(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
22 March 2018 (22.03.2018)

W ! P O | PCT

(10) International Publication Number
WO 2018/050249 Al

(51) International Patent Classification:
G06F 3/01 (2006.01)

(21) International Application Number:
PCT/EP20 16/072 137

(22) International Filing Date:
19 September 2016 (19.09.2016)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **TELEFONAKTIEBOLAGET LM ERICSSON (PUBL)** [SE/SE]; SE-164 83 Stockholm (SE).

(72) Inventors: **GRANCHAROV, Volodya;** Ericsson AB, Farogatan 6, 16480 Stockholm (SE). **ANDERSSON, Lars;** Ericsson AB, Farogatan 6, 16480 Stockholm (SE). **ARAÚJO, Jose;** Ericsson AB, Farogatan 6, 16480 Stockholm (SE).

(74) Agent: **ERICSSON;** Torshamnsgatan21-23, 164 80 Stockholm (SE).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available):* AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available):* ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report (Art. 21(3))*

(54) Title: ENCODING AND DECODING MULTICHANNEL HAPTIC DATA BY DETERMINING AN ORDER OF A PLURALITY OF CHANNELS

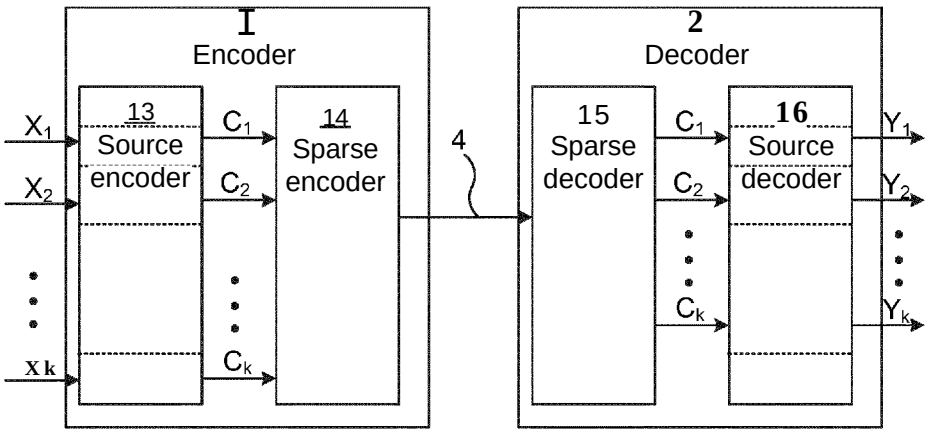

Fig. 2

(57) Abstract: It is presented a method for encoding haptic data for a plurality of channels. The method comprises the steps of: determining an order of the plurality of channels in a first list based on a respective previous codeword data item for each of the plurality of channels, each previous codeword data item being either a no-change data item, or a change data item, wherein the order is such that all no-change previous codeword data items are arranged at one end of the first list; arranging current codewords in a second list accordance with the channel order of the first list; grouping the current codewords in consecutive groups; generating a group index indicating whether each group contains only no-change codewords or at least one change codeword; and outputting the group index along with current codewords, wherein current codewords are only output for groups containing at least one change codeword.

WO 2018/050249 A1

ENCODING AND DECODING MULTICHANNEL HAPTIC DATA BY DETERMINING AN ORDER OF A PLURALITY OF CHANNELS

TECHNICAL FIELD

The invention relates to methods, encoders, decoders, computer programs, and computer program products relating to encoding and decoding haptic data for a plurality of channels.

BACKGROUND

Haptic internet is seen as the next step on mobile networking. Users are currently able to efficiently communicate over voice and video, but in the future networked society, it is envisioned that people will be able to communicate the sense of touch via haptic devices. There has been a large amount of research on devices, which allow such communication to take place. Several haptic interfaces are now appearing in the market, which deliver haptic feedback using different actuation technologies such as ultrasound, vibrotactile, electrostatic and piezoelectric.

Haptic perception consists of both kinesthetic and tactile sense and relates to the sensation of the size, shape, mass, texture and stiffness of physical objects, surfaces, etc. Kinesthetic information refers to the information perceived when moving joints, muscles and tendons, while tactile information refers to information retrieved via the skin. In a haptic system, the amount of data representing all the sensors quickly becomes large. Hence, data compression is needed to keep the data transfer requirements at a reasonable level. However, the data compression and decompression can introduce delays. Since any delay can seriously reduce the experience of the user, any compression and decompression which is used should be designed such that the amount of delay which is introduced is as small as possible.

R. Chaudhariu, C. Schuwerk, M. Danaei, and E. Steinbach, "Perceptual and Bitrate-scalable Coding of Haptic Surface Texture Signals," ***IEEE J. Selected Topics in Signal Processing***, vol. 9, no. 3, pp. 462-473, 2015, discloses a bitrate scalable haptic texture codec, which incorporates a masking model

based on weaker signals close in frequency to a powerful signal being masked.

SUMMARY

It is an object to further reduce bitrate requirements for haptic data.

According to a first aspect, it is presented a method for encoding haptic data for a plurality of channels. The method is performed by an encoder and comprises the steps of: determining an order of the plurality of channels in a first list based on a respective previous codeword data item for each of the plurality of channels, each previous codeword data item being either a no-change data item indicating no change of channel data, or a change data item indicating a change of channel data, wherein the order is such that all no-change previous codeword data items are arranged at one end of the first list; arranging current codewords in a second list accordance with the channel order of the first list; grouping the current codewords in consecutive groups based on the order of the current codewords in the second list; wherein each group contains a plurality of current codewords; generating a group index indicating whether each group contains only no-change codewords or at least one change codeword; and outputting the group index along with current codewords, wherein current codewords are only output for groups containing at least one change codeword.

The step of determining an order may comprise the sub-steps of: obtaining a respective previous codeword data item for each of the plurality of channels, the previous codeword data item being ordered by channels in a first list, each previous codeword data item being based on at least one previous codeword for the respective channel; setting a first pointer on the first previous codeword data item, in the first list, which is a no-change data item; setting a second pointer on the last previous codeword data item, in the first list, which is a change data item; swapping the previous codeword data item pointed to by the first pointer and the previous codeword data item pointed to by the second pointer, when the first pointer is before the second pointer in the first list, wherein each previous codeword data item keeps a channel

reference during the swapping; and repeating the steps of setting a first pointer, setting a second pointer and swapping until the first pointer is after the second pointer in the first list.

The no-change codeword may consist of one or more zeros.

The group index may comprise an indicator for each group, each indicator indicating whether the respective group contains only no-change data items or at least one change data item.

Each previous codeword data item may be the most recent previous codeword for the channel in question.

Each previous codeword data item may be derived based on a plurality of previous codewords for the channel in question.

According to a second aspect, it is presented an encoder for encoding haptic data for a plurality of channels. The encoder comprises: a processor storing instructions that, when executed by the processor, causes the encoder to: determine an order of the plurality of channels in a first list based on a respective previous codeword data item for each of the plurality of channels, each previous codeword data item being either a no-change data item indicating no change of channel data, or a change data item indicating a change of channel data, wherein the order is such that all no-change previous codeword data items are arranged at one end of the first list; arrange current codewords in a second list accordance with the channel order of the first list; group the current codewords in consecutive groups based on the order of the current codewords in the second list; wherein each group contains a plurality of current codewords; generate a group index indicating whether each group contains only no-change codewords or at least one change codeword; and output the group index along with current codewords, wherein current codewords are only output for groups containing at least one change codeword. encoder

The instructions to determine an order may comprise instructions to: obtain a respective previous codeword data item for each of the plurality of channels, the previous codeword data item being ordered by channels in a first list, each previous codeword data item being based on at least one previous codeword for the respective channel; set a first pointer on the first previous codeword data item, in the first list, which is a no-change data item; set a second pointer on the last previous codeword data item, in the first list, which is a change data item; swap the previous codeword data item pointed to by the first pointer and the previous codeword data item pointed to by the second pointer, when the first pointer is before the second pointer in the first list, wherein each previous codeword data item keeps a channel reference during the swapping; and repeat the instructions to set a first pointer, set a second pointer and swap until the first pointer is after the second pointer in the first list.

The no-change codeword may consist of one or more zeros.

The group index may comprise an indicator for each group, each indicator indicating whether the respective group contains only no-change data items or at least one change data item.

Each previous codeword data item may be the most recent previous codeword for the channel in question.

Each previous codeword data item may be derived based on a plurality of previous codewords for the channel in question.

According to a third aspect, it is presented an encoder comprising: means for determining an order of a plurality of channels providing haptic data in a first list based on a respective previous codeword data item for each of the plurality of channels, each previous codeword data item being either a no-change data item indicating no change of channel data, or a change data item indicating a change of channel data, wherein the order is such that all no-change previous codeword data items are arranged at one end of the first list; means for arranging current codewords in a second list accordance with the

channel order of the first list; means for grouping the current codewords in consecutive groups based on the order of the current codewords in the second list; wherein each group contains a plurality of current codewords; means for generating a group index indicating whether each group contains only no-change codewords or at least one change codeword; and means for outputting the group index along with current codewords, wherein current codewords are only output for groups containing at least one change codeword.

