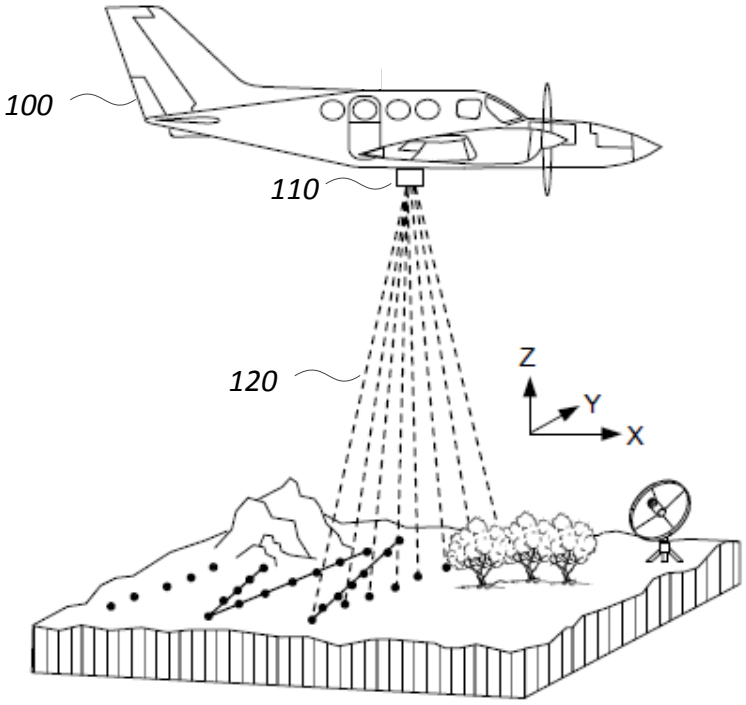

FIG 1A

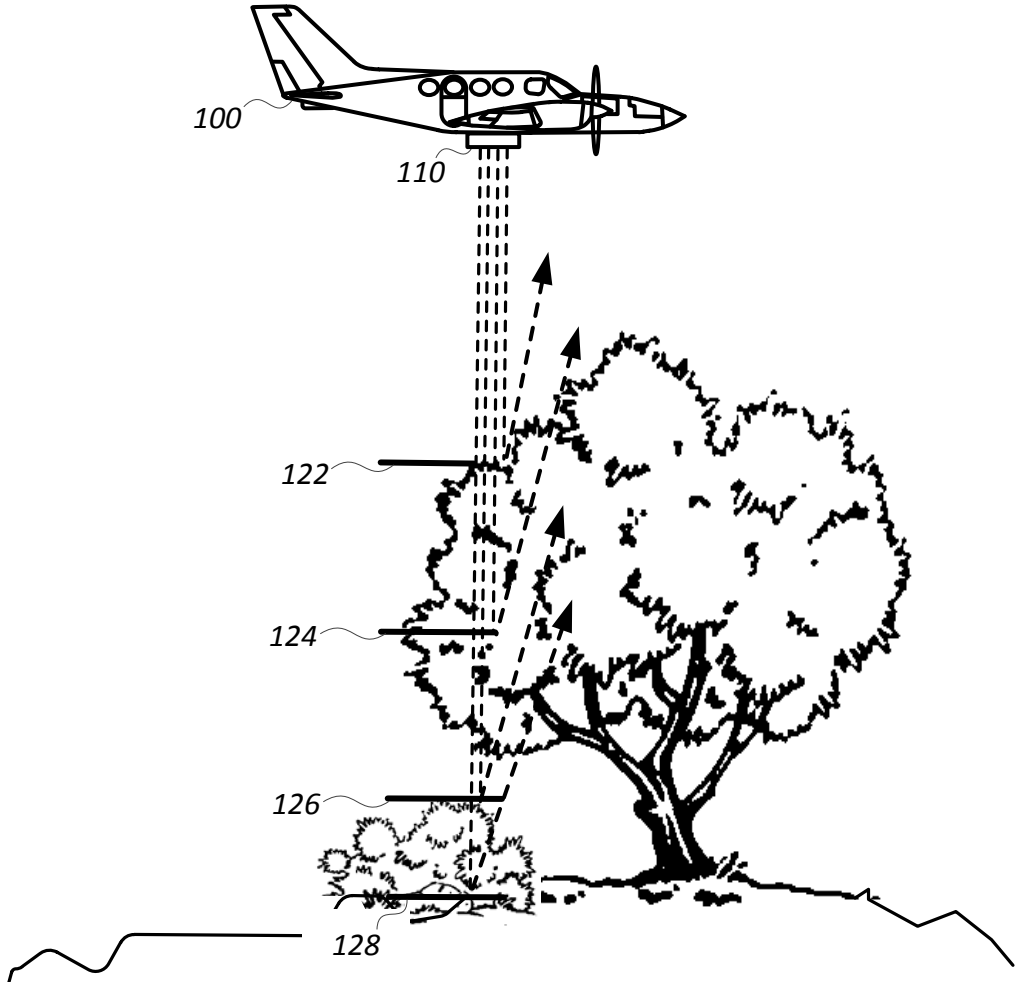

FIG 1B

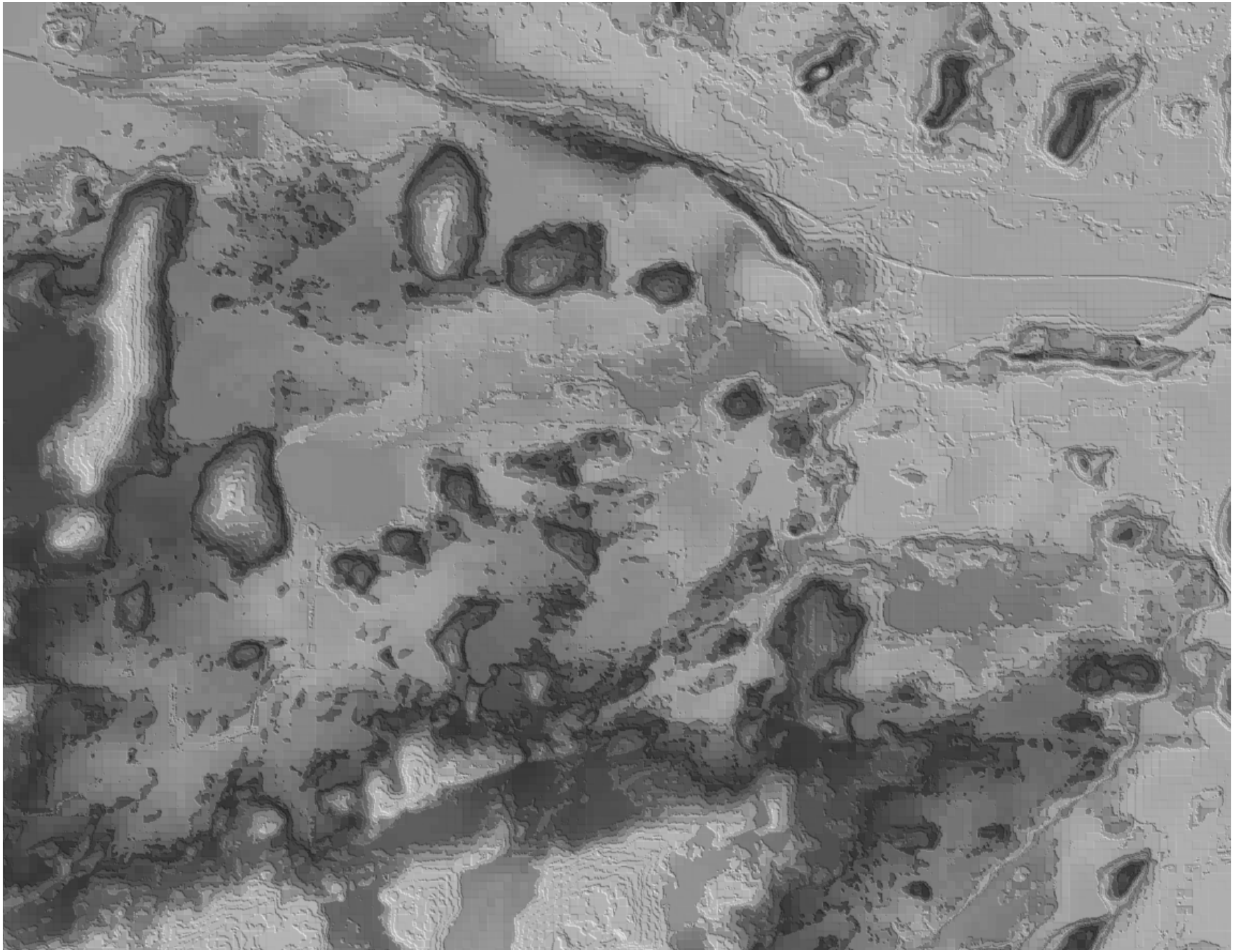

FIG 2A

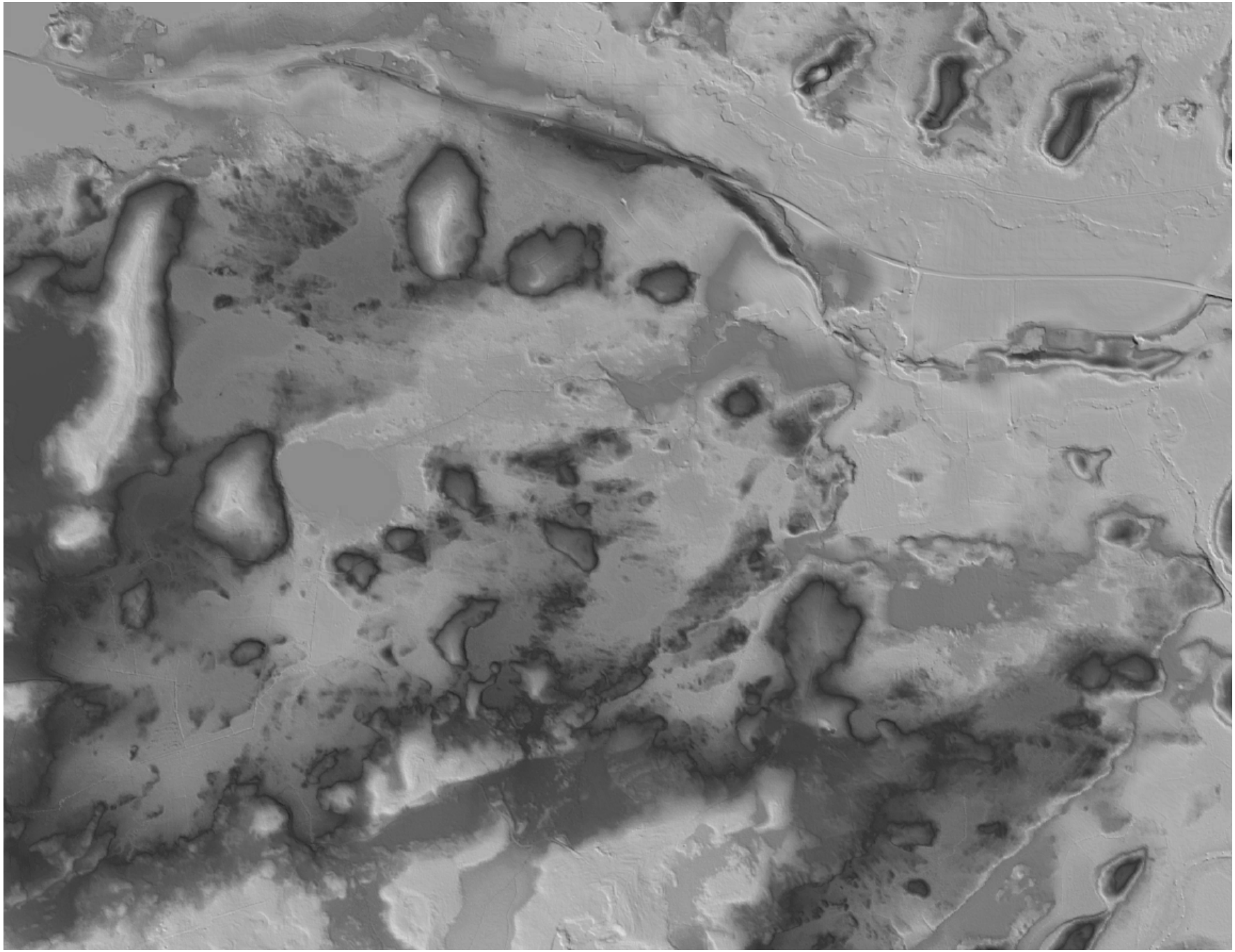

