The present describes low latency streaming using temporal frame transformation. An execution component in an edge server executes a first instance of an application. A server interface component receives, from a remote server, a resolution delta frame indicating differences between a high resolution first frame and a low resolution first frame of a second instance of the application or, alternatively, receives the high resolution first frame. A video manipulation component generates a motion delta frame by identifying differences between a low resolution first frame and a low resolution second frame of the first instance of the application. The video manipulation component generates a high resolution transformed frame by applying the resolution delta frame and the motion delta frame to the low resolution second frame.
Published:

— with international search report (Art. 21(3))
LOW LATENCY APPLICATION STREAMING USING TEMPORAL FRAME TRANFORMATION

TECHNICAL FIELD

[0001] The present disclosure relates to low latency application streaming using temporal frame transformation.

BACKGROUND

[0002] Client devices, particularly mobile devices, equipped with high-resolution touchscreens, powerful processors, and graphics processing units (GPUs) are increasingly in use for many applications including those associated with interactive applications such as electronic gaming. As client device screens increase in size and resolution, finer graphical detail and advanced graphical effects are becoming more common for interactive applications. And the processing capabilities of client device GPUs are constantly improving. Even so, many client devices cannot yet provide the rich visual experience and sophisticated graphical detail found in dedicated gaming consoles or high-end desktop computers. Several reasons exist for this discrepancy including power consumption and size. A high-end desktop GPU may consume 500 watts, whereas a high-end client device GPU may consume less than 10 watts. Moreover, client device GPUs likely will continue to lag behind desktop GPUs since battery capacity of many client devices is limited given that larger capacity batteries are bulky and may be inconsistent with many client device form factors particularly mobile device form factors.

[0003] A common approach to address these issues is to develop a version of the interactive application specifically designed considering the constraints of the hardware included in particular client devices. Doing so, oftentimes results in a lack of graphical details, such as complex lighting, shadows, particles, fog, detailed textures, complex models, non-critical graphical objects such as blades of grass in the field, and the like. This approach is oftentimes employed in modern games to allow such games to run on underpowered hardware, such as low-end desktop and laptop computers. Thus, through custom graphics settings, different levels of visual detail and effects can be enabled or disabled (e.g., using a configuration screen) to obtain a sufficient level of detail while allowing the game to be rendered at a full frame rate.

[0004] Alternatively, execution of the interactive application may be fully or partially offloaded to a remote server to make the application more accessible to all users. Such an approach may utilize general-purpose workloads running on the central processing units
(CPUs) of the client devices. Or execution of game logic and rendering may be offloaded to a remote server using a thin client architecture in which the client device (thin client) receives and transmits user input to the server. The server, in turn, executes the game logic and rendering based on the received user input and provides the resulting audio-visual or visual only output encoded as compressed video to the client device. This approach is vulnerable to excessive latency if the server is not relatively geographically close to the client. Excessive latency may adversely and undesirably affect the user gaming experience. A need remains, therefore, for improved low latency interactive application solutions.

**BRIEF DRAWINGS DESCRIPTION**

[0005] The present disclosure describes various embodiments that may be understood and fully appreciated in conjunction with the following drawings:

Fig. 1 diagrams an embodiment of a system according to the present disclosure;

Figs. 2A-C diagram embodiments of a system according to the present disclosure;

Fig. 3 diagrams an embodiment of a timeline of a system according to the present disclosure;

Fig. 4 diagrams an embodiment of a system according to the present disclosure; and

Fig. 5 diagrams an embodiment of a method according to the present disclosure.

**DETAILED DESCRIPTION**

[0006] The present disclosure describes embodiments with reference to the drawing figures listed above. Persons of ordinary skill in the art will appreciate that the description and figures illustrate rather than limit the disclosure and that, in general, the figures are not drawn to scale for clarity of presentation. Such skilled persons will also realize that many more embodiments are possible by applying the inventive principles contained herein and that such embodiments fall within the scope of the disclosure that is not to be limited except by the claims.

[0007] Remote rendering and streaming of interactive applications may involve a user on a client device that may interact with a remote server. The remote server may render the interactive application frames and audio and may stream the resulting audiovisual output stream to a client device via a network. Given the general availability of rendering and encoding hardware, a differentiator of the user experience is perceived latency, which may be defined as a time between the user providing input to the application and the user observing the result of that input. A factor that may influence perceived latency is a physical distance between the client device and the remote server that renders the
application. Unfortunately, a factor that influences cost is the location of the remote server. Latency has been improved by geographically locating render servers close to the client devices that receive user input. Unfortunately, many such users cluster in dense metropolitan areas that lack availability of large spaces to locate large datacenters that include remote servers. And the space tends to cost a premium in metropolitan areas.