According to a fourth aspect, it is presented a computer program for encoding haptic data for a plurality of channels. The computer program comprises computer program code which, when run on an encoder causes the encoder to: obtain a respective previous codeword data item for each of the plurality of channels, the previous codeword data item being ordered by channels in a first list, each previous codeword data item being based on at least one previous codeword for the respective channel, the previous codeword data item being either a no-change data item indicating no change of channel data, or a change data item indicating a change of channel data; set a first pointer on the first previous codeword data item, in the first list, which is a no-change data item; set a second pointer on the last previous codeword data item, in the first list, which is a change data item; swap the previous codeword data item pointed to by the first pointer and the previous codeword data item pointed to by the second pointer, when the first pointer is before the second pointer in the first list, wherein each previous codeword data item keeps a channel reference during the swapping; and repeat the instructions to set a first pointer, set a second pointer and swap until the first pointer is after the second pointer in the first list.

According to a fifth aspect, it is presented a computer program product comprising a computer program according to the fourth aspect and a computer readable means on which the computer program is stored.

According to a sixth aspect, it is presented a method for decoding haptic data for a plurality of channels. The method is performed by a decoder and

comprises the steps of: determining an order of the plurality of channels in a first list based on a respective previous codeword data item for each of the plurality of channels, each previous codeword data item being either a no-change data item indicating no change of channel data, or a change data item indicating a change of channel data, wherein the order is such that all no-change previous codeword data items are arranged at one end of the first list; receiving a group index and current codewords for groups of channels of the plurality of channels; generating no-change codewords for channels in any group of channels indicated by the group index to contain only no-change codewords; combining the generated codewords with the received codewords, yielding current codewords; arranging the current codewords in a second list accordance with the channel order of the first list; reordering the second list by channels; and outputting the reordered second list.

The step of determining an order may comprise the sub-steps of: obtaining a respective previous codeword data item for each of the plurality of channels, the previous codeword data items being ordered by channels in a first list, each previous codeword data item being based on at least one previous codeword for the respective channel; setting a first pointer on the first previous codeword data item, in the first list, which is a no-change data item; setting a second pointer on the last previous codeword data item, in the first list, which is a change data item; swapping the previous codeword data item pointed to by the first pointer and the previous codeword data item pointed to by the second pointer, when the first pointer is before the second pointer in the first list, wherein each previous codeword data item keeps a channel reference during the swapping; and repeating the steps of setting a first pointer, setting a second pointer and swapping until the first pointer is after the second pointer in the first list.

The no-change codeword may consist of one or more zeros.

The group index may comprise an indicator for each group, each indicator indicating whether the respective group contains only no-change data items or at least one change data item.

Each previous codeword data item may be the most recent previous codeword for the channel in question.

Each previous codeword data item may be derived based on a plurality of previous codewords for the channel in question.

According to a seventh aspect, it is presented a decoder for decoding haptic data for a plurality of channels. The decoder comprises: a processor storing instructions that, when executed by the processor, causes the decoder to: determine an order of the plurality of channels in a first list based on a respective previous codeword data item for each of the plurality of channels, each previous codeword data item being either a no-change data item indicating no change of channel data, or a change data item indicating a change of channel data, wherein the order is such that all no-change previous codeword data items are arranged at one end of the first list; receive a group index and current codewords for groups of channels of the plurality of channels; generate no-change codewords for channels in any group of channels indicated by the group index to contain only no-change codewords; combine the generated codewords with the received codewords, yielding current codewords; arrange the current codewords in a second list accordance with the channel order of the first list; reorder the second list by channels; and output the reordered second list.

The instructions to determine an order may comprise instructions to: obtain a respective previous codeword data item for each of the plurality of channels, the previous codeword data items being ordered by channels in a first list, each previous codeword data item being based on at least one previous codeword for the respective channel; set a first pointer on the first previous codeword data item, in the first list, which is a no-change data item; set a second pointer on the last previous codeword data item, in the first list, which is a change data item; swap the previous codeword data item pointed to by the first pointer and the previous codeword data item pointed to by the second pointer, when the first pointer is before the second pointer in the first list, wherein each previous codeword data item keeps a channel reference

during the swapping; and repeat the instructions to set a first pointer, set a second pointer and swap until the first pointer is after the second pointer in the first list.

The no-change codeword may consist of one or more zeros.

The group index may comprise an indicator for each group, each indicator indicating whether the respective group contains only no-change data items or at least one change data item.

Each previous codeword data item may be the most recent previous codeword for the channel in question.

Each previous codeword data item may be derived based on a plurality of previous codewords for the channel in question.

According to an eighth aspect, it is presented a decoder comprising: means for determining an order of a plurality of channels, for providing haptic data, in a first list based on a respective previous codeword data item for each of the plurality of channels, each previous codeword data item being either a no-change data item indicating no change of channel data, or a change data item indicating a change of channel data, wherein the order is such that all no-change previous codeword data items are arranged at one end of the first list; means for receiving a group index and current codewords for groups of channels of the plurality of channels; means for generating no-change codewords for channels in any group of channels indicated by the group index to contain only no-change codewords; means for combining the generated codewords with the received codewords, yielding current codewords; means for arranging the current codewords in a second list accordance with the channel order of the first list; means for reordering the second list by channels; and means for outputting the reordered second list.

According to a ninth aspect, it is presented a computer program for decoding haptic data for a plurality of channels, the computer program comprising computer program code which, when run on a decoder causes the decoder to:

determine an order of the plurality of channels in a first list based on a respective previous codeword data item for each of the plurality of channels, each previous codeword data item being either a no-change data item indicating no change of channel data, or a change data item indicating a change of channel data, wherein the order is such that all no-change previous codeword data items are arranged at one end of the first list; receive a group index and current codewords for groups of channels of the plurality of channels; generate no-change codewords for channels in any group of channels indicated by the group index to contain only no-change codewords; combine the generated codewords with the received codewords, yielding current codewords; arrange the current codewords in a second list accordance with the channel order of the first list; reorder the second list by channels; and output the reordered second list.

According to a tenth aspect, it is presented a computer program product comprising a computer program according to the ninth aspect and a computer readable means on which the computer program is stored.

Generally, all terms used in the claims are to be interpreted according to their ordinary meaning in the technical field, unless explicitly defined otherwise herein. All references to "a/an/the element, apparatus, component, means, step, etc." are to be interpreted openly as referring to at least one instance of the element, apparatus, component, means, step, etc., unless explicitly stated otherwise. The steps of any method disclosed herein do not have to be performed in the exact order disclosed, unless explicitly stated.

BRIEF DESCRIPTION OF THE DRAWINGS

The invention is now described, by way of example, with reference to the accompanying drawings, in which:

Fig 1 is a schematic diagram illustrating an environment in which embodiments presented herein can be applied;

Fig 2 is a schematic diagram illustrating details of the encoder and decoder of Fig 1 according to one embodiment;

Figs 3A-B are flow charts illustrating embodiments of methods for encoding haptic data for a plurality of channels performed in an encoder;

Figs 4A-B are flow charts illustrating embodiments of methods for decoding haptic data for a plurality of channels performed in a decoder;

Fig 5 is a schematic diagram illustrating components of the encoder and decoder of Fig 1, according to one embodiment;

Fig 6 is a schematic diagram showing functional modules of the encoder of Fig 1 according to one embodiment;

Fig 7 is a schematic diagram showing functional modules of the decoder of Fig 1 according to one embodiment; and

Fig 8 shows one example of a computer program product comprising computer readable means.

DETAILED DESCRIPTION

The invention will now be described more fully hereinafter with reference to the accompanying drawings, in which certain embodiments of the invention are shown. This invention may, however, be embodied in many different forms and should not be construed as limited to the embodiments set forth herein; rather, these embodiments are provided by way of example so that this disclosure will be thorough and complete, and will fully convey the scope of the invention to those skilled in the art. Like numbers refer to like elements throughout the description.

Fig 1 is a schematic diagram illustrating an environment in which embodiments presented herein can be applied. A user 9 is here connected to interact with a remote device 6. The remote device 6 comprises a number of sensors 5. Although not shown, the remote device 6 also comprises one or more actuators which can be controlled by the user 9. The sensors 5 generate data which is input to an encoder 1. Since there are multiple sensors 5, the

data input to the encoder 1 is multichannel data, where each channel is associated with a separate set of one or more sensors.

The encoder 1 compresses the multichannel data to encoded data 4 and outputs the data for transmission to a decoder 2. The transmission can occur using any communication technology known in the art per se, e.g. over a Local Area Network (LAN), such as Ethernet and/or using a Wide Area Network (WAN), e.g. using Internet Protocol (IP). In any case, the encoded data 4 is received by the decoder 2, which decompresses the encoded data 4 to signals which are suitable for provision to actuators 10 of a user device 8. It is to be noted that the term 'remote' in this context does not need to be far away. 'Remote' only implies that there is communication between the user device 8 and the remote device 6.

Hence, the actuators 10 of the user device 8 provide haptic sensations reflecting what is detected by the sensors 5. This allows a user 9 of the user device 8 to haptically experience a physical environment which can be located remotely, which can be in another room or in another country from the user 9.