FIG 2B

Compressing a 2D pixel array

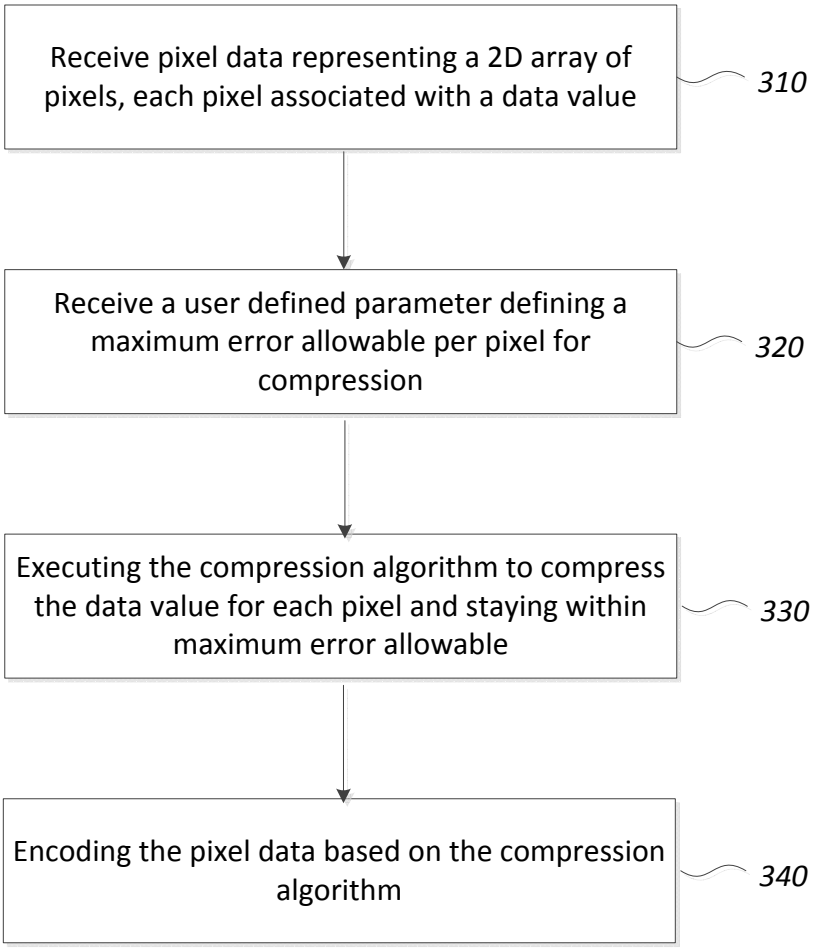

FIG. 3

300

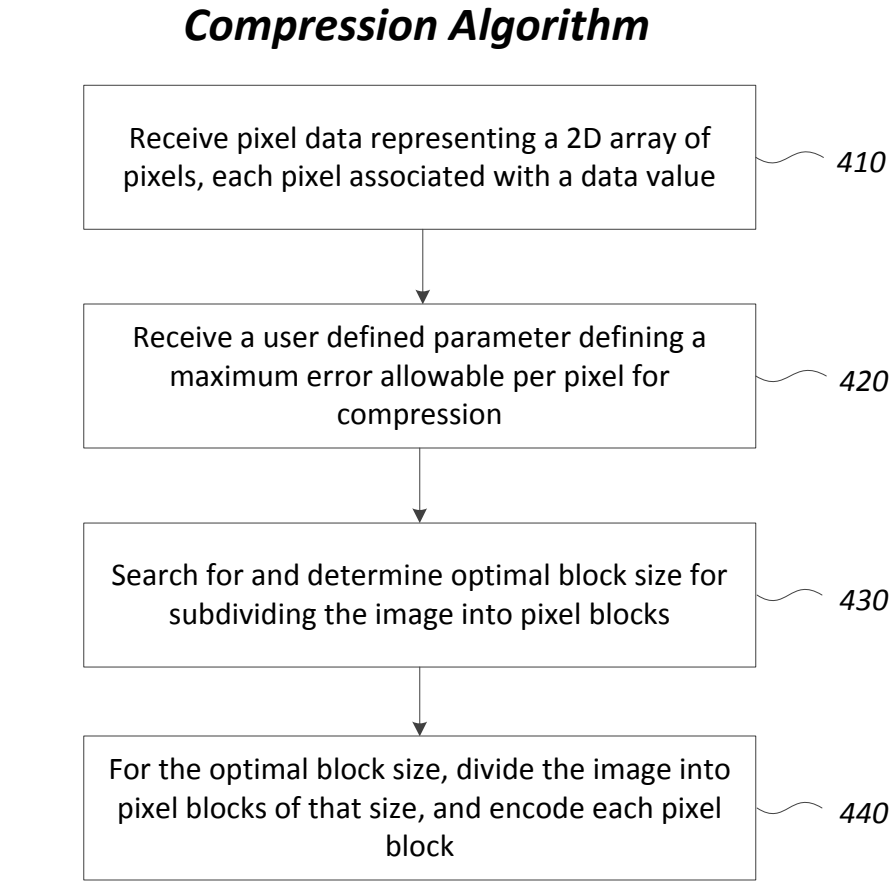
Compression Algorithm
Receive pixel data representing a 2D array of pixels, each pixel associated with a data value
410
Receive a user defined parameter defining a maximum error allowable per pixel for compression
420
Search for and determine optimal block size for subdividing the image into pixel blocks
430
For the optimal block size, divide the image into pixel blocks of that size, and encode each pixel block
440
400