Given the high cost of space in metropolitan areas and its limited availability, it is often challenging to provide low latency capacity for many of these markets using remote servers in datacenters. The techniques described herein may allow leveraging geographically distant or remote, cost effective, servers in datacenters that do not introduce undesirable levels of latency to adversely affect the user interactive experience. The techniques described herein may further enable a client device to offload high performance graphics rendering and processing to a remote server and an edge server in a manner that ensures low latency. The client device may interact with the remote server and the edge server in a common streaming model. The remote server, located geographically remote from the client device, may execute a second instance of the application and may render a first high resolution frame and a first low resolution frame of the application. In an embodiment, the remote server may generate a resolution delta frame that indicates the differences between the first high resolution frame and the first low resolution frame. The remote server may transmit the resolution delta frame to the edge server. In an alternative embodiment, the remote server may transmit the high resolution frames it generates to the edge server, which, in turn, may generate the resolution delta frame. Transmitting the resolution delta frame from the remote server to the edge server may reduce the bandwidth requirements between the remote server and the edge server as compared to transmitting the high resolution frame from the remote server to the edge server. In either circumstance, however, the resolution delta frame or the high resolution frames generated at the remote server provide a level of detail lacking in the low resolution frames generated by the edge server.

The edge server, located geographically close to the client device, may execute a first instance of the application and may render first and second low resolution frames. The edge server and the remote server may execute the first instance and the second instance, respectively, of the application in a synchronized manner explained in more detail below. The edge server generates a motion delta frame that indicates differences between the first and second low resolution frames. The edge server may provide the
client device with a high resolution transformed frame generated by transforming the low
resolution second frame based at least in part on the resolution delta frame received by the
remote server (or alternatively, the edge server may generate the resolution delta frame
based on receiving, from the remote server, the high resolution first frame generated by
the remote server). By doing so, the client device may experience the same low latency as
if the remote server were located geographically close to the client device while achieving
improved game graphics rendering and processing.

[0010] Fig. 1 diagrams an embodiment of a system 100 according to the present
disclosure. Referring to Fig. 1, system 100 includes a computing device 102 that may
execute instructions defining applications, components, objects, routines, programs,
instructions, data structures, virtual machines, and the like that perform particular tasks or
functions or that implement particular data types. Instructions that define such
applications, components, objects, routines, programs, virtual machines, and the like may
be stored in memory, e.g., system memory 106, remote memory 134, or external memory
136. Some or all of the applications may be instantiated at run time by one or more
processors comprised in a processing unit, e.g., processing device 104. Some or all of the
applications may be interactive in that they operate in response to user input. A person of
ordinary skill in the art will recognize that many of the concepts associated with the
exemplary embodiment of system 100 may be implemented as computer instructions,
firmware, or software in any of a variety of computing architectures, e.g., computing
device 102, to achieve a same or equivalent result.

[0011] Moreover, a person of ordinary skill in the art will recognize that the exemplary
embodiment of system 100 may be implemented on other types of computing
architectures, e.g., general purpose or personal computers, hand-held devices, mobile
communication devices, gaming devices, music devices, photographic devices, multi-
processor systems, microprocessor-based or programmable consumer electronics,
minicomputers, mainframe computers, application specific integrated circuits, and like.
For illustrative purposes only, system 100 is shown in Fig. 1 to include computing devices
102, geographically remote computing devices 102R, tablet computing device 102T,
mobile computing device 102M, and laptop computing device 102L. A person of ordinary
skill in the art may recognize that computing device 102 may be embodied in any of tablet
computing device 102T, mobile computing device 102M, or laptop computing device
102L. Mobile computing device 102M may include mobile cellular devices, mobile
gaming devices, gaming consoles, mobile reader devices, mobile photographic devices, and the like.

[0012] A person of ordinary skill in the art will recognize that an exemplary embodiment of system 100 may be implemented in a distributed computing system in which various computing entities or devices, often geographically remote from one another, e.g., computing device 102 and remote computing device 102R, perform particular tasks or execute particular applications, objects, components, routines, programs, instructions, data structures, virtual machines, and the like. For example, the exemplary embodiment of system 100 may be implemented in a server/client configuration connected via network 130 (e.g., computing device 102 may operate as a server and remote computing device 102R or tablet computing device 102T may operate as a client, all connected through network 130). Network 130 may be a single network or several logically or physically distinct networks. In distributed computing systems, applications may be stored in local memory 106, external memory 136, or remote memory 134. Local memory 106, external memory 136, or remote memory 134 may be any kind of memory, volatile or non-volatile, removable or non-removable, known to a person of ordinary skill in the art including random access memory (RAM), flash memory, read only memory (ROM), ferroelectric RAM, magnetic storage devices, optical discs, and the like.