Optionally, the equivalent communication occurs in the other direction from the master (the user device 8) to the slave (the remote device 6). In other words, there is may also be an encoder 1 by the user device 8 and a decoder 2 by the remote device 6, to thereby communicate haptic data from sensors (not shown) in the user device 8 to control actuators (not shown) in the remote device 6.

While the user device 8 is here shown as a glove where the user can insert a hand in the user device 8, the user device 8 can be provided in any suitable way that allows haptic perception for the user 9. For instance, the user device 8 could be an exoskeleton, etc. In any case, the remote device 6 provides sensors in a way that makes sense in terms of the physical appearance of the user device 8.

To illustrate the requirements for such a multichannel system, one can consider a teleoperation scenario where an operator uses an exoskeleton (master device), which is capable of providing both tactile and kinesthetic feedback to his/her hands and his/her forearm and upper arms, to control a slave device (e.g. a robot).

In order to perform the teleoperation, the measured values (e.g. position, acceleration, and/force force) at the master are communicated to the control unit at the slave side, while the measured values at the slave side are communicated to the control unit at the master side. With such a setup, the operator is able to perform the precise control of the robotic arms, while he/she is also provided with tactile feedback.

This setup possesses three data channels (position, acceleration and force) per sensor measurement per location. As a possible implementation, one may consider that there exists one acquisition point for each finger, one for the wrist, and one for the forearm and one for the upper arm, resulting in a total of 13 acquisition points.

This would result in a total number of sensors of 39 (13*3 data channels (position, acceleration and force)), where each sensor has three dimensions, resulting in 117 (13*3*3) data channels. If each data channel requires 32 Kbps (kilobits per second), the total bandwidth needed can be calculated as 32 Kbps * 117 = 3.744 Mbps. Hence, it is evident that an efficient compression system is of great benefit to enable this type of haptic application with a reasonable data rate requirement.

Teleoperation systems with haptic feedback may become unstable if the closed-loop system experiences latency larger than tens of milliseconds. Additionally, research has shown than for latency above 10 ms, the teleoperation transparency is greatly reduced, affecting both the perception quality as well as the task performance. Hence, the compression system should not introduce further algorithmic delay. This implies that block

processing is not allowed, preventing conventional multichannel compression schemes from being applied.

Fig 2 is a schematic diagram illustrating details of the encoder and decoder of Fig 1 according to one embodiment. There are here k channels, resulting in k samples x_{i-xk} representing haptic data. The samples x_{i-xk} are received in parallel from the different sensors connected to the encoder. The samples x_{i-xk} are here in digital form, whereby the sensor data has previously been quantized, which can occur prior to the encoder 1 or as part of the encoder 1 (not shown). The encoder 1 comprises a source encoder 13 and a sparse encoder 14. The source encoder 13 encodes, using lossy encoding, the samples x_{i-xk} into an equivalent number of codewords c_{i-ck}. The sparse encoder 14 then converts the codewords c_{i-ck} in a lossless way to an encoded data stream 4 which is communicated to the decoder as explained above with reference to Fig 1.

The decoder 2, on its side, comprises a sparse decoder 15 and a source decoder 16. The sparse decoder converts the data stream 4 to the codewords c_{i-ck}, which are the same as the codewords c_{i-ck} in the encoder 1 since the sparse encoder 14 employs a lossless algorithm. It is to be noted that transmission errors between the encoder 1 and decoder 2 are not considered here and are dealt with as known in the art per se. The source decoder 16 then converts the codewords c_{i-ck} to reconstructed sample values y_{i-yk} which can be provided to actuators of the user device 8 to achieve a haptic sensation for the user.

The details of how the encoding and decoding is performed will be explained below with reference to Figs 3A-B (for the encoder) and Figs 4A-B (for the decoder).

Figs 3A-B are flow charts illustrating embodiments of methods for encoding haptic data for a plurality of channels. Prior to this method, the current codewords have been generated, e.g. by means of a lossy encoding which can occur in the source encoder (13 of Fig 2).

In a *determine order* step 40, an order of the plurality of channels in a first list is determined based on a respective previous codeword data item for each of the plurality of channels. Each previous codeword data item is either a no-change data item indicating no change of channel data, or a change data item indicating a change of channel data. The order in the first list is such that all no-change previous codeword data items are arranged at one end of the first list. The encoder maintains an association between each previous codeword data item and its channel. The no-change codeword data items can be arranged at either end of the first list.

The no-change codeword can consist of one or more zeros. However, it is equally possible that the no-change codewords can be represented by any suitable predetermined set of one or more bits which are known to both the encoder and the decoder. In one embodiment, each previous codeword data item is the most recent previous codeword for the channel in question. In one embodiment, each previous codeword data item is derived based on a plurality of previous codewords for the channel in question. For instance, each codeword data item can be based on a sliding window of the last x number of previous codewords for the channel in question. In such an embodiment, the weights of the previous codewords forming part of the sliding window can be identical or weights gradually decrease for older previous codewords.

In an *arrange current codewords* step 46, current codewords are arranged in a second list accordance with the channel order of the first list. In other words, the current codewords are arranged such that their respective associated channels are in the same order as the channels of the first list.

In a *group* step 47, the current codewords are grouped in consecutive groups based on the order of the current codewords in the second list. Each group contains a plurality of current codewords.

In a *generate group index* step 48, a group index is generated. The group index indicates whether each group contains only no-change codewords or at

least one change codeword. In one embodiment, the group index comprises an indicator for each group, each indicator indicating whether the respective group contains only no-change codewords or at least one change codeword indicating a change. In one embodiment, the group index consists of identifiers of those groups which contain at least one change codeword indicating a change. All other groups are then groups containing only no-change codewords. The complement is also possible; the group index then consists of identifiers of those groups which only contain no-change codewords.

In an *output* step 49, the group index is output along with current codewords, wherein current codewords are only output for groups containing at least one change codeword. In other words, no codewords for groups containing only no-change codewords are output. The data that is output is then transmitted to the decoder.

Looking now to Fig 3B, only new or modified steps will be described.

The *determine order* step 40 here comprises a number of optional substeps 41-45.

In an optional *obtain codeword data items* step 41, a respective previous codeword data item is obtained for each of the plurality of channels. The previous codeword data item is ordered by channels in a first list. Each previous codeword data item is based on at least one previous codeword for the respective channel.

In an optional *setfirst pointer* step 42, a first pointer is set on the first previous codeword data item, in the first list, which is a no-change data item.

In an optional *set second pointer* step 43, a second pointer is set on the last previous codeword data item, in the first list, which is a change data item.

In an optional conditional *first pointer before second pointer* step 45, it is determined whether the first pointer is before the second pointer, in the first

list. If this is the case, the method proceeds to an optional *swap* step 44. Otherwise, the method proceeds to the *arrange current codewords* step 46.

In the optional *swap* step 44, the previous codeword data item pointed to by the first pointer and the previous codeword data item pointed to by the second pointer are swapped. In the swapping, the encoder maintains an association between each previous codeword data item and a channel.

Figs 4A-B are flow charts illustrating embodiments of methods for decoding haptic data for a plurality of channels performed in a decoder.

In a *determine order* step 50, an order of the plurality of channels in a first list is determined based on a respective previous codeword data item for each of the plurality of channels. Each previous codeword data item is either a no-change data item indicating no change of channel data, or a change data item indicating a change of channel data. The order of the first list is such that all no-change previous codeword data items are arranged at one end of the first list. In one embodiment, a no-change codeword indicates a difference between samples which is lower than a perceptual threshold. For instance, the no-change codeword can consist of one or more zeros. In one embodiment, each previous codeword data item can is the most recent previous codeword for the channel in question. In one embodiment, each previous codeword data item is derived based on a plurality of previous codewords for the channel in question.

In a *receive data* step 56, a group index and current codewords for groups of channels of the plurality of channels are received. In one embodiment, the group index comprises an indicator for each group, each indicator indicating whether the respective group contains only no-change codewords or at least one change codeword indicating a change. In one embodiment, the group index consists of identifiers of those groups which contain at least one change codeword indicating a change. All other groups are then groups containing only no-change codewords. The complement is also possible; the group index

then consists of identifiers of those groups which only contain no-change codewords.

In a *generate no change data items* step 57, no-change codewords are generated for channels in any group of channels indicated by the group index to contain only no-change codewords.

In a *combine* step 58, the generated codewords are combined with the received codewords, yielding current codewords.

In an *arrange current codewords* step 59, the current codewords are arranged in a second list accordance with the channel order of the first list.

In a *reorder* step 60, the second list is reordered by channels.

In an *output* step 61, the reordered second list is output.

Looking now to Fig 4B, only new or modified steps will be described.

The *determine order* step 50 here comprises a number of optional substeps 51-55, corresponding to steps 41-45 of Fig 3B.

In an optional *obtain codeword data items* step 51, a respective previous codeword data item is obtained for each of the plurality of channels. The previous codeword data item is ordered by channels in a first list. Each previous codeword data item is based on at least one previous codeword for the respective channel.

