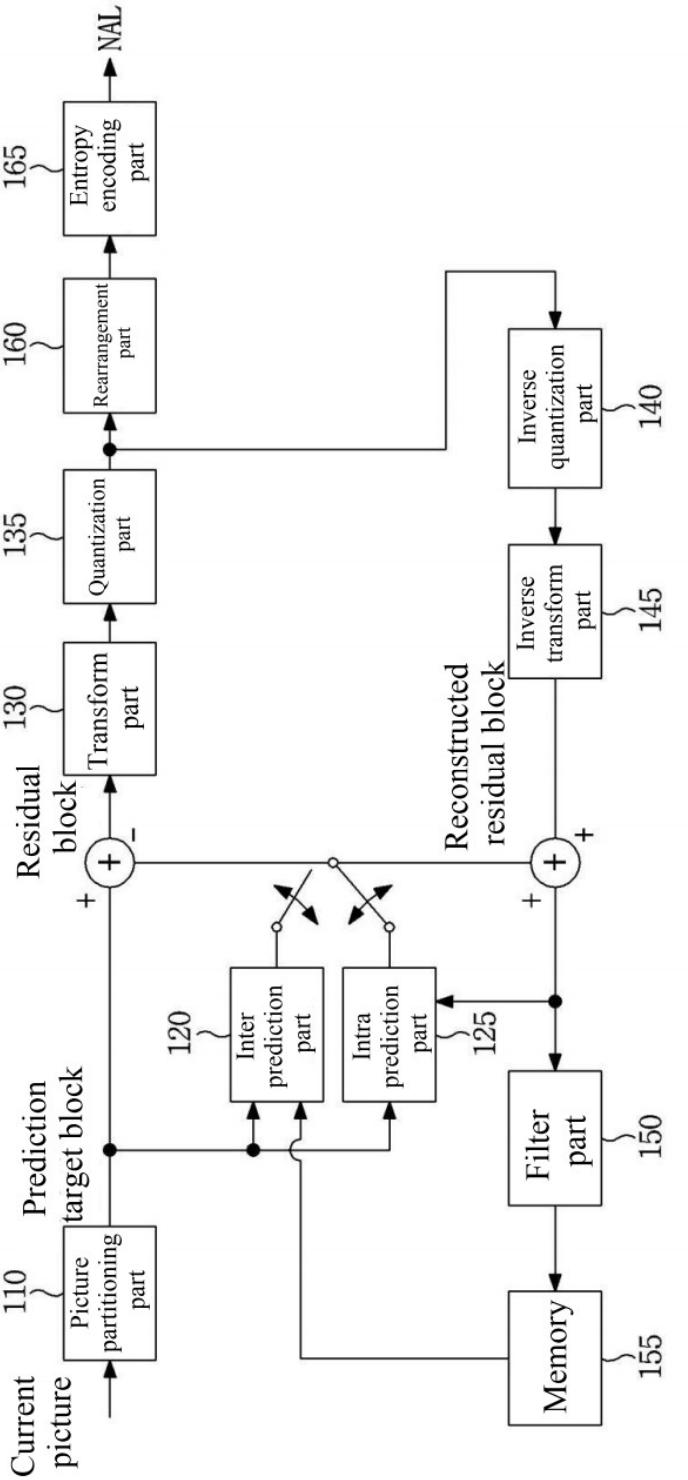

Current picture
110
Picture partitioning part
Prediction target block
120
Inter prediction part
125
Intra prediction part
Residual block
130
Transform part
135
Quantization part
160
Rearrangement part
165
Entropy encoding part
NAL
Reconstructed residual block
Inverse quantization part
140
Inverse transform part
145
Filter part
150
Memory
155