[0013] Computing device 102 comprises processing device 104, memory 106, device interface 108, and network interface 110, which may all be interconnected through bus 112. Processing device 104 represents a single, central processing unit, or a plurality of processing units in a single or two or more computing devices 102, e.g., computing device 102 and remote computing device 102R. In some circumstances, computing device 102 may further comprise a graphics processing unit (GPU) 105 specially designed to rapidly manipulate or alter memory 106 to accelerate the creation of images in a frame buffer intended for output to a display. GPU 105 may be part of processing device 104 or distinct from processing device 104. GPU 105 may be specially designed to efficiently manipulate computer graphics and images particularly computer graphics and images comprising large blocks of data.

[0014] Local memory 106, as well as external memory 136 or remote memory 134, may be any type memory device known to a person of ordinary skill in the art including any combination of RAM, flash memory, ROM, ferroelectric RAM, magnetic storage devices, optical discs, and the like that is appropriate for the particular task. Local memory 106 may store a database, indexed or otherwise. Local memory 106 may store a basic
input/output system (BIOS) 106A with routines executable by processing device 104 to transfer data, including data 106D, between the various elements of system 100. Local memory 106 also may store an operating system (OS) 106B executable by processing device 104 that, after being initially loaded by a boot program, manages other programs in the computing device 102. Memory 106 may store routines or programs executable by processing device 104, e.g., applications or programs 106C. Applications or programs 106C may make use of the OS 106B by making requests for services through a defined application program interface (API). Applications or programs 106C may be used to enable the generation or creation of any application program designed to perform a specific function directly for a user or, in some cases, for another application program. Examples of application programs include games, interactive entertainment, word processors, database programs, browsers, development tools, drawing, paint, and image editing programs, communication programs, tailored applications, and the like. Users may interact directly with computing device 102 through a user interface such as a command language or a user interface displayed on a monitor (not shown). Local memory 106 may be comprised in a processing unit, e.g., processing device 104.

Device interface 108 may be any one of several types of interfaces. Device interface 108 may operatively couple any of a variety of devices, e.g., hard disk drive, optical disk drive, magnetic disk drive, or the like, to the bus 112. Device interface 108 may represent either one interface or various distinct interfaces, each specially constructed to support the particular device that it interfaces to the bus 112. Device interface 108 may additionally interface input or output devices utilized by a user to provide direction to the computing device 102 and to receive information from the computing device 102. These input or output devices may include voice recognition devices, gesture recognition devices, touch recognition devices, keyboards, monitors, mice, pointing devices, speakers, stylus, microphone, joystick, game pad, satellite dish, printer, scanner, camera, video equipment, modem, monitor, and the like (not shown). Device interface 108 may be a serial interface, parallel port, game port, firewire port, universal serial bus, or the like.

A person of ordinary skill in the art will recognize that the system 100 may use any type of computer readable medium accessible by a computer, such as magnetic cassettes, flash memory cards, compact discs (CDs), digital video disks (DVDs), cartridges, RAM, ROM, flash memory, magnetic disc drives, optical disc drives, and the like. A computer readable medium as described herein includes any manner of computer program product, computer storage, machine readable storage, or the like.
Network interface 110 operatively couples the computing device 102 to one or more remote computing devices 102R, tablet computing devices 102T, mobile computing devices 102M, and laptop computing devices 102L, on a local, wide, or global area network 130. Computing devices 102R may be geographically remote from computing device 102. Remote computing device 102R may have the structure of computing device 102 and may operate as server, client, router, switch, peer device, network node, or other networked device and typically includes some or all of the elements of computing device 102. Computing device 102 may connect to network 130 through a network interface or adapter included in the interface 110. Computing device 102 may connect to network 130 through a modem or other communications device included in the network interface 110. Computing device 102 alternatively may connect to network 130 using a wireless device 132. The modem or communications device may establish communications to remote computing devices 102R through global communications network 130. A person of ordinary skill in the art will recognize that programs 106C might be stored remotely through such networked connections. Network 130 may be local, wide, global, or otherwise and may include wired or wireless connections employing electrical, optical, electromagnetic, acoustic, or other carriers as is known to a person of ordinary skill in the art.