In an optional *setfirst pointer* step 52, a first pointer is set on the first previous codeword data item, in the first list, which is a no-change data item.

In an optional *set second pointer* step 53, a second pointer is set on the last previous codeword data item, in the first list, which is a change data item.

In an optional conditional *first pointer before second pointer* step 55, it is determined whether the first pointer is before the second pointer in the first

list. If this is the case, the method proceeds to an optional *swap* step 54. Otherwise, the method proceeds to the *receive data* step 56.

In the optional *swap* step 54, the previous codeword data item pointed to by the first pointer and the previous codeword data item pointed to by the second pointer are swapped. In the swapping, the decoder maintains an association between each previous codeword data item and a channel.

Using the methods of Figs 3A-B and Figs 4A-B, the encoding and decoding can occur without introducing any significant delay, which would be the case e.g. if block processing is utilised.

The presented embodiment reduces the size of the bitstream by more efficient bit packetization. The embodiments can operate directly on input haptic data, or on the output of a source coding (lossy compression) scheme, which would then operate as a pre-processor to the presented scheme.

The re-ordering of channels in these embodiments is built on the observation from statistics that haptic data (sampled at lkHz) do not vary rapidly, and therefore optimal re-shuffling derived from the last encoded sample vector could be applied to the current input.

The compression system is mostly efficient when a significant number of sensors are inactive or the amplitude change from sample to sample is below the perceptual threshold. This is a typical situation in haptic devices.

An embodiment of encoding and decoding, employed by the methods of Figs 3A-B and Figs 4A-B, will now be described using a few examples.

In these examples, the codewords discussed below only assumes the values 1 or o. In reality, the codewords could be the output of n-bit quantizer, where in such case, we assume that 1is assigned to all codewords that contain at least one non-zero bit, while o is assigned to codewords that contain all-zero bits. As an example in 2-bit quantization, {oo}->o, while {01, 10, n}->i .

Now a first example will be described where the reordering does not occur. This will later be compared with a second example where reordering does occur, thereby illustrating the benefits of the reordering.

In the first example, at a time instance ***t,*** the sparse encoder (14 of Fig 2) of the encoder receives a set of ***k*** codewords, $c_{i,} \ldots_{,Ck,}$ which correspond to quantization levels of ***k*** channels, and groups them into ***k/N*** groups with ***N*** codewords in each group. For each group, an indicator is calculated, which indicates whether the group has any change codewords. These indicators together form the group index which is sent to the decoder, together with the codewords for groups that have at least one change codeword.

In this example, the current codewords provided to the encoder comprises ***k=i2*** codewords and each group contains ***N=3*** channels, shown in (1)

{010, 010, 000, 001} (1)

where commas define group boundaries. According to the description above the sparse encoder will generate a bitstream shown in (2)

{1101, 010, 010, 001} (2)

where the first four bits, in bold, is the group index (one indicator per group). The third group in the input bitstream (1) contains only zeros, indicating no change, and is therefore omitted from the output of the encoder. This will be detected by the decoder and the decoder can reconstruct the original bitstream by generating no-change codewords where the group index indicates a complete no-change group, in this case for the third group only.

However, as illustrated here, the input bitstream (1) consists of twelve bits, while the output bitstream (2) consists of thirteen bits. In other words, instead of compression, the encoder actually increases the number of bits required by encoding the data.

This leads to the second example, in accordance with embodiments herein, where reordering is performed to overcome this problem. In this way, a more

efficient grouping of codewords is achieved, which leads to more efficient compression, since more groups contain only no-change codewords.

In the second example, we assume that the codewords in (1) is the most recent previous set of codewords. Here, for each channel, the previous codeword data item is the most recent previous codeword. As shown below, the most recent previous set of codewords is used in step 40 to determine the order of channels. In this example, the algorithm of Fig 3B is employed.

In step 41, the previous codeword data items are obtained, in this case the codewords in (1).

In step 42, the first pointer Pi is set on the first previous codeword, in the first list, which is a no-change data item. In this example, Pi is set on channel 1.

In step 43, a second pointer P2 is set on the last previous codeword data item, in the first list, which is a change data item. In this example, P2 is set on channel 12.

At this point, the state in the encoder is reflected in Table 1.

Ch	**Prv cw**	**P:s**
1	0	<-Pi
2	1	
3	0	
4	0	
5	1	
6	0	
7	0	
8	0	
9	0	
10	0	
11	0	

12	1	<-P2

Table 1: Original encoder state

Ch denoted the channel number, *Prv cw* denotes the previous codeword and *P:s* show the pointers.

Since Pi is before P2 (step 45), a swap (step 44) is performed. Hence, channel 1 and channel 12 are swapped (including the *Prv cw).* Setting the pointers again (steps 42 and 43) results in an encoder state according to Table 2.

Ch	**Prv cw**	**P:s**
12	1	
2	1	
3	0	<-P1
4	0	
5	1	<-P2
6	0	
7	0	
8	0	
9	0	
10	0	
11	0	
1	0	

Table 2: Encoder state after one iteration

Since Pi is still before P2 (step 45), a swap (step 44) is performed again. Hence, channel 3 and channel 5 are swapped (including the Prv cw). Setting the pointers again (steps 42 and 43) results in an encoder state according to Table 3.

Ch	**Prv cw**	**P:s**
12	1	
2	1	
5	1	<-P2

4	0	<-Pi
3	0	
6	0	
7	0	
8	0	
9	0	
10	0	
11	0	
1	0	

Table 3: Encoder state after two iterations

Since Pi is now after P2 (step 45), the current codewords can be added. In this example, the current codewords are represented by the following set of codewords:

{010, 010, 001, 001} (3)

The current codewords are almost the same as the previous codeword; only channel 9 has changed from a '0' to a '1'. The current codewords are thus added (step 46) at the correct lines (based on channel number). Moreover, the grouping (step 47) is done, which then results in an encoder state as shown in Table 4:

Ch	**Curr. Cw**	**Prv cw**	**Group**
12	1	1	1
2	1	1	1
5	1	1	1
4	0	0	II
3	0	0	II
6	0	0	II
7	0	0	III
8	0	0	III
9	1	0	III

10	0	0	IV
11	0	0	IV
1	0	0	IV

Table 4: EIncoder state with current codewords and grouping

where *Curr. cw* denotes the current codeword. Hence, resulting set of data in the groups is represented by (4), i.e. the current codewords in the order of Table 4:

{111, 000, 001, 000} (4)

Note that the current codeword differs from the previous codeword for channel 9.

In step 48, the group index is generated, which becomes {1010}. The resulting encoded data which is output (step 49) is then shown in (5):

{1010, 111, 001} (5)

Hence, there is a data reduction from twelve bits in (3) to ten bits in (5). Note that if there had been no data change in channel 9, the encoding would have been even more efficient since then the codewords of group III could also have been omitted..

In the decoder, steps 51-55 correspond to steps 41-45. Since the decoder also works on the same previous data items as the encoder, the decoder also works out Table 3.

Once the reordering is determined, the data (5) is received from the encoder (step 56). The no-change codewords are then generated in step 57, and combined in step 58 with the received data, resulting in the data of (4). Note that the steps 56, 57 and 58 could also be performed prior to, or in parallel with, the reordering of steps 51-55.

The combined codewords are arranged in step 59 in accordance with the reordered channels (based on Table 3). Hence, the decoder is then in a state according to Table 5:

Ch	Curr. cw
12	1
2	1
5	1
4	0
3	0
6	0
7	0
8	0
9	1
10	0
11	0
1	0

Table 5: Decoder state based on reordering

This table is then sorted according to channel number in the reorder step 60. This results in the decoder state according to Table 6.

Ch	Curr. cw
1	0
2	1
3	0
4	0
5	1
6	0
7	0
8	0
9	1
10	0

11	0
12	1

Table 6: End state at the decoder

The current codewords of Table 6 can be expressed in a more condensed form as (6), which is identical to (3), whereby the decoding is completed.

{010, 010, 001, 001} (6)

Since the reordering is the same in both the encoder and the decoder, and is based on the same (previous codeword data), the reordering does not need to be explicitly provided from the encoder to the decoder. This allows the encoder to compress signals efficiently while still allowing the decoder to restore the data without any loss.

Fig 5 is a schematic diagram illustrating components of the encoder 1 and decoder 2 of Fig 1, according to one embodiment. Each of the encoder 1 and the decoder 2 comprises the components described here. A processor 60 is provided using any combination of one or more of a suitable central processing unit (CPU), multiprocessor, microcontroller, digital signal processor (DSP), application specific integrated circuit etc., capable of executing software instructions 67 stored in a memory 64, which can thus be a computer program product. The processor 60 can be configured to execute the method described with reference to Fig 3 and 4 above for the encoder 1 and the decoder 2, respectively.

The memory 64 can be any combination of read and write memory (RAM) and read only memory (ROM). The memory 64 also comprises persistent storage, which, for example, can be any single one or combination of magnetic memory, optical memory, solid state memory or even remotely mounted memory.

A data memory 66 is also provided for reading and/or storing data during execution of software instructions in the processor 60, e.g. the current codewords and previous codeword data. The data memory 66 can be any

combination of read and write memory (RAM) and read only memory (ROM).

