor, when executing the program instruction, may perform the method in the embodiments corresponding to FIG. 1 and FIG. 5. Therefore, the

details will not be repeated here again. For the technical details which are not disclosed in the embodiment of the computer storage medium involved in this disclosure, please refer to the description of the method embodiment of this disclosure. As an example, the program instruction is executed on a single computer device, or on a plurality of computer devices in a location, or on a plurality of computer devices distributed in a plurality of locations and interconnected through a communication network.

[00194] According to one aspect of this disclosure, a computer program product or a computer program is provided. The computer program product or the computer program includes a computer-readable instruction. The computer-readable instruction is stored in a computer-readable storage medium. A processor of a computer device reads the computer-readable instruction from the computer-readable storage medium, and the processor executes the computer-readable instruction, so that the computer device may perform the method in the embodiments corresponding to FIG. 1 and FIG. 5. Therefore, the details will not be repeated here.

[00195] A person of ordinary skill in the art may understand that all or some of procedures of the method in the foregoing embodiments may be implemented by the computer-readable instruction instructing relevant hardware. The program may be stored in a computer-readable storage medium. When the program is executed, the procedures of the foregoing method embodiments may be implemented. The foregoing storage medium may be a floppy disk, an optical disc, a read-only memory (ROM), or a random-access memory (RAM), etc.

[00196] What is disclosed above is merely exemplary embodiments of this disclosure, and certainly is not intended to limit the scope of the claims of this disclosure. Therefore, equivalent variations made in accordance with the claims of this disclosure shall fall within the scope of this disclosure.

CLAIMS

1. A method for decoding processing, performed by a computer device, and comprising:
 - acquiring a running state of a decoder in a process of performing hardware decoding on first multimedia data, the running state comprising a working state or an idle state;
 - when the decoder is in an idle state,
 - acquiring second multimedia data, the second multimedia data being a copy of the first multimedia data; and
 - performing, by the decoder, decoding on the second multimedia data in the idle state.
2. The method according to claim 1, further comprising:
 - pausing the decoding of the second multimedia data when the running state of the decoder switches from the idle state to a working state.
3. The method according to claim 1, wherein the first multimedia data comprises a multimedia data stream, the multimedia data stream comprising N data packets, M media frames are encapsulated in each of data packet, and N and M are both positive integers; and the method further comprises:
 - copying an i-th data packet of the multimedia data stream when the decoder receives the i-th data packet to obtain a cloned packet of the i-th data packet, the second multimedia data comprising the cloned packet of the i-th data packet, i being a positive integer, and i being less than or equal to N; and
 - performing decoding on media frames in the i-th data packet when the decoder is in a working state, and
 - outputting the media frames subjected to the decoding in the i-th data packet.
4. The method according to claim 3, wherein the cloned packet of the i-th data packet comprises M cloned frames, and the M cloned frames have a one-to-one correspondence to the M media frames in the i-th data packet; and
 - performing, by the decoder, decoding on the second multimedia data in the idle state comprises:
 - performing, by the decoder, decoding on the cloned frames in the cloned packet of the i-th data packet in the idle state.

5. The method according to claim 3, further comprising:

copying an (i+1)-th data packet of the multimedia data stream when the decoder receives the (i+1)-th data packet to obtain a cloned packet of the (i+1)-th data packet; and

deleting the cloned packet of the i-th data packet, so that the second multimedia data comprises the cloned packet of the (i+1)-th data packet.

6. The method according to claim 1, further comprising:

acquiring a performance parameter of the decoder in the process of performing the hardware decoding on the first multimedia data; and

acquiring the running state of the decoder when the performance parameter of the decoder does not meet a performance indicator requirement.

7. The method according to claim 6, wherein the performance parameter of the decoder comprises an actual decoding duration of the first multimedia data, and the performance indicator requirement comprises a reference decoding frame rate; and the method further comprises:

determining an actual decoding frame rate of the decoder according to the actual decoding duration; and

determining that the performance parameter of the decoder does not meet the performance indicator requirement when the actual decoding frame rate of the decoder is less than the reference decoding frame rate.

8. The method according to claim 6, wherein the first multimedia data comprises a plurality of media frames, the performance parameter of the decoder comprises an actual decoding duration of each media frame in the first multimedia data, and the performance indicator requirement comprises a reference decoding time series range; and the method further comprises:

calculating a decoding speed time series of the decoder according to the actual decoding duration of each media frame; and

determining that the performance parameter of the decoder does not meet the performance indicator requirement when the decoding speed time series of the decoder exceeds the reference decoding time series range.

9. The method according to claim 8, wherein the reference decoding time series range comprises reference decoding duration ranges respectively corresponding to the media frames, and the method further comprises:

determining, according to an actual decoding duration and a reference decoding duration range corresponding to a same media frame, a target proportion of the media frames of which the actual decoding durations do not fall within the reference decoding duration ranges; and

determining that the decoding speed time series of the decoder exceeds the reference decoding time series range when the target proportion is greater than or equal to a proportion threshold.

10. The method according to claim 9, further comprising:

determining, according to the actual decoding duration of a first media frame in the first multimedia data, the reference decoding duration ranges respectively corresponding to the media frames.

11. The method according to claim 1, wherein a first thread, a second thread, a third thread, and a fourth thread run in parallel in the decoder;

the first thread is configured to perform hardware decoding on the first multimedia data, and the second thread is configured to sequentially output the first multimedia data subjected to the decoding processing;

the third thread is configured to copy the first multimedia data to obtain the second multimedia data; and

the fourth thread is configured to acquire the running state of the decoder, and acquire and decode the second multimedia data when the decoder is in the idle state.

12. The method according to any one of claims 1 to 11, wherein the first multimedia data includes a cloud gaming video stream; and the method further comprises:

acquiring a scenario requirement of a cloud gaming, the scenario requirement indicating a decoding mode of the cloud gaming video stream; and

acquiring the running state of the decoder in the process of performing the hardware decoding on the cloud gaming video stream when the scenario requirement indicates that an accelerated decoding mode is required for the cloud gaming video stream.

13. A decoding processing apparatus, comprising:

an acquisition unit, configured to acquire a running state of a decoder in a process of performing hardware decoding on first multimedia data, the running state comprising a working state or an idle state;

when the decoder is in an idle state,

the acquisition unit being further configured to acquire second multimedia data, the second multimedia data being a copy of the first multimedia data; and

perform decoding, by the decoder, on the second multimedia data in the idle state.

14. A computer device, comprising a memory and one or more processors, the memory storing a computer-readable instruction, and the one or more processors, when executing the computer-readable instruction, implementing the decoding processing method according to any one of claims 1 to 12.

15. One or more non-volatile computer-readable storage media, storing computer-readable instructions, the computer-readable instructions, when executed by one or more processors, implementing the decoding processing method according to any one of claims 1 to 12.

16. A computer program product, comprising a computer-readable instruction, the computer-readable instruction, when executed by one or more processors, performing the steps of the method according to any one of claims 1 to 12.