The present disclosure may describe some portions of the exemplary system 100 using algorithms and symbolic representations of operations on data bits within a memory, e.g., memory 106. A person of ordinary skill in the art will understand these algorithms and symbolic representations as most effectively conveying the substance of their work to others of ordinary skill in the art. An algorithm is a self-consistent sequence leading to a desired result. The sequence requires physical manipulations of physical quantities. Usually, but not necessarily, these quantities take the form of electrical or magnetic signals capable of being stored, transferred, combined, compared, and otherwise manipulated. For simplicity, the present disclosure refers to these signals as bits, values, elements, symbols, characters, terms, numbers, or like. The terms are merely convenient labels. A person of skill in the art will recognize that terms such as computing, calculating, generating, loading, determining, displaying, or like refer to the actions and processes of a computing device, e.g., computing device 102. The computing device 102 may manipulate and transform data represented as physical electronic quantities within a memory into other data similarly represented as physical electronic quantities within the memory.
In an embodiment, system 100 may be a distributed network in which some computing devices 102 operate as servers, e.g., computing device 102, to provide content, services, or the like, through network 130 to other computing devices operating as clients, e.g., remote computing device 102R, laptop computing device 102L, tablet computing device 102T. In some circumstances, computing devices 102 may operate as servers that execute application programs offloaded from other computing devices 102 operating as clients.

Figs. 2A-C diagram embodiments of a system 200 according to the present disclosure. Referring to Figs. 1 and 2A-C, a system 200 may comprise a remote server 202RS, an edge server 202ES, and a client 202C. In an embodiment, remote server 202RS, edge server 202ES, and client device 202C may be structured similarly to computing device 102 to include processing device 104, memory 106, device interface 108, bus 112, and network interface 110. Remote server 202RS may include a more powerful processing device 104, GPU 105, or larger capacity or different type of memory 106 relative to those in edge server 202ES or client device 202C or both based on various design constraints including performance, cost, size, mobility, and the like. Similarly, edge server 202ES may include a more powerful processing device 104, GPU 105, or larger capacity or different type of memory 106 relative to those in client device 202C or both based on various design constraints including performance, cost, size, mobility, and the like. In some circumstances, client device 202C may be a thin client that relies on remote server 202RS or edge server 202ES for computationally intensive tasks, e.g., rendering and processing interactive application graphics.

A user may interact with client device 202C using a graphical user interface that, in turn, transmits user input 203 to remote server 202RS and edge server 202ES via network 230. Client device 202C may transmit user input 203 to remote server 202RS and edge server 202ES in any manner known to a person of ordinary skill in the art.

For simplicity, the description below will describe remote server 202RS and edge server 202ES executing instances of an interactive application, e.g., a game, a program, or the like. Remote server 202RS and edge server 202ES may execute distinct instances of a particular interactive application, e.g., a second application instance 208RS and a first application instance 208ES. As is described in more detail below, remote server 202RS and edge server 202ES may render frames and audio corresponding to second application instance 208RS and first application instance 208ES, respectively. In an embodiment, remote server 202RS may render a first quality first frame 209 of second application...
instance 208RS and a second quality first frame 211 of second application instance 208RS. First quality first frame 209 may have a first quality attribute, e.g., a high resolution. Similarly, second quality first frame 211 may have a second quality attribute, e.g., a low resolution. Note that the labels "first" and "second" as it refers to application instances, first quality, and the like are merely a convenience and not intended to denote any temporal or other relationship between the items so labeled.

[0023] Remote server 202RS may be one of a large group of networked computer servers in a datacenter located geographically remote from client device 202C. Edge server 202ES may render a second quality first frame 215 of first application instance 208ES and a second quality second frame 217 of first application instance 208ES. Second quality first frame 215 and second quality second frame 217 may have the second quality attribute, e.g., a low resolution, e.g., 320x240 pixels, lower than a high resolution, e.g., 1920x1080 pixels. In an embodiment, second quality second frame 217 may sequentially follow second quality first frame 215 in first application instance 208ES. In an embodiment, client device 202C may not be aware of that edge server 202ES is coupled to remote server 202RS so that, e.g., client device 202C may run legacy thin client software without needing to upgrade to support the resolution boost provided by the remote server 202RS.

[0024] In an embodiment, edge server 202ES may be located geographically close to client device 202C. Client device 202C may only interact directly with edge server 202ES. As is explained in more detail below, locating edge server 202ES geographically close to client 202C may improve the perceived latency of a user interacting with the application by reducing the time between the user providing an input and the user observing a result of that input in the application while offloading computationally intensive application graphics processing to remote server 202RS. Client device 202C may interact with edge server 202ES in a known streaming model that receives a video stream in any format known to a person of ordinary skill in the art, e.g., H.264/5 stream.