Fig. 1

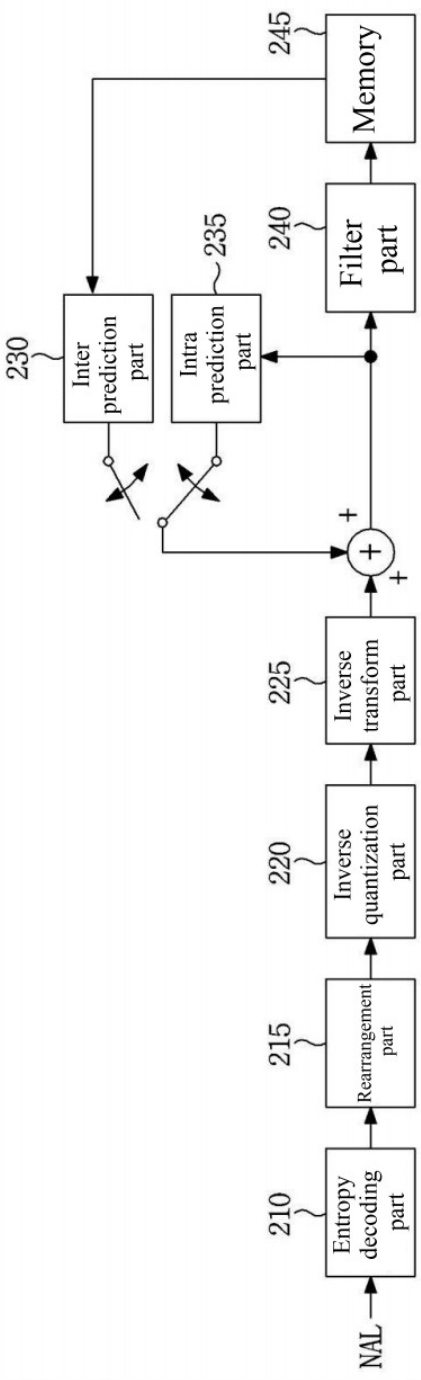

NAL
Entropy decoding part
210
Rearrangement part
215
Inverse quantization part
220
Inverse transform part
225
+
+
+
Inter prediction part
230
Intra prediction part
235
Filter part
240
Memory
245