The encoder i/decoder 2 further comprises an I/O interface 62 for communicating with other external entities. Optionally, the I/O interface 62 also includes a user interface.

Fig 6 is a schematic diagram showing functional modules of the encoder 1 of Fig 1 according to one embodiment. The modules are implemented using software instructions such as a computer program executing in the encoder 1. Alternatively or additionally, the modules are implemented using hardware, such as any one or more of an ASIC (Application Specific Integrated Circuit), an FPGA (Field Programmable Gate Array), or discrete logical circuits. The modules correspond to the steps in the methods illustrated in Figs 3A-B.

An obtainer 70 corresponds to step 41. A pointer setter 71 corresponds to steps 42 and 43. A swapper 72 corresponds to step 44. A pointer determiner 73 corresponds to step 45. A codeword arranger 74 corresponds to step 46. A grouper 75 corresponds to step 47. An index generator 76 corresponds to step 48. An outputter 77 corresponds to step 49. An order determiner 78 corresponds to step 40.

Fig 7 is a schematic diagram showing functional modules of the decoder 2 of Fig 1 according to one embodiment. The modules are implemented using software instructions such as a computer program executing in the decoder 2. Alternatively or additionally, the modules are implemented using hardware, such as any one or more of an ASIC (Application Specific Integrated Circuit), an FPGA (Field Programmable Gate Array), or discrete logical circuits. The modules correspond to the steps in the methods illustrated in Figs 4A-B.

An obtainer 80 corresponds to step 51. A pointer setter 81 corresponds to steps 52 and 53. A swapper 82 corresponds to step 54. A pointer determiner 83 corresponds to step 55. A receiver 84 corresponds to step 56. A generator 85 corresponds to step 57. A combiner 86 corresponds to step 58. A codeword arranger 87 corresponds to step 59. A reorderer 88 corresponds to

step 60. An outputter 89 corresponds to step 61. An order determiner 95 corresponds to step 50.

Fig 8 shows one example of a computer program product comprising computer readable means. On this computer readable means a computer program 91 can be stored, which computer program can cause a processor to execute a method according to embodiments described herein. In this example, the computer program product is an optical disc, such as a CD (compact disc) or a DVD (digital versatile disc) or a Blu-Ray disc. As explained above, the computer program product could also be embodied in a memory of a device, such as the computer program product 164 of Fig 5. While the computer program 91 is here schematically shown as a track on the depicted optical disk, the computer program can be stored in any way which is suitable for the computer program product, such as a removable solid state memory, e.g. a Universal Serial Bus (USB) drive.

The invention has mainly been described above with reference to a few embodiments. However, as is readily appreciated by a person skilled in the art, other embodiments than the ones disclosed above are equally possible within the scope of the invention, as defined by the appended patent claims.

CLAIMS

1. A method for encoding haptic data for a plurality of channels, the method being performed by an encoder (1) and comprising the steps of:

determining (40) an order of the plurality of channels in a first list based on a respective previous codeword data item for each of the plurality of channels, each previous codeword data item being either a no-change data item indicating no change of channel data, or a change data item indicating a change of channel data, wherein the order is such that all no-change previous codeword data items are arranged at one end of the first list;

arranging (46) current codewords in a second list accordance with the channel order of the first list;

grouping (47) the current codewords in consecutive groups based on the order of the current codewords in the second list; wherein each group contains a plurality of current codewords;

generating (48) a group index indicating whether each group contains only no-change codewords or at least one change codeword; and

outputting (49) the group index along with current codewords, wherein current codewords are only output for groups containing at least one change codeword.

2. The method according to claim 1, wherein the step of determining (40) an order comprises the sub-steps of:

obtaining (41) a respective previous codeword data item for each of the plurality of channels, the previous codeword data item being ordered by channels in a first list, each previous codeword data item being based on at least one previous codeword for the respective channel;

setting (42) a first pointer on the first previous codeword data item, in the first list, which is a no-change data item;

setting (43) a second pointer on the last previous codeword data item, in the first list, which is a change data item;

swapping (44) the previous codeword data item pointed to by the first pointer and the previous codeword data item pointed to by the second pointer, when the first pointer is before the second pointer in the first list,

wherein each previous codeword data item keeps a channel reference during the swapping; and

repeating (45) the steps of setting (42) a first pointer, setting (43) a second pointer and swapping (44) until the first pointer is after the second pointer in the first list.

3. The method according to claim 1 or 2, wherein the no-change codeword consists of one or more zeros.

4. The method according to any one of the preceding claims, wherein the group index comprises an indicator for each group, each indicator indicating whether the respective group contains only no-change data items or at least one change data item.

5. The method according to any one of the preceding claims, wherein each previous codeword data item is the most recent previous codeword for the channel in question.

6. The method according to any one of claims 1 to 4, wherein each previous codeword data item is derived based on a plurality of previous codewords for the channel in question.

7. An encoder (1) for encoding haptic data for a plurality of channels, the encoder (1) comprising:

a processor (160); and

a memory (164) storing instructions (166) that, when executed by the processor, causes the encoder (1) to:

determine an order of the plurality of channels in a first list based on a respective previous codeword data item for each of the plurality of channels, each previous codeword data item being either a no-change data item indicating no change of channel data, or a change data item indicating a change of channel data, wherein the order is such that all no-change previous codeword data items are arranged at one end of the first list;

arrange current codewords in a second list accordance with the channel order of the first list;

group the current codewords in consecutive groups based on the order of the current codewords in the second list; wherein each group contains a plurality of current codewords;

generate a group index indicating whether each group contains only no-change codewords or at least one change codeword; and

output the group index along with current codewords, wherein current codewords are only output for groups containing at least one change codeword. encoder (1)

8. The encoder (1) according to claim 7, wherein the instructions to determine an order comprise instructions to:

obtain a respective previous codeword data item for each of the plurality of channels, the previous codeword data item being ordered by channels in a first list, each previous codeword data item being based on at least one previous codeword for the respective channel;

set a first pointer on the first previous codeword data item, in the first list, which is a no-change data item;

set a second pointer on the last previous codeword data item, in the first list, which is a change data item;

swap the previous codeword data item pointed to by the first pointer and the previous codeword data item pointed to by the second pointer, when the first pointer is before the second pointer in the first list, wherein each previous codeword data item keeps a channel reference during the swapping; and

repeat the instructions to set a first pointer, set a second pointer and swap until the first pointer is after the second pointer in the first list.

9. The encoder (1) according to claim 7 or 8, wherein the no-change codeword consists of one or more zeros.

10. The encoder (1) according to any one of claims 7 to 9, wherein the group index comprises an indicator for each group, each indicator indicating whether the respective group contains only no-change data items or at least one change data item.

11. The encoder (1) according to any one of claims 7 to 10, wherein each previous codeword data item is the most recent previous codeword for the channel in question.

12. The encoder (1) according to any one of claims 7 to 11, wherein each previous codeword data item is derived based on a plurality of previous codewords for the channel in question.

13. An encoder (1) comprising:

means for determining an order of a plurality of channels providing haptic data in a first list based on a respective previous codeword data item for each of the plurality of channels, each previous codeword data item being either a no-change data item indicating no change of channel data, or a change data item indicating a change of channel data, wherein the order is such that all no-change previous codeword data items are arranged at one end of the first list;

means for arranging current codewords in a second list accordance with the channel order of the first list;

means for grouping the current codewords in consecutive groups based on the order of the current codewords in the second list; wherein each group contains a plurality of current codewords;

means for generating a group index indicating whether each group contains only no-change codewords or at least one change codeword; and

means for outputting the group index along with current codewords, wherein current codewords are only output for groups containing at least one change codeword.

14. A computer program (167, 91) for encoding haptic data for a plurality of channels, the computer program comprising computer program code which, when run on an encoder (1) causes the encoder (1) to:

obtain a respective previous codeword data item for each of the plurality of channels, the previous codeword data item being ordered by channels in a first list, each previous codeword data item being based on at least one previous codeword for the respective channel, the previous codeword data

item being either a no-change data item indicating no change of channel data, or a change data item indicating a change of channel data;

set a first pointer on the first previous codeword data item, in the first list, which is a no-change data item;

set a second pointer on the last previous codeword data item, in the first list, which is a change data item;

swap the previous codeword data item pointed to by the first pointer and the previous codeword data item pointed to by the second pointer, when the first pointer is before the second pointer in the first list, wherein each previous codeword data item keeps a channel reference during the swapping; and

repeat the instructions to set a first pointer, set a second pointer and swap until the first pointer is after the second pointer in the first list.

15. A computer program product (164, 90) comprising a computer program according to claim 14 and a computer readable means on which the computer program is stored.

16. A method for decoding haptic data for a plurality of channels, the method being performed by a decoder (2) and comprising the steps of:

determining (50) an order of the plurality of channels in a first list based on a respective previous codeword data item for each of the plurality of channels, each previous codeword data item being either a no-change data item indicating no change of channel data, or a change data item indicating a change of channel data, wherein the order is such that all no-change previous codeword data items are arranged at one end of the first list;

receiving (56) a group index and current codewords for groups of channels of the plurality of channels;

generating (57) no-change codewords for channels in any group of channels indicated by the group index to contain only no-change codewords;

combining (58) the generated codewords with the received codewords, yielding current codewords;

arranging (59) the current codewords in a second list accordance with the channel order of the first list;

reordering (60) the second list by channels; and

outputting (61) the reordered second list.