[0025] Referring to Figs. 2A-C, remote server 202RS may comprise a synchronization component 204RS, an execution component 206RS, and a video manipulation component 210RS. Synchronization component 204RS, execution component 206RS, and/or video manipulation component 210RS may be implemented in hardware, firmware, or software components, objects, circuits, devices, or the like, or a combination thereof.
Synchronization component 204RS may operate cooperatively with synchronization component 204ES in edge server 202ES to synchronize the execution of second application instance 208RS on remote server 202RS with execution of first application instance 208ES on edge server 202ES. Remote server 202RS and edge server 202ES may therefore be running distinct instances 208RS and 208ES of a same application. In an embodiment, edge server 202ES may operate as a master while remote server 202RS may operate in a synchronized or spectate mode locked to an identical user point of view. In another embodiment, edge server 202ES may synchronize game play by forwarding inputs. In yet another embodiment, edge server 202ES may operate in a real-time record and playback mode in which it forks and forwards render calls (either complete instructions or only send frame instruction deltas) to remote server 202RS, which would have copies of application assets, textures, geometries, and the like since these are costly to transfer from edge server 202ES. Remote server 202RS may execute instructions on, e.g., GPU 105, and return values are not sent back to the source and instead stored as needed to map memory/pointer locations.

Execution component 206RS may be configured to execute a second application instance 208RS. Execution component 206RS may generate a first quality first frame 209 of second application instance 208RS having a frame rate, e.g., 60 frames per second, and a first quality attribute, e.g., a first resolution of 1920x1080 pixels. Execution component 206RS may further generate a second quality first frame 211 of second application instance 208RS having the (same) frame rate, e.g., 60 frames per second, and a second quality attribute, e.g., a second resolution of 320x240 pixels. In an embodiment, first quality first frame 209 and second quality first frame 211 represent a same frame, e.g., a first frame, of second application instance 208RS with each having different quality attributes, e.g., a first resolution and a second resolution, respectively.

The first and second quality attributes may refer to any quality attribute of frames of application instance 208RS or 208ES that are well known to a person of ordinary skill in the art. In some circumstances, the first and second quality attributes may refer to first and second resolutions, respectively, of frames 209, 211, 215, or 217. The first resolution may be different, e.g., lower, than the first resolution. Thus, execution component 206RS may generate first quality first frame 209 and second quality first frame 211 as a high resolution first frame and a low resolution first frame, respectively, of second application instance 208RS.
In an embodiment, video manipulation component 210RS may generate a resolution delta frame 213 by identifying quality attribute differences, e.g., resolution differences, between first quality first frame 209 and second quality first frame 211. Video manipulation component 210RS may identify quality attribute differences between first quality first frame 209 and second quality first frame 211 using any video analysis techniques that are known to a person of ordinary skill in the art. Resolution delta frame 213 may aid in creating a high resolution transformed frame 221 from a low resolution second frame 217 generated by edge server 202ES.

Remote server 202RS may transmit resolution delta frame 213 to edge server 202ES via network 230 in any format known to a person of ordinary skill in the art, including H.264/5 stream. Alternatively, remote server 202RS may transmit first quality first frame 209 to edge server 202ES via network 230 in any manner known to a person of ordinary skill in the art, including H.264/5 stream.

Alternatively, remote server 202RS may transmit first quality first frame 209 to edge server 202ES and video manipulation component 210ES at the edge server 202ES may generate the resolution delta frame 213ES (Figs. 2B or 2C) at the edge server 202ES based at least in part on the first quality first frame 209 received from remote server 202RS.

Edge server 202ES may comprise a synchronization component 204ES, an execution component 206ES, and a video manipulation component 210ES. Execution component 206ES may be configured to execute a first application instance 208ES. Execution component 206ES may generate a second quality first frame 215 of first application instance 208ES having the second quality attribute, e.g., a low resolution. Execution component 206ES may further generate a second quality second frame 217 of first application instance 208ES having the second quality attribute, e.g., the low resolution. In an embodiment, execution component 206ES may generate audio 225 corresponding to first application instance 208ES. In an embodiment, second quality first frame 215 and second quality second frame 217 represent sequential frames, e.g., a first frame and a second frame, respectively, of first application instance 208ES. Execution component 206ES may generate second quality first frame 215 and second quality second frame 217 as a low resolution, e.g., 320x240 pixels, first frame 215 and a low resolution, e.g., 320x240 pixels, second frame 217.

Video manipulation component 210ES may generate a motion delta frame 219 by identifying differences, e.g., motion differences, between second quality first frame 215
and second quality second frame 217. Video manipulation component 210RS may identify motion differences between second quality first frame 215 and second quality second frame 217 using video analysis techniques that are known to a person of ordinary skill in the art, e.g., pixel-by-pixel basis, motion vectors, macro-blocks.