Fig. 2

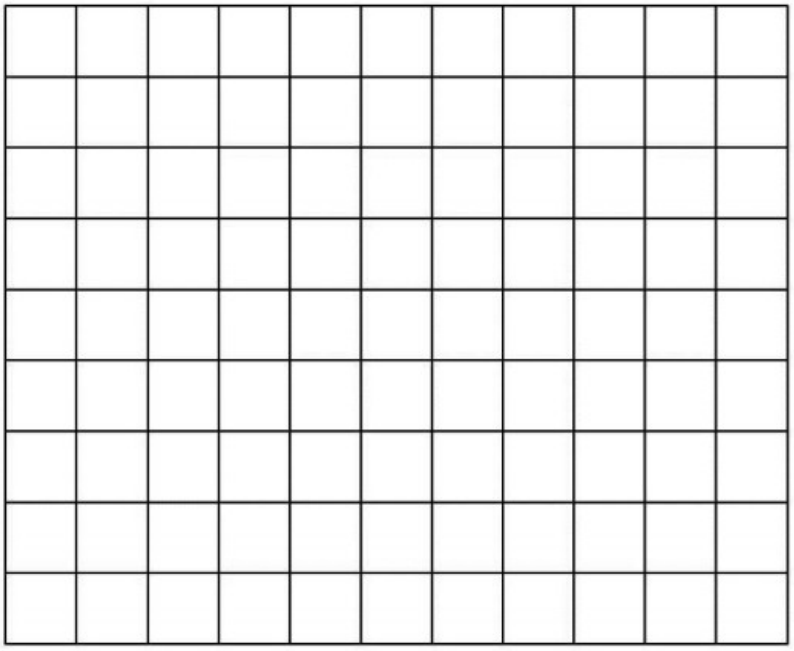


Fig. 3

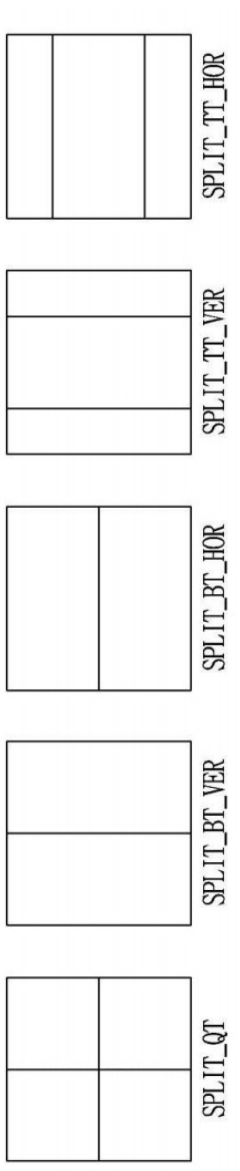


Fig. 4

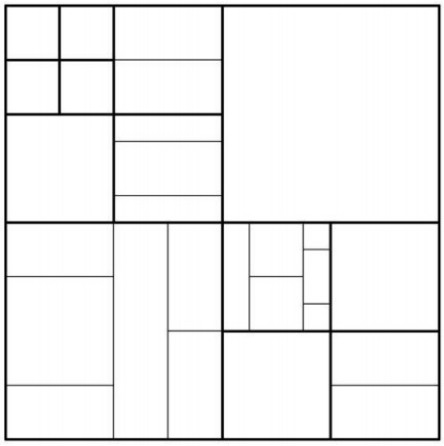


Fig. 5

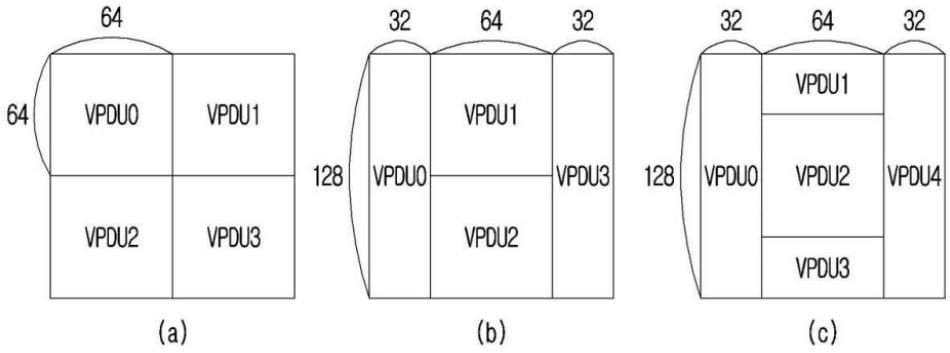

64
64
VPDU0
VPDU1
VPDU2
VPDU3
(a)
32
64
32
128
VPDU0
VPDU1
VPDU2
VPDU3
(b)
32
64
32
128
VPDU0
VPDU1
VPDU2
VPDU3
VPDU4
(c)