17. The method according to claim 16, wherein the step of determining (50) an order comprises the sub-steps of:

obtaining (51) a respective previous codeword data item for each of the plurality of channels, the previous codeword data items being ordered by channels in a first list, each previous codeword data item being based on at least one previous codeword for the respective channel;

setting (52) a first pointer on the first previous codeword data item, in the first list, which is a no-change data item;

setting (53) a second pointer on the last previous codeword data item, in the first list, which is a change data item;

swapping (54) the previous codeword data item pointed to by the first pointer and the previous codeword data item pointed to by the second pointer, when the first pointer is before the second pointer in the first list, wherein each previous codeword data item keeps a channel reference during the swapping; and

repeating (55) the steps of setting (42) a first pointer, setting (43) a second pointer and swapping (44) until the first pointer is after the second pointer in the first list.

18. The method according to claim 16 or 17, wherein the no-change codeword consists of one or more zeros.

19. The method according to any one of claims 16 to 18, wherein the group index comprises an indicator for each group, each indicator indicating whether the respective group contains only no-change data items or at least one change data item.

20. The method according to any one of claims 16 to 19, wherein each previous codeword data item is the most recent previous codeword for the channel in question.

21. The method according to any one of claims 16 to 19, wherein each previous codeword data item is derived based on a plurality of previous codewords for the channel in question.

22. A decoder (2) for decoding haptic data for a plurality of channels, the decoder comprising:

a processor (160); and

a memory (164) storing instructions (166) that, when executed by the processor, causes the decoder (2) to:

determine an order of the plurality of channels in a first list based on a respective previous codeword data item for each of the plurality of channels, each previous codeword data item being either a no-change data item indicating no change of channel data, or a change data item indicating a change of channel data, wherein the order is such that all no-change previous codeword data items are arranged at one end of the first list;

receive a group index and current codewords for groups of channels of the plurality of channels;

generate no-change codewords for channels in any group of channels indicated by the group index to contain only no-change codewords;

combine the generated codewords with the received codewords, yielding current codewords;

arrange the current codewords in a second list accordance with the channel order of the first list;

reorder the second list by channels; and

output the reordered second list.

23. The decoder (2) according to claim 22, wherein the instructions to determine an order comprise instructions to:

obtain a respective previous codeword data item for each of the plurality of channels, the previous codeword data items being ordered by channels in a first list, each previous codeword data item being based on at least one previous codeword for the respective channel;

set a first pointer on the first previous codeword data item, in the first list, which is a no-change data item;

set a second pointer on the last previous codeword data item, in the first list, which is a change data item;

swap the previous codeword data item pointed to by the first pointer and the previous codeword data item pointed to by the second pointer, when the first pointer is before the second pointer in the first list, wherein each previous codeword data item keeps a channel reference during the swapping; and

repeat the instructions to set a first pointer, set a second pointer and swap until the first pointer is after the second pointer in the first list.

24. The decoder (2) according to claim 22 or 23, wherein the no-change codeword consists of one or more zeros.

25. The decoder (2) according to any one of claims 22 to 24, wherein the group index comprises an indicator for each group, each indicator indicating whether the respective group contains only no-change data items or at least one change data item.

26. The decoder (2) according to any one of claims 22 to 25, wherein each previous codeword data item is the most recent previous codeword for the channel in question.

27. The decoder (2) according to any one of claims 22 to 25, wherein each previous codeword data item is derived based on a plurality of previous codewords for the channel in question.

28. A decoder (2) comprising:

means for determining an order of a plurality of channels, for providing haptic data, in a first list based on a respective previous codeword data item for each of the plurality of channels, each previous codeword data item being either a no-change data item indicating no change of channel data, or a change data item indicating a change of channel data, wherein the order is such that all no-change previous codeword data items are arranged at one end of the first list;

means for receiving a group index and current codewords for groups of

channels of the plurality of channels;

means for generating no-change codewords for channels in any group of channels indicated by the group index to contain only no-change codewords;

means for combining the generated codewords with the received codewords, yielding current codewords;

means for arranging the current codewords in a second list accordance with the channel order of the first list;

means for reordering the second list by channels; and

means for outputting the reordered second list.

29. A computer program (167, 91) for decoding haptic data for a plurality of channels, the computer program comprising computer program code which, when run on a decoder (2) causes the decoder (2) to:

determine an order of the plurality of channels in a first list based on a respective previous codeword data item for each of the plurality of channels, each previous codeword data item being either a no-change data item indicating no change of channel data, or a change data item indicating a change of channel data, wherein the order is such that all no-change previous codeword data items are arranged at one end of the first list;

receive a group index and current codewords for groups of channels of the plurality of channels;

generate no-change codewords for channels in any group of channels indicated by the group index to contain only no-change codewords;

combine the generated codewords with the received codewords, yielding current codewords;

arrange the current codewords in a second list accordance with the channel order of the first list;

reorder the second list by channels; and

output the reordered second list.

30. A computer program product (164, 90) comprising a computer program according to claim 29 and a computer readable means on which the computer program is stored.

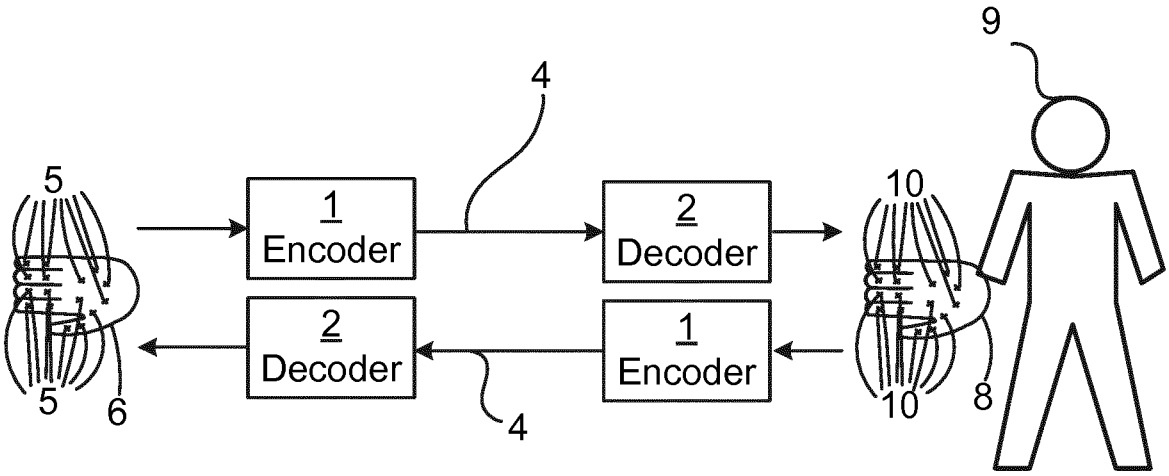
9
4
5
1
Encoder
2
Decoder
10
2
Decoder
1
Encoder
5
6
4
10
8

Fig. 1

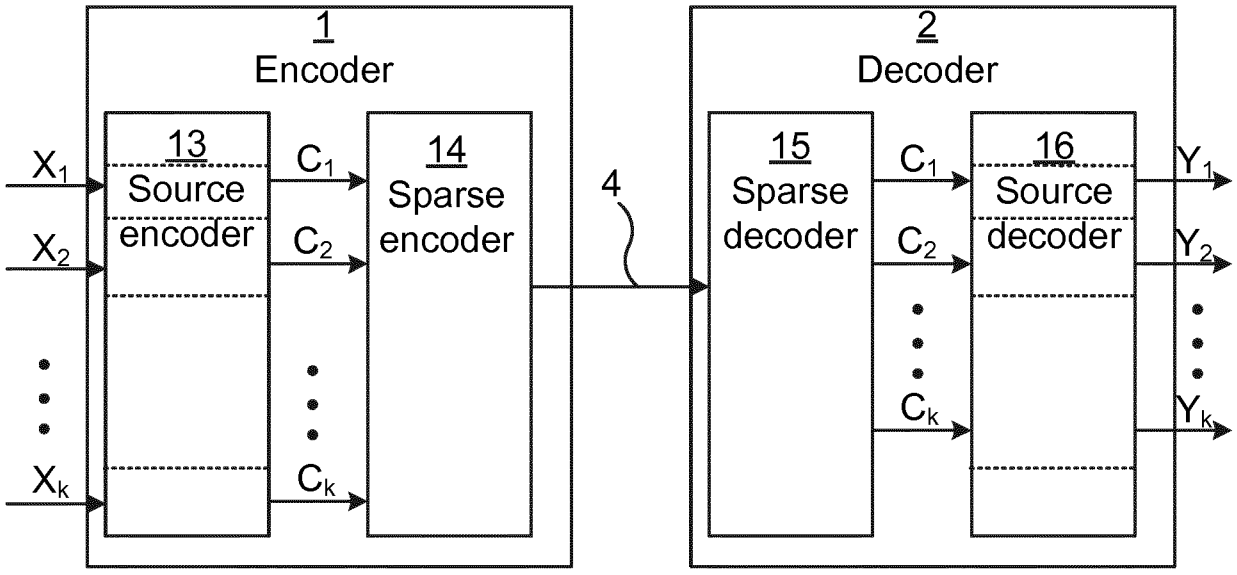
1
Encoder
2
Decoder
13
Source
encoder
14
Sparse
encoder
15
Sparse
decoder
16
Source
decoder
4
X1
X2
Xk
C1
C2
Ck
Y1
Y2
Yk