Video manipulation component 210ES may generate a transformed frame 221 by applying resolution delta frame 213 and motion delta frame 219 to second quality second frame 217 using any video manipulation techniques known to a person of ordinary skill in the art. Video manipulation component 210ES may transmit transformed frame 221 and audio 225 to client device 202C using any means known to a person of ordinary skill in the art for display on display 214C.

Edge server 202ES may operate in a collaborative rendering mode and an edge only mode. In the collaborative rendering mode, edge server 202ES may operate to collaboratively with remote server 208RS as is explained herein. In the edge only mode, edge server 202ES may render frames having a quality attribute supported by its hardware components without the aid of remote server 202RS. This situation may arise when, for example, remote server 202RS is not available for edge server 202ES to offload rendering.

Collaborative rendering mode between remote server 202RS and edge server 202ES allow users to experience low latencies as if remote server 202RS were located geographically close to client device 202C but with a density improvement in the edge server 202ES. Rendering performance for edge server 202ES may only determine density not capability since edge server 202ES need only render frames at a low resolution, e.g., 320x240 pixels and remote server 202RS may generate frames at a high resolution, e.g., 1920x1080 pixels, even if edge server 202ES may not natively support that high resolution.

Fig. 3 diagrams an embodiment of a timeline according to the present disclosure. Referring to Figs. 2A-C and 3, remote server 202RS may generate sequential high resolution frames Hi_1, Hi_2, and so on synchronous with edge server 202ES generating sequential low resolution frames Lo_1, Lo_2, and so on. In the example shown in Fig. 4, remote server 202RS and edge server 202ES generate frames every 16ms but any frame rate comes within the scope of the disclosure. Remote server 202RS will transmit either the high resolution frame, e.g., Hi_1, or a resolution delta frame 213 to edge server 202ES during the generation of the corresponding low resolution frame, e.g., Lo_1. Edge server 202ES may generate transformed frame 221, e.g., Hi_2', by applying the resolution delta
frame 213 and the motion delta frame 219 to the low resolution second frame, e.g., Lo_2. Edge server 202ES may transmit transformed frame 221, e.g., Hi_2', to client 202C. By doing so, latency is reduced as shown at 302.

[0038] Fig. 4 diagrams an embodiment of a system 400 according to the present disclosure. Referring to Figs. 1 and 4, a remote server 402RS may have a similar structure as remote server 202RS including a synchronization component 404RS, execution component 406RS, and video manipulation component 410RS. Like remote server 202RS, remote server 402RS may execute a second application instance 408RS and generate a resolution delta frame 421RS. System 400 includes a client 402C that operates similarly to edge server 202ES in system 200. Client 402C includes a synchronization component 404C, execution component 406C, and video manipulation component 410C that operate similarly to synchronization component 204ES, execution component 206ES, and video manipulation component 210ES as described above. Like edge server 202ES, client 402C may execute a first application instance 408C and generate a motion delta frame 412C and a transformed frame 421C. Collaborative rendering mode between remote server 402RS and client 402C may improve the native rendering capability of client device 402C at a small tradeoff of latency. Collaborative rendering mode may allow a client device 402C that can only render frames at a low resolution to instead interact with the application in a high resolution, e.g., 1920x1080 or 4k pixels, by having remote server 402RS render the high resolution frames and having client device 402C do the motion transform and display.

[0039] Further, in a collaborative rendering mode could also be leveraged for a seamless 'stream to play local' scenario. In this scenario, a user starts an application stream on a local device while the application downloads in the background. Once the download is complete, the application starts on the local device, synchronizes state with the remote server, and provides the natively rendered frames to the application instead of the video stream from the service.

[0040] And further yet, in streaming spectator style latency tolerant viewing scenarios, a remote server may provide the video stream (and possibly re-encode to lower target) to viewers thereby removing any additional load from valuable high density edge servers.

[0041] Fig. 5 diagrams an embodiment of a method 500 according to the present disclosure. Referring to Figs. 2A-C and 5, at 502, a method 500 executes an instance of an application on edge server 202ES. At 504, a method 500 generates a low resolution first and second frames 215, 217 of the application on edger server 202ES. At 506, method
500 determines whether edge server 202ES is operating in collaborative rendering mode, and if it is, method 500 generates a motion delta frame 219 at 508. At 510, edge server 202ES receives a resolution delta frame 213 from remote server 202RS that indicates the differences between a high resolution first frame and a low resolution first frame of an instance of the application executing on remote server 202RS synchronously with the instance of the application executing on edge server 202ES. At 512, method 500 generates a high resolution transformed frame on edge server 202ES by applying resolution delta frame 213 and motion delta frame 219 to low resolution first frame 215. At 514, method 500 transmits the transformed frame to a client device 202C. If the edge server 202ES is not operating in collaborative rendering mode at 506, method 500 generates application frames in a resolution consistent with its hardware, and transmits those frames to the client device 202C at 516.