Fig. 6

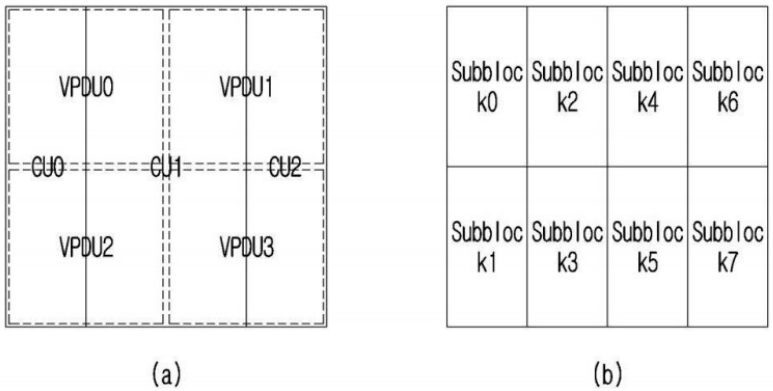


Fig. 7

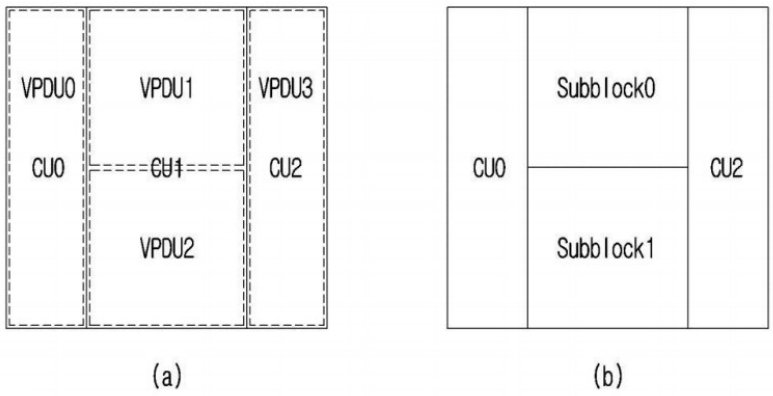


Fig. 8

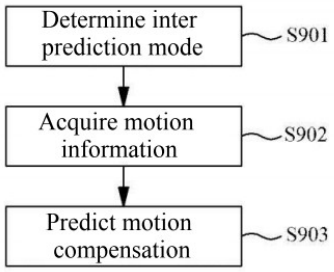


Fig. 9

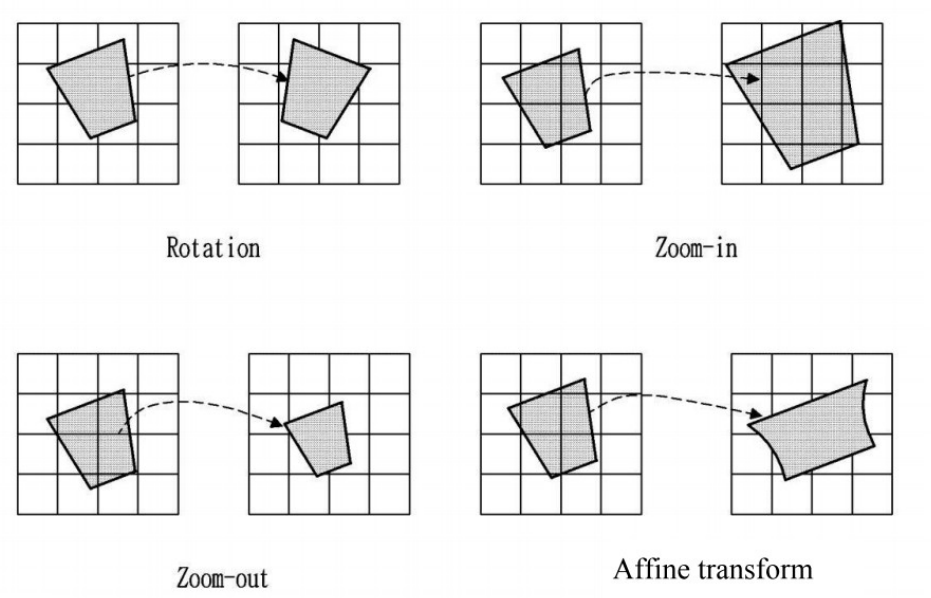

Rotation
Zoom-in
Zoom-out
Affine transform

Fig. 10

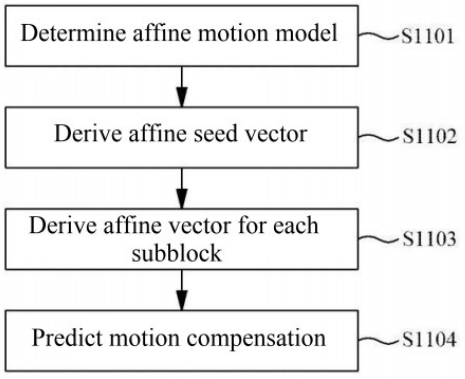

Determine affine motion model
S1101
Derive affine seed vector
S1102
Derive affine vector for each subblock
S1103
Predict motion compensation
S1104

Fig. 11

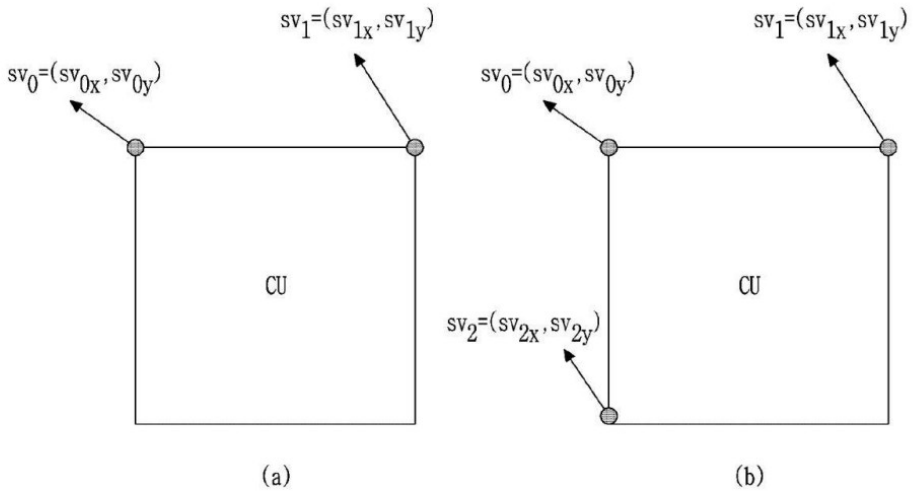

$sv_1=(sv_{1x},sv_{1y})$
$sv_0=(sv_{0x},sv_{0y})$
CU
(a)
$sv_1=(sv_{1x},sv_{1y})$
$sv_0=(sv_{0x},sv_{0y})$
CU
$sv_2=(sv_{2x},sv_{2y})$
(b)

Fig. 12

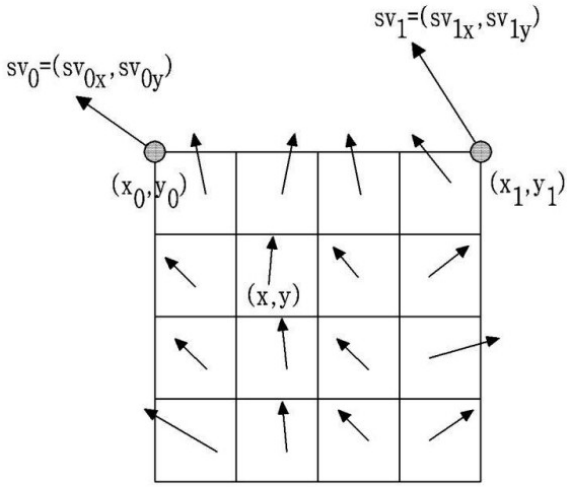

$sv_1=(sv_{1x},sv_{1y})$
$sv_0=(sv_{0x},sv_{0y})$
(x_0,y_0)
(x_1,y_1)
(x,y)

Fig. 13

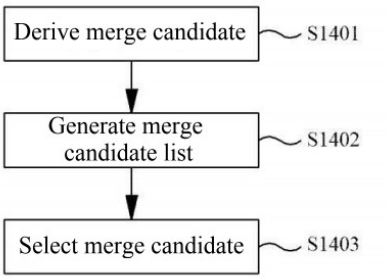

Derive merge candidate
S1401
Generate merge candidate list
S1402
Select merge candidate
S1403