FIG. 4

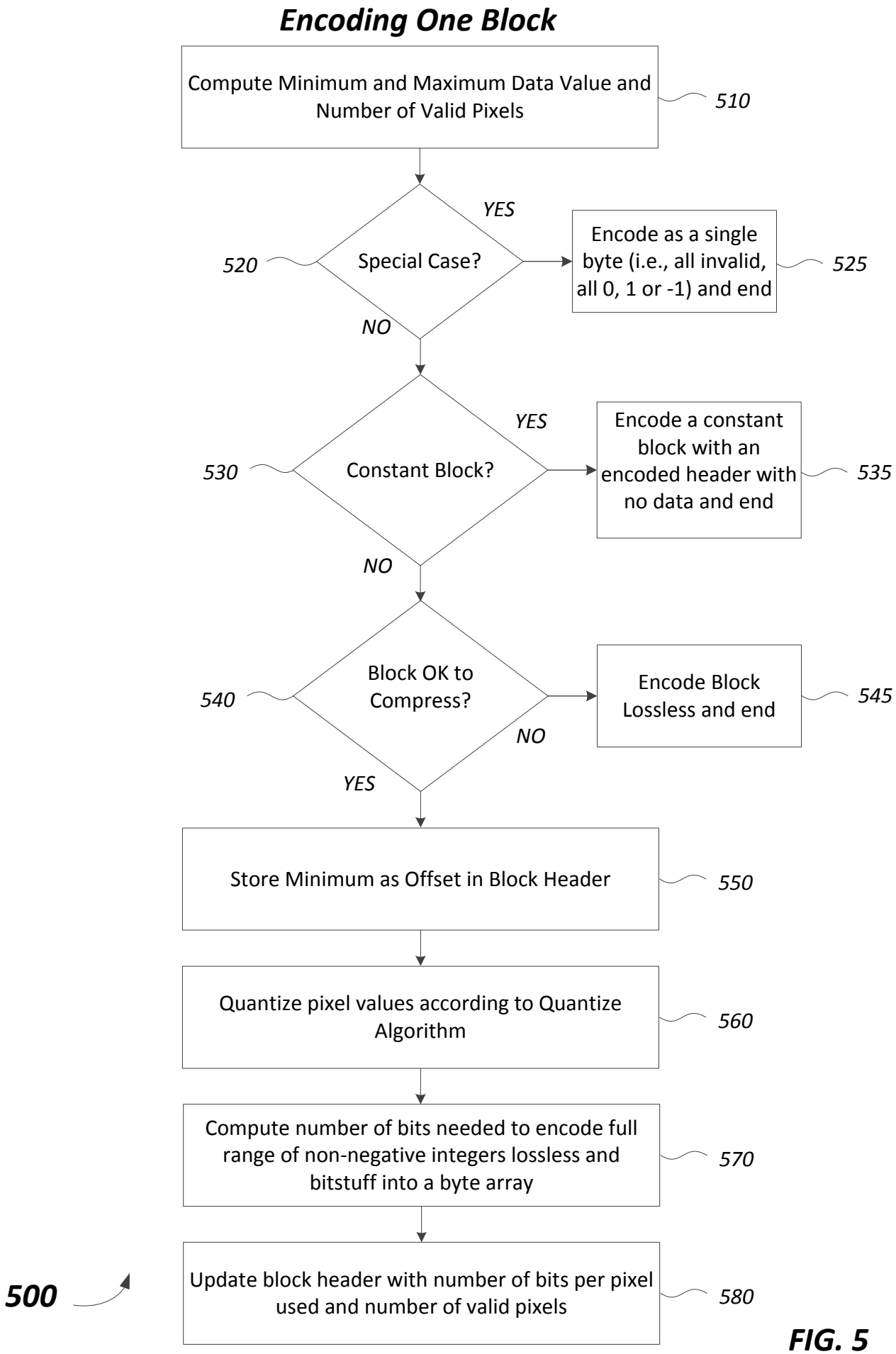
Encoding One Block
Compute Minimum and Maximum Data Value and Number of Valid Pixels
510
Special Case?
520
YES
Encode as a single byte (i.e., all invalid, all 0, 1 or -1) and end
525
NO
Constant Block?
530
YES
Encode a constant block with an encoded header with no data and end
535
NO
Block OK to Compress?
540
NO
Encode Block Lossless and end
545
YES
Store Minimum as Offset in Block Header
550
Quantize pixel values according to Quantize Algorithm
560
Compute number of bits needed to encode full range of non-negative integers lossless and bitstuff into a byte array
570
Update block header with number of bits per pixel used and number of valid pixels
580
500