[0042] It will also be appreciated by persons of ordinary skill in the art that the present disclosure is not limited to what has been particularly shown and described hereinabove. Rather, the scope of the present disclosure includes both combinations and sub-combinations of the various features described hereinabove as well as modifications and variations which would occur to such skilled persons upon reading the foregoing description. Thus the disclosure is limited only by the appended claims.
CLAIMS

1. A system, comprising:
   at least one processing device; and
   a storage device configured to store instructions that, when executed by the at least one processing device, cause the at least one processing device to:
   execute a first instance of an application;
   generate a first instance first frame and a first instance second frame corresponding to the first instance of the application, the first instance first frame and the first instance second frame each comprising a first level quality attribute;
   receive, from a remote server, a resolution delta frame identifying a difference between a second instance first quality first frame and a second instance second quality first frame corresponding to a second instance of the application executing on the remote server, the second instance first quality first frame comprising the first level quality attribute and the second instance second quality first frame comprising a second level quality attribute;
   transform the first instance second frame into a transformed frame based at least in part on applying the resolution delta frame received from the remote server to the first instance second frame, the transformed frame comprising the second quality level attribute; and
   transmit the transformed frame to a client.

2. The system of claim 1, the storage device further configured to store instructions that, when executed by the at least one processing device, cause the at least one processing device further to:
   generate a motion delta frame configured to identify a difference between the first instance first frame and the first instance second frame, the motion delta frame comprising motion compensation information; and
   transform the first instance second frame into the transformed frame based at least in part on the motion delta frame.

3. The system of claim 1,
   wherein the first quality attribute is a first resolution; and
   wherein the second quality attribute is a second resolution higher than the first resolution.

4. The system of claim 1, wherein the storage device further configured to store instructions that, when executed by the at least one processing device, cause the at
least one processing device further to:

substantially synchronize the first instance of the application with the second instance of the application executing on the remote server.

5. The system of claim 4,
wherein the first instance first frame corresponds to the second instance first quality first frame executing on the remote server.

6. The system of claim 4, the storage device further configured to store instructions that, when executed by the at least one processing device, cause the at least one processing device further to:
lock execution of the second instance of the application by the remote server to execution of the first instance of the application.

7. The system of claim 1, the storage device further configured to store instructions that, when executed by the at least one processing device, cause the at least one processing device further to:
genenerate audio corresponding to the application; and
transmit the audio to the client together with the transformed frame.

8. A method, comprising:
executing, on an edge server, a first instance of an application;
generating, on the edge server, a motion delta frame comprising motion compensation information to identify differences between a first frame of the first instance of the application and a second frame of the first instance of the application, the first frame of the first instance and the second frame of the first instance comprising a first level quality attribute;
receiving, at the edge server from a remote server, a resolution delta frame identifying differences between a first quality first frame of a second instance of the application executing on the remote server and a second quality first frame of the second instance of the application executing on the remote server, the first quality first frame of the second instance comprising the first level quality attribute and the second quality first frame of the second instance comprising a second level quality attribute;
transforming, at the edge server, the second frame of the first instance into a transformed frame based at least in part on applying the motion delta frame and the resolution delta frame to the second frame of the first instance; and
transmitting, from the edge server, the transformed frame to a client.
9. The method of claim 8, wherein the motion delta frame identifies macro-block differences between the first frame of the first instance of the application and the second frame of the first instance of the application.

10. The method of claim 9, wherein the resolution delta frame identifies pixel-by-pixel differences between the first quality first frame of the second instance of the application and the second quality first frame of the second instance of the application.

11. The method of claim 8, wherein the first level quality attribute comprises a first resolution; and wherein the second level quality attribute comprises a second resolution.

12. The method of claim 11, wherein the first resolution is a lower resolution than the second resolution; and wherein the transformed frame comprises the second resolution.