Fig. 14

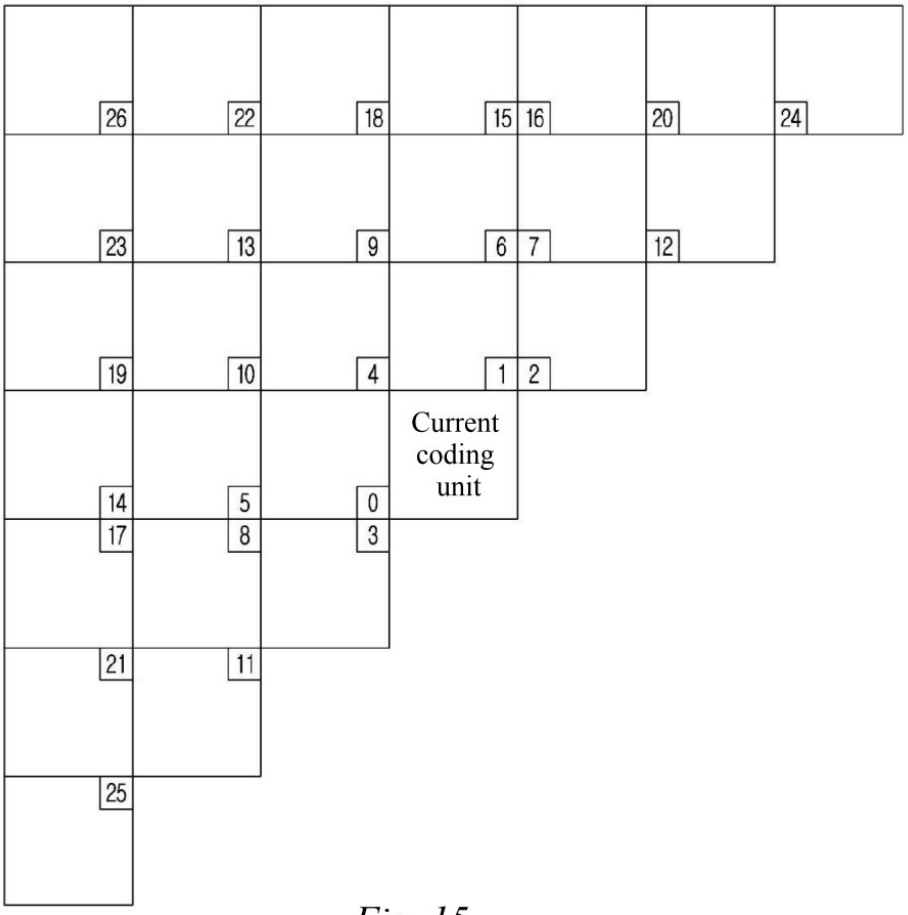

26
22
18
15
16
20
24
23
13
9
6
7
12
19
10
4
1
2
Current coding unit
14
5
0
17
8
3
21
11
25

Fig. 15

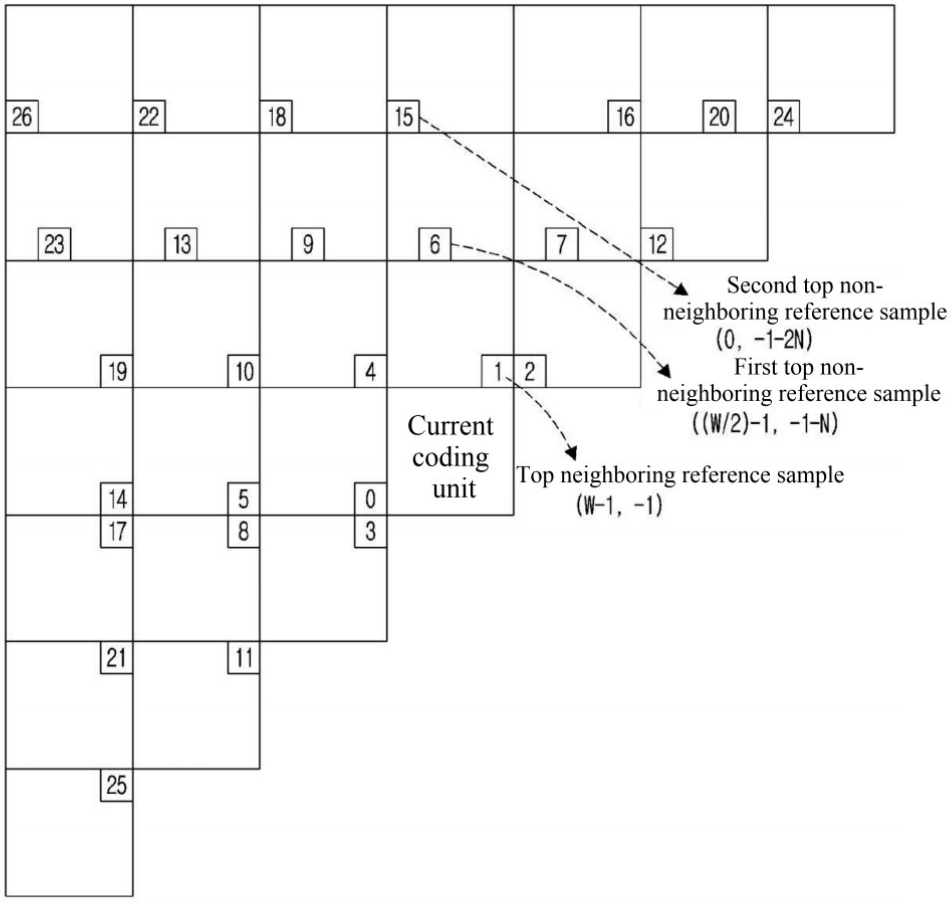

26
22
18
15
16
20
24
23
13
9
6
7
12
19
10
4
1
2
14
5
0
17
8
3
21
11
25
Current coding unit
Second top non-neighboring reference sample (0, -1-2N)
First top non-neighboring reference sample ((W/2)-1, -1-N)
Top neighboring reference sample (W-1, -1)