Fig. 2

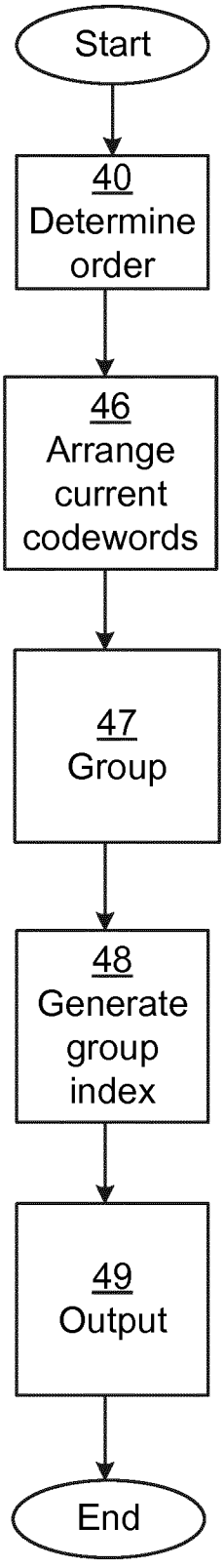
Start
40 Determine order
46 Arrange current codewords
47 Group
48 Generate group index
49 Output
End

Fig. 3A

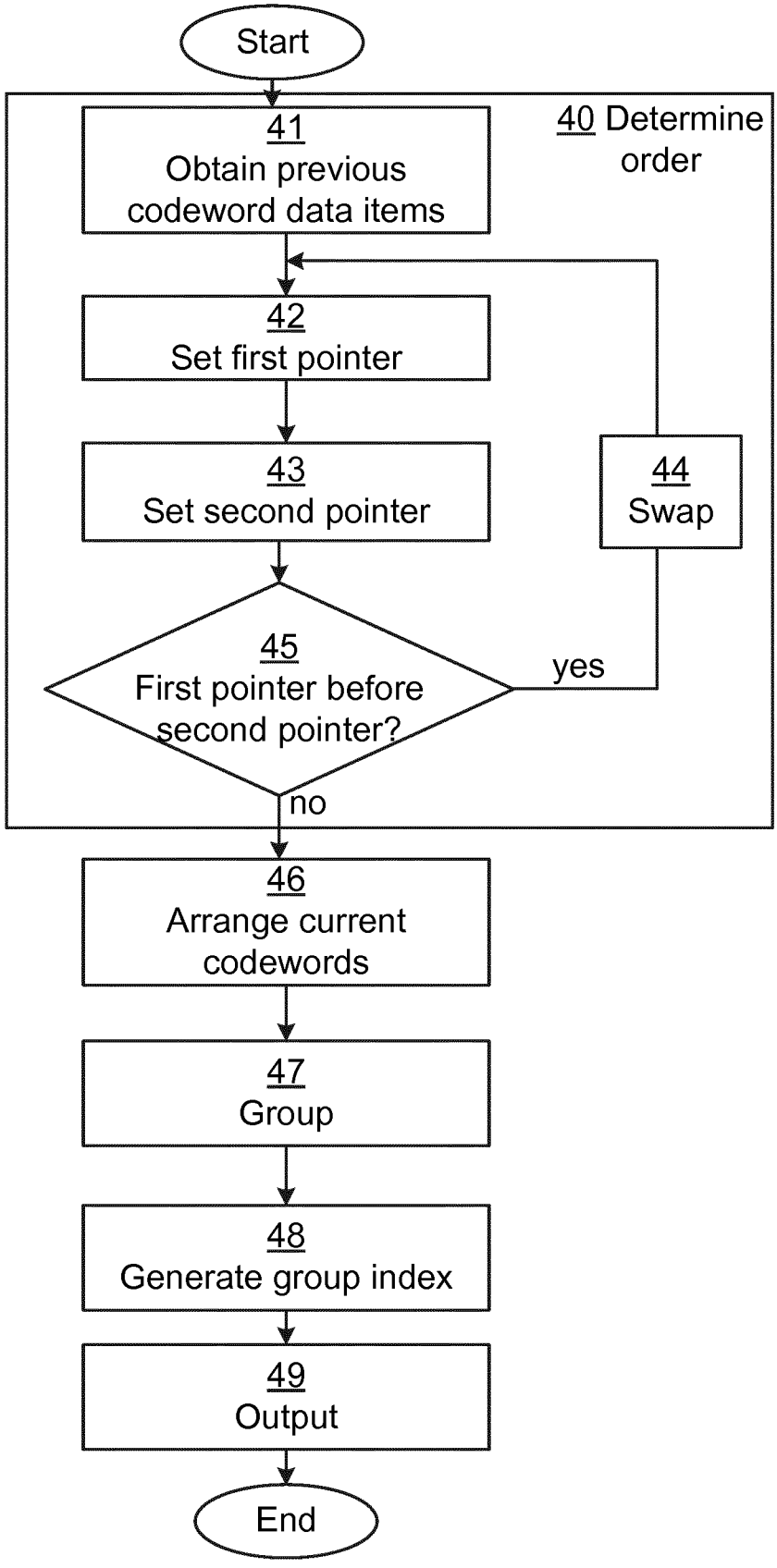
Start
40 Determine order
41 Obtain previous codeword data items
42 Set first pointer
43 Set second pointer
44 Swap
45 First pointer before second pointer?
yes
no
46 Arrange current codewords
47 Group
48 Generate group index
49 Output
End

Fig. 3B

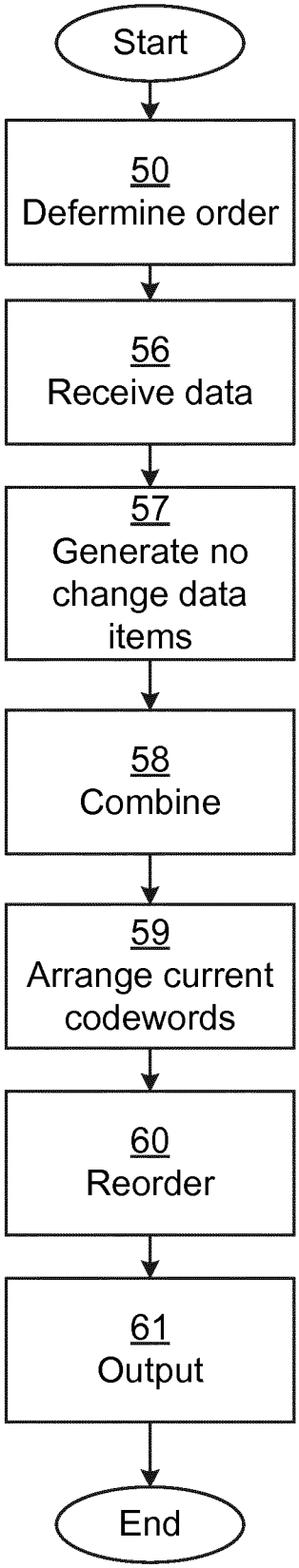
Start
50
Defermine order
56
Receive data
57
Generate no change data items
58
Combine
59
Arrange current codewords
60
Reorder
61
Output
End