13. A system, comprising:
   an execution component configured to execute a first instance of an application;
   a server interface component configured to receive, from a remote server, a resolution delta frame that indicates differences between a first resolution first frame of a second instance of the application executing on the remote server and a second resolution first frame of the second instance of the application executing on the remote server;
   a video manipulation component configured to:
      identify a first resolution second frame of the first instance of the application; and
      transform the first resolution second frame of the first instance into a second resolution transformed frame by applying the resolution delta frame to the first resolution second frame of the first instance of the application; and
   a client interface component configured to transmit the second resolution transformed frame to a client, the first resolution second frame of the first instance and the first resolution first frame of the second instance comprising a first resolution and the second resolution first frame of the second instance comprising a second resolution higher than the first resolution.

14. The system of claim 13, wherein the video manipulation component is further configured to:
generate a motion delta frame to identify differences between a first resolution first frame of the first instance and the first resolution second frame of the first instance; and transform the first resolution second frame of the first instance into the second resolution transformed frame based at least in part on the motion delta frame.

15. The system of claim 13, further comprising:
a synchronization component to synchronize the second instance of the application executing on the remote server with the first instance of the application.
FIG. 3
Execute an instance of an application on an edge server

Generate low resolution first and second frames on the edge server

Collaborative Mode?

Yes

Generate a motion delta frame

Receive a resolution delta frame (or the high resolution frame) from a remote server

Generate transformed frame by applying the resolution delta frame and the motion delta frame to the low resolution second frame

Transmit the high resolution transformed frame to a client device

Generate application frames and transmit frames to the client device

Fig. 5
**INTERNATIONAL SEARCH REPORT**

**A. CLASSIFICATION OF SUBJECT MATTER**

INV. H04N 19/132 H04N 19/61

ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

**B. FIELDS SEARCHED**

Minimum documentation searched (classification system followed by classification symbols)

H04N

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic database consulted during the international search (name of data base and, where practicable, search terms used)

EPO-Internal, WPI Data

**C. DOCUMENTS CONSIDERED TO BE RELEVANT**

<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No.</th>
</tr>
</thead>
</table>

[X] Further documents are listed in the continuation of Box C.  
[X] See patent family annex.

* Special categories of cited documents:

- "A" document defining the general state of the art which is not considered to be of particular relevance
- "E" earlier application or patent but published on or after the international filing date
- "L" document which may throw doubts on priority claim(s) on which the later document is based
- "O" document referring to an oral disclosure, use, exhibition or other means
- "P" document published prior to the international filing date but later than the priority date claimed

"*" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered to be novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"S" document member of the same patent family

Date of the actual completion of the international search: 13 September 2016

Date of mailing of the international search report: 21/09/2016

Name and mailing address of the ISA:

European Patent Office, P.B. 5818 Patentlaan 2  
NL - 2280 HV Rijswijk  
Tel. (+31-70) 340-2040,  
Fax: (+31-70) 340-3016

Authorized officer: Le Gillet, Ronan
<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>paragraph [0050] - paragraph [0055]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>paragraph [0069] - paragraph [0074]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>figure 1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-----</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>US 2012/141042 A1 (IKETANI AKIHiko [JP])</td>
<td>1-15</td>
</tr>
<tr>
<td></td>
<td>7 June 2012 (2012-06-07)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>paragraph [0010] - paragraph [0012]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>paragraph [0057] - paragraph [0060]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>claims 1-4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-----</td>
<td></td>
</tr>
<tr>
<td></td>
<td>paragraph [0010] - paragraph [0013]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>paragraph [0047] - paragraph [0056]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-----</td>
<td></td>
</tr>
<tr>
<td></td>
<td>paragraph [0116] - paragraph [0126]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>paragraph [0170] - paragraph [0176]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-----</td>
<td></td>
</tr>
<tr>
<td></td>
<td>22 February 2007 (2007-02-22)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>paragraph [0033] - paragraph [0054]</td>
<td></td>
</tr>
<tr>
<td>Patent document cited in search report</td>
<td>Publication date</td>
<td>Patent family member(s)</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>-----------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2016134907 AI</td>
</tr>
<tr>
<td>US 2005060421 AI</td>
<td>17-03-2005</td>
<td>NONE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>JP 2005525056 A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2003210744 AI</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Wo 03096700 AI</td>
</tr>
<tr>
<td>US 2012141042 AI</td>
<td>07-06-2012</td>
<td>US 2012141042 AI</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Wo 2011018878 AI</td>
</tr>
<tr>
<td>US 2005213833 AI</td>
<td>29-09-2005</td>
<td>CN 1678070 A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>JP 2005286472 A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2005213833 AI</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2003185456 AI</td>
</tr>
<tr>
<td>US 2007041664 AI</td>
<td>22-02-2007</td>
<td>JP 4600209 B2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>JP 2007052672 A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2007041664 AI</td>
</tr>
</tbody>
</table>