Fig. 16

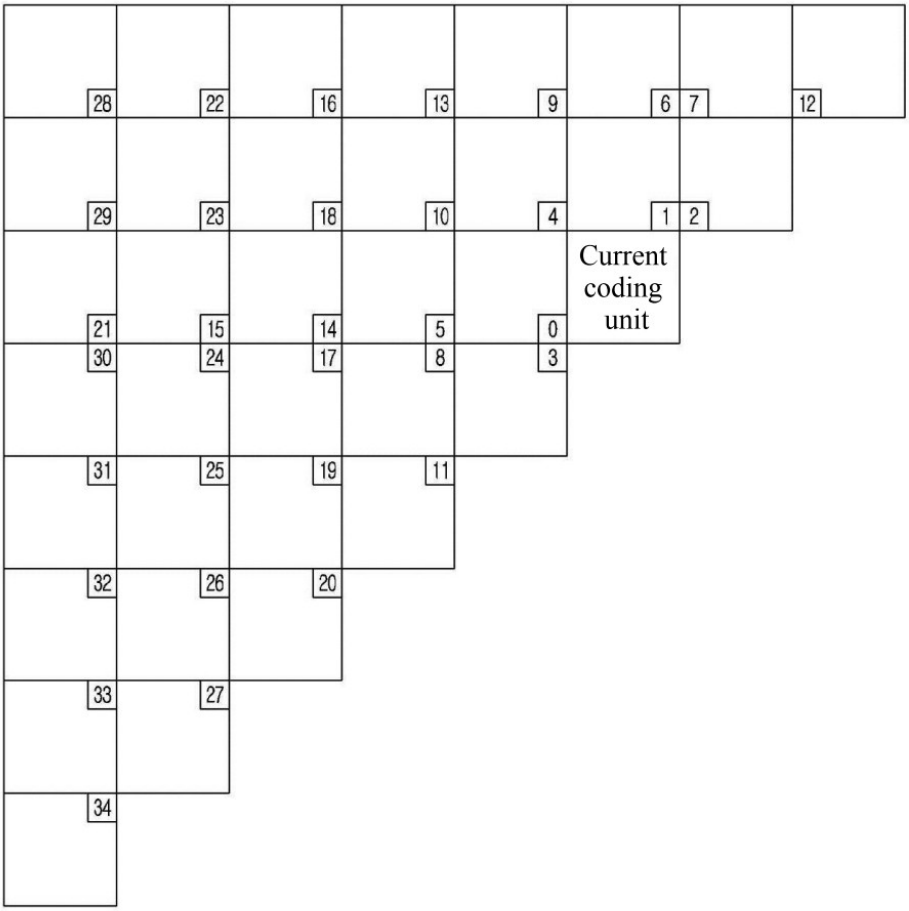

28
22
16
13
9
6
7
12
29
23
18
10
4
1
2
Current coding unit
21
15
14
5
0
30
24
17
8
3
31
25
19
11
32
26
20
33
27
34