FIG. 5

Search for Optimal Block Size

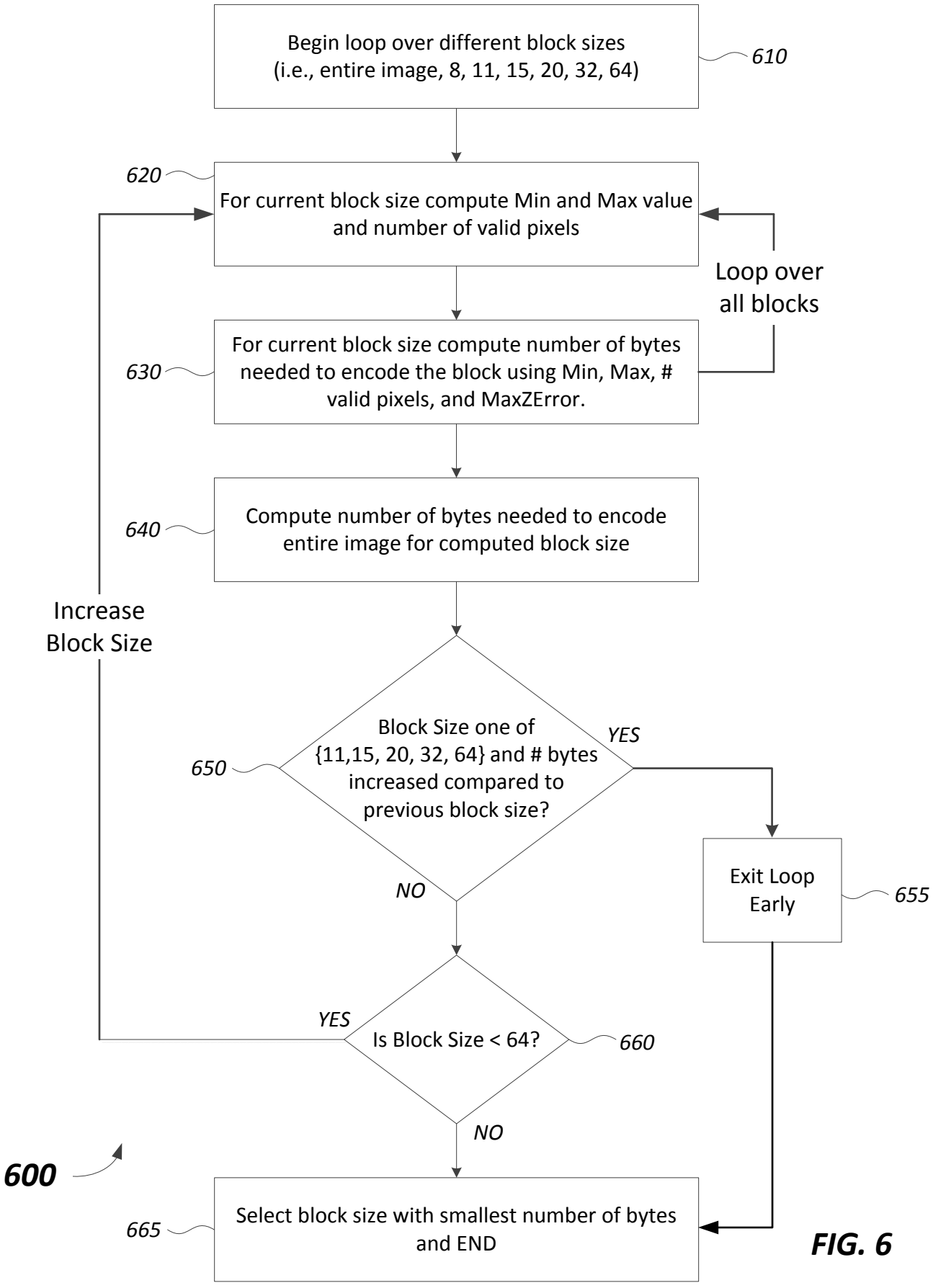

FIG. 6

One Tile Featuring Uniform Size Blocks

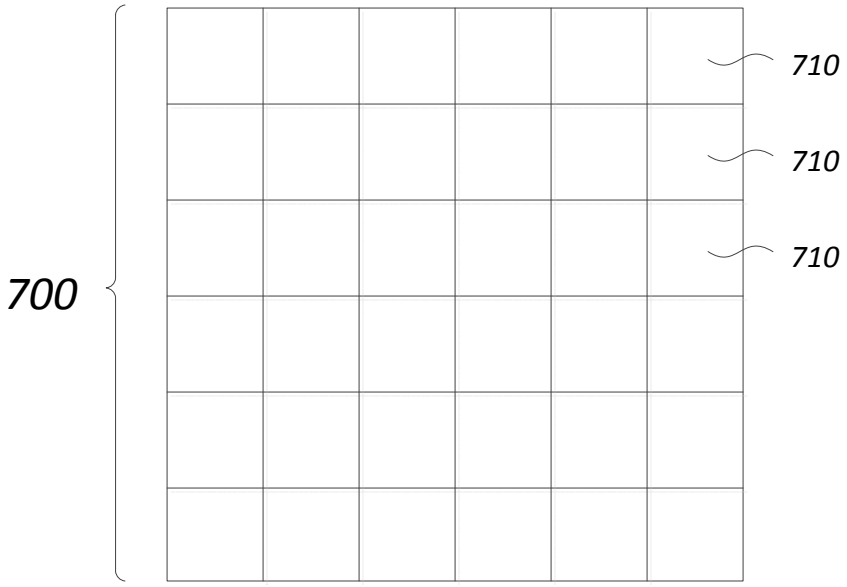

FIG. 7A

Multiple Tiles Featuring Multiple Size Blocks

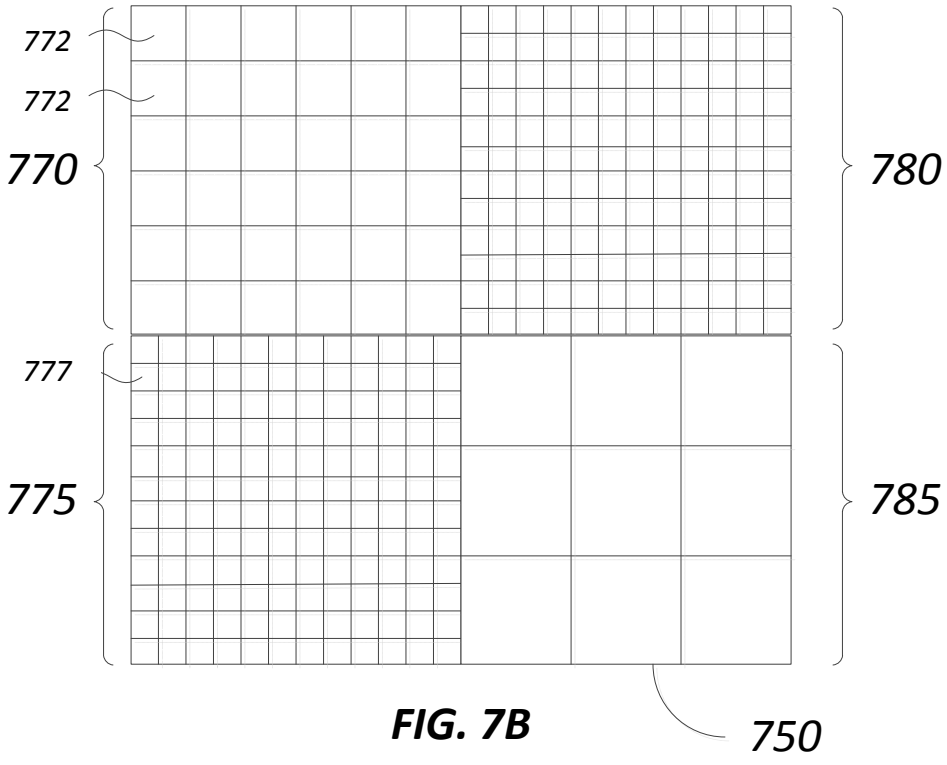

FIG. 7B

LERC File Structure
File Header
Mask Header
Mask Data
Pixel Values Header
Pixel Values Data

FIG. 8A

LERC File Header		
Item	Format	Size
File Identifier String	Char[10]	10
File Version	Int	4
Image Type	Int	4
Image Height in Pixel	Long	4
Image Width in Pixel	Long	4
MaxZError	Double	8

FIG. 8B

LERC Mask or Pixel Values Header		
Item	Format	Size
No. of Blocks, Vertical	Long	4
No. of Blocks, Horizontal	Long	4
Data Size in Bytes	Long	4
Max Value in Image	Float	4