Fig. 4A

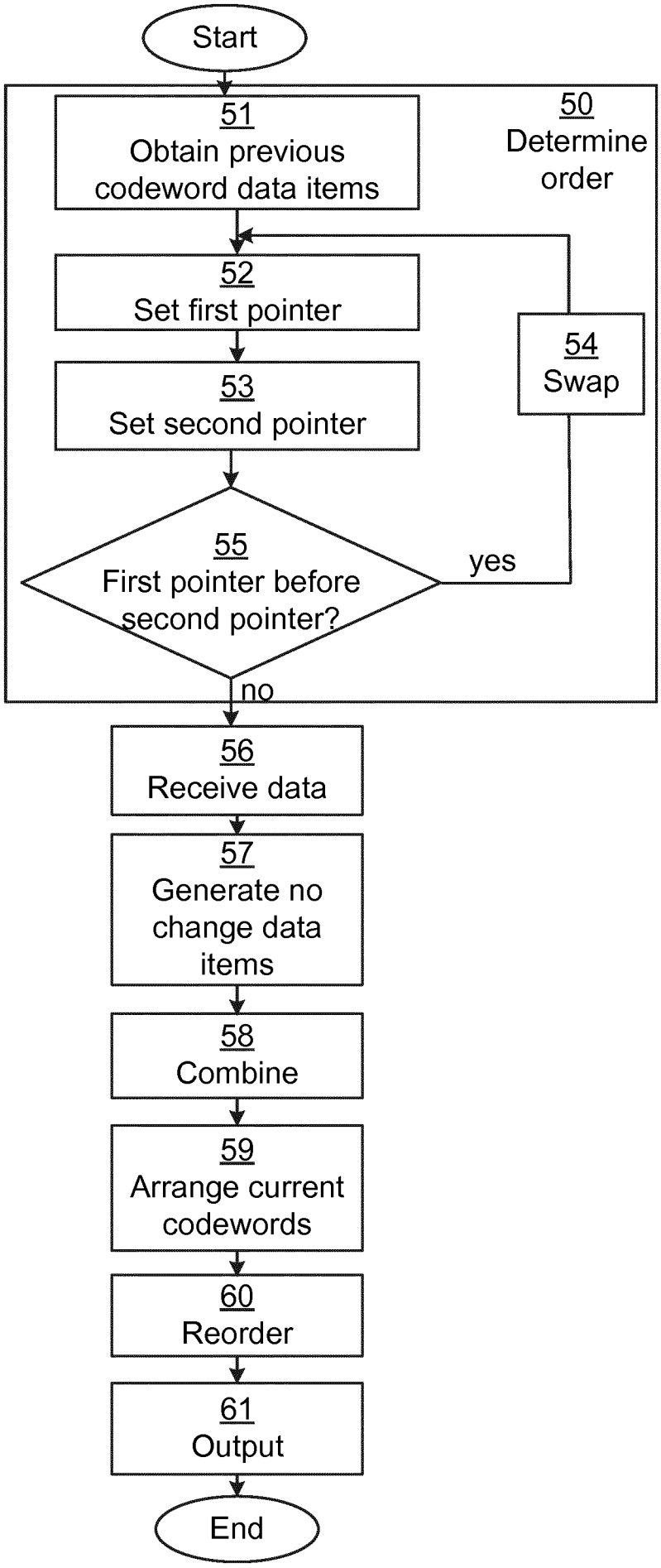
Start
51
Obtain previous codeword data items
50
Determine order
52
Set first pointer
53
Set second pointer
54
Swap
55
First pointer before second pointer?
yes
no
56
Receive data
57
Generate no change data items
58
Combine
59
Arrange current codewords
60
Reorder
61
Output
End

Fig. 4B

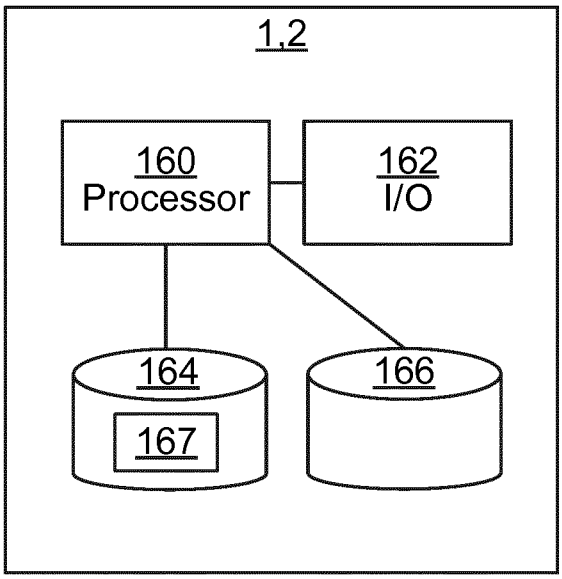
1,2
160
Processor
162
I/O
164
167
166

Fig. 5

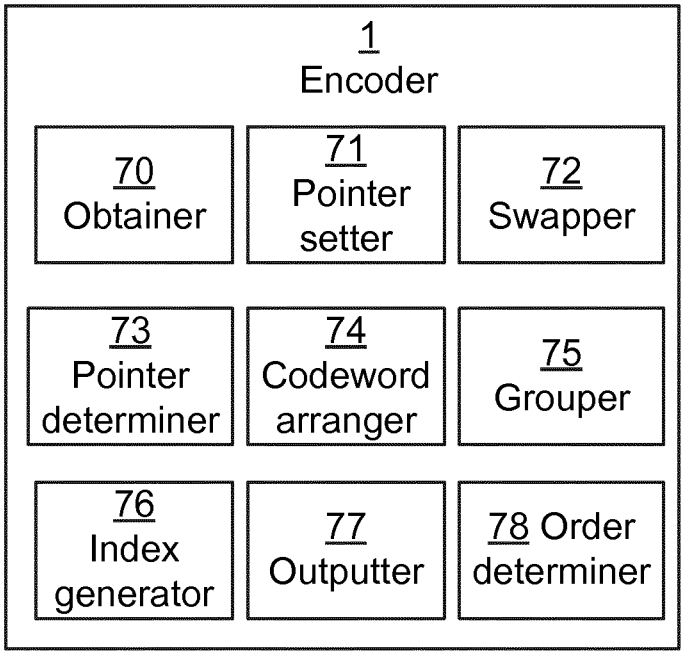
1
Encoder
70
Obtainer
71
Pointer setter
72
Swapper
73
Pointer determiner
74
Codeword arranger
75
Grouper
76
Index generator
77
Outputter
78 Order determiner

Fig. 6

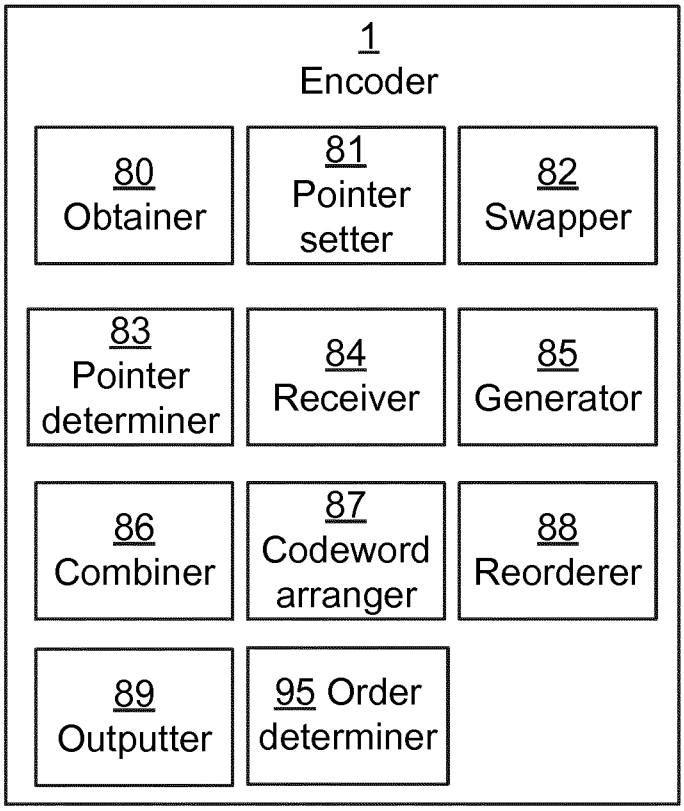
1
Encoder
80
Obtainer
81
Pointer setter
82
Swapper
83
Pointer determiner
84
Receiver
85
Generator
86
Combiner
87
Codeword arranger
88
Reorderer
89
Outputter
95 Order determiner

Fig. 7

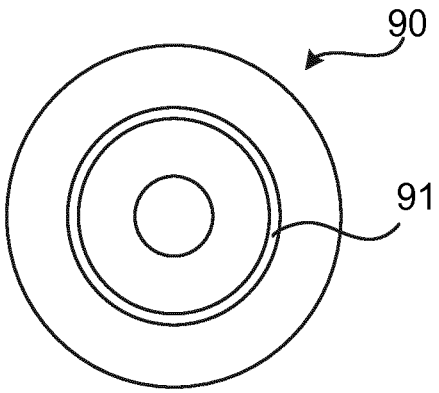
90
91

Fig. 8

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2016/072137

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F3/01
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)
G06F

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

Electronic data base 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

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2003/058216 A1 (LACROIX ROBERT [CA] ET AL) 27 March 2003 (2003-03-27) abstract paragraphs [0002] - [0008]	1-30
A	WO 95/20788 A1 (EXOS INC [US]) 3 August 1995 (1995-08-03) abstract page 5, line 6 - page 6, line 5 page 7, line 1 - line 17 page 11, line 13 - page 13, line 2	1-30
A	US 2009/122006 A1 (NIELSEN KURT TORBEN [US] ET AL) 14 May 2009 (2009-05-14) paragraphs [0003] , [0004]	1-30

☐ Further documents are listed in the continuation of Box C.

☒ 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" documentwhich may throw doubts on priority claim(s) orwhich is cited to establish the publication date of another citation or other special reason (as specified)

"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

"T" 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 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

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
17 May 2017	24/05/2017

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Veaux, Christophe

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/EP2016/072137

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2003058216 A1	27-03-2003	CN 1559066 A	29-12-2004
		US 2003058216 A1	27-03-2003
		WO 03041046 A1	15-05-2003
WO 9520788 A1	03-08-1995	NONE	
us 2009122006 A1	14-05-2009	NONE	