Fig. 17

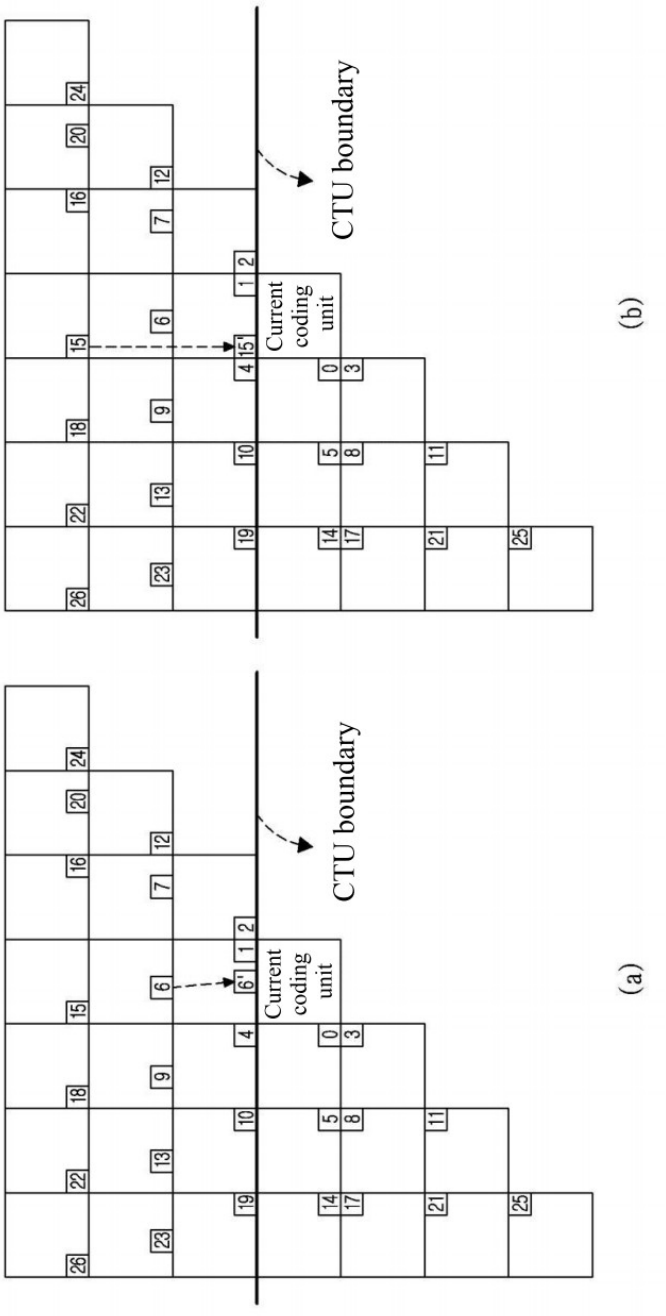

CTU boundary
Current coding unit
(a)
CTU boundary
Current coding unit
(b)

Fig. 18

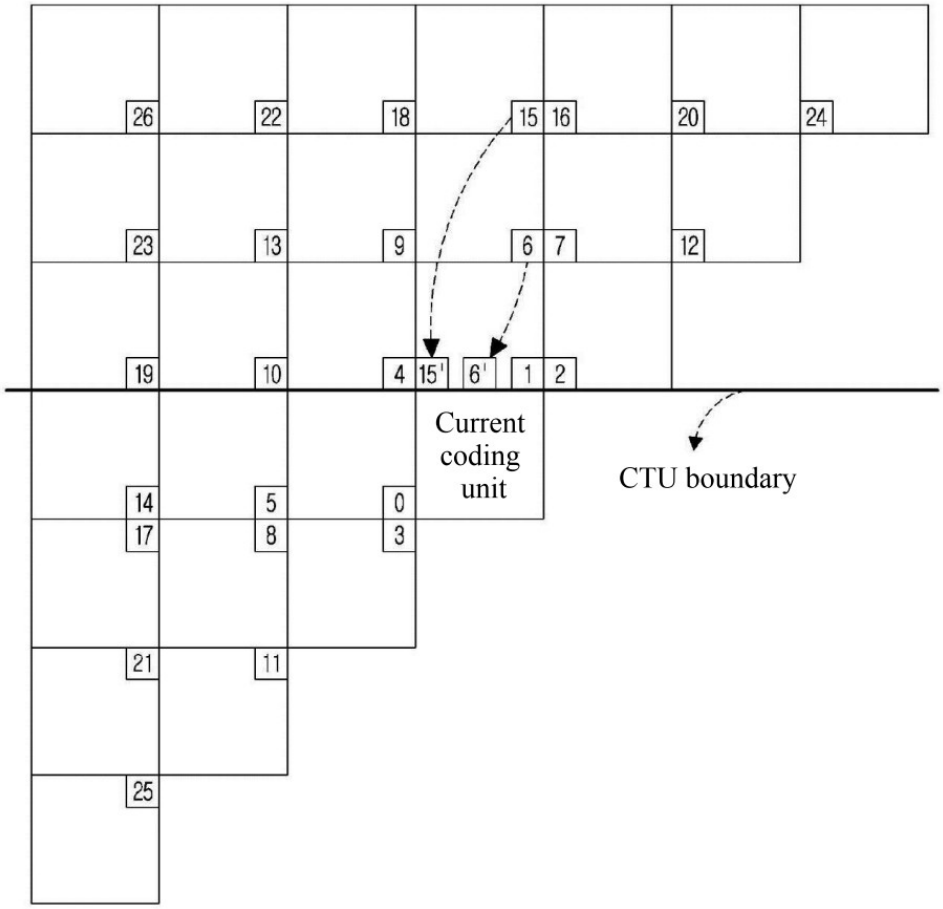

26
22
18
15
16
20
24
23
13
9
6
7
12
19
10
4
15'
6'
1
2
Current coding unit
CTU boundary
14
5
0
17
8
3
21
11
25