FIG. 8C

LERC Block Header			
Item		Format	Size
Encoding Type (bits 0-5)		Byte	1
	Value = 0: uncompressed float		
	Value = 1: bit stuffed		
	Value = 2: all 0 (encode only this byte)		
	Value = 3: all constant (encode only header)		
	For 0 and 2, the rest of header is skipped		
	Bits 6-7 encode next type		
Offset		Float or short or char	4 or 2 or 1
No. of bits per Pixel (bits 0-5)		Byte	1
	Bits 6-7 encode the next type		
No. of valid pixels		Byte or unsigned short or unsigned long	1 or 2 or 4

FIG. 8D

1234.1234	1241.8741	1256.2759	1267.2950
1280.8725	1248.2917	1272.7511	1279.3802
void	1222.2943	1239.3072	void
1264.9720	1250.0852	void	void

910

FIG. 9A

591	979	1699	2250
2929	1300	2523	2854
void	0	851	void
2134	1390	void	void

FIG. 9B

920

Item	Value	Size
Encoding Type (bits 0-5): Bits 6-7 encode the next type	1	1
Offset	1222.2943	4
Number of bits per pixel (bits 0-5): Bits 6-7 encode the next type	12	1
Number of valid pixel	12	1

930

FIG. 9C

6	10	17	23
29	13	25	29
void	0	9	void
21	14	void	void

940

FIG. 9D

Item	Value	Size
Encoding Type (bits 0-5): Bits 6-7 encode the next type	1	1
Offset	1222.2943	4
Number of bits per pixel (bits 0-5): Bits 6-7 encode the next type	5	1
Number of valid pixel	12	1

950

FIG. 9E

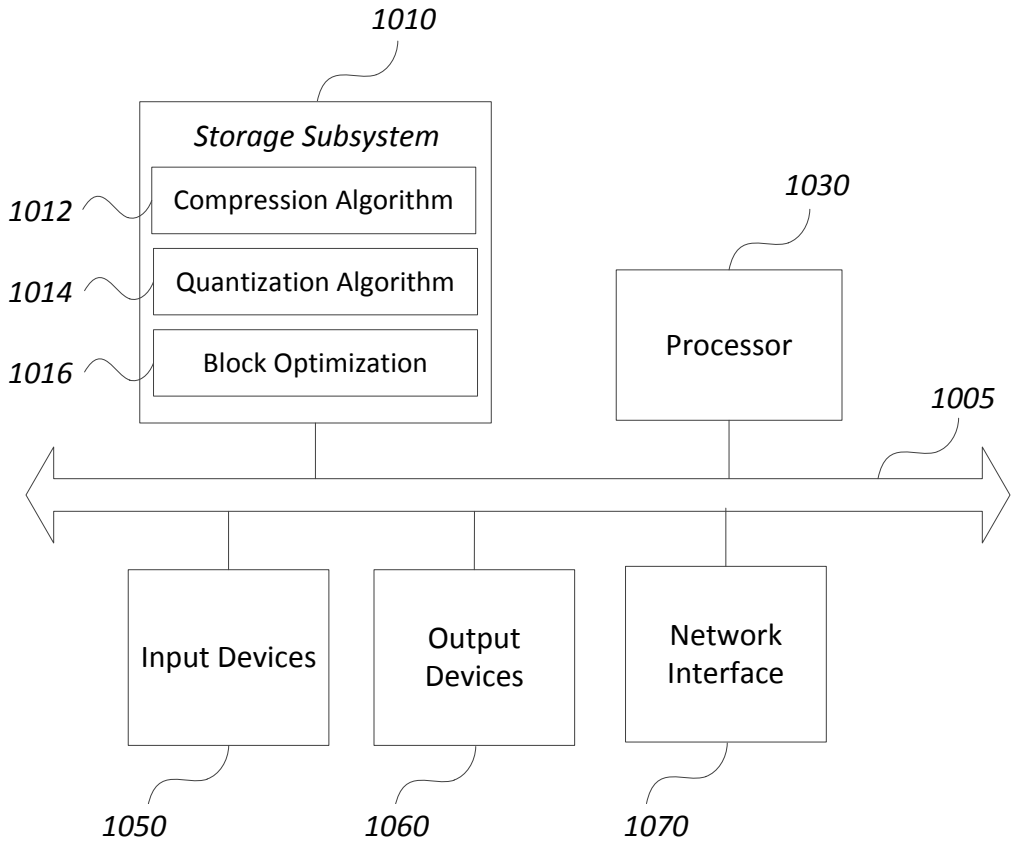
1010
Storage Subsystem
1012
Compression Algorithm
1014
Quantization Algorithm
1016
Block Optimization
1030
Processor
1005
Input Devices
Output Devices
Network Interface
1050
1060
1070

FIG. 10

1000