Fig. 19

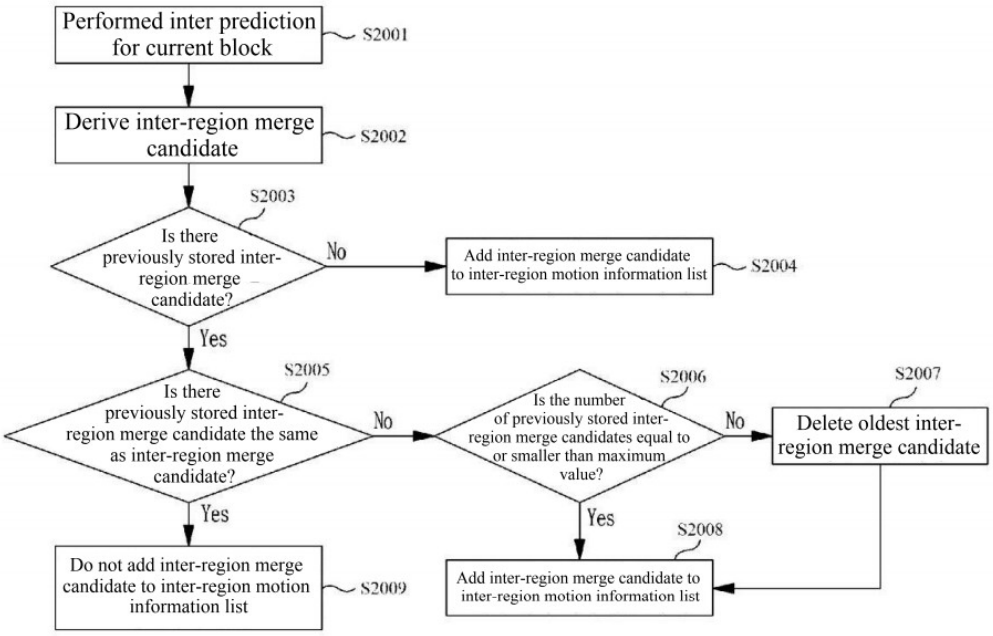

Performed inter prediction for current block
S2001
Derive inter-region merge candidate
S2002
S2003
Is there previously stored inter-region merge candidate?
No
Add inter-region merge candidate to inter-region motion information list
S2004
Yes
S2005
Is there previously stored inter-region merge candidate the same as inter-region merge candidate?
No
S2006
Is the number of previously stored inter-region merge candidates equal to or smaller than maximum value?
No
S2007
Delete oldest inter-region merge candidate
Yes
Yes
S2008
Do not add inter-region merge candidate to inter-region motion information list
S2009
Add inter-region merge candidate to inter-region motion information list

Fig. 20

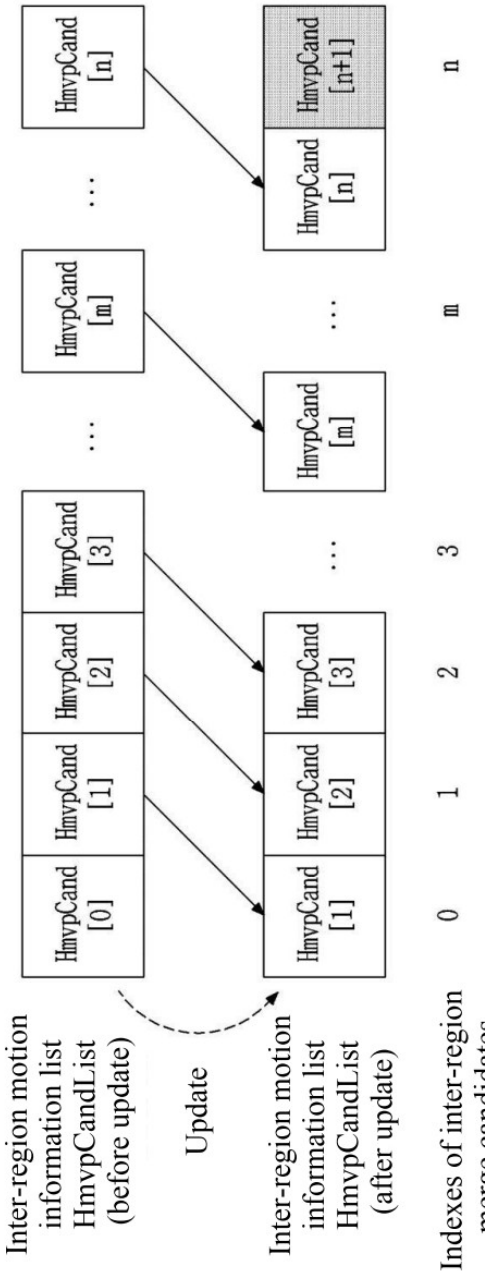

Inter-region motion information list HmvpCandList (before update)
HmvpCand [0]
HmvpCand [1]
HmvpCand [2]
HmvpCand [3]
...
HmvpCand [m]
HmvpCand [n]
Update
Inter-region motion information list HmvpCandList (after update)
HmvpCand [1]
HmvpCand [2]
HmvpCand [3]
...
HmvpCand [m]
HmvpCand [n]
HmvpCand [n+1]
Indexes of inter-region merge candidates
0
1
2
3
...
m
n

Fig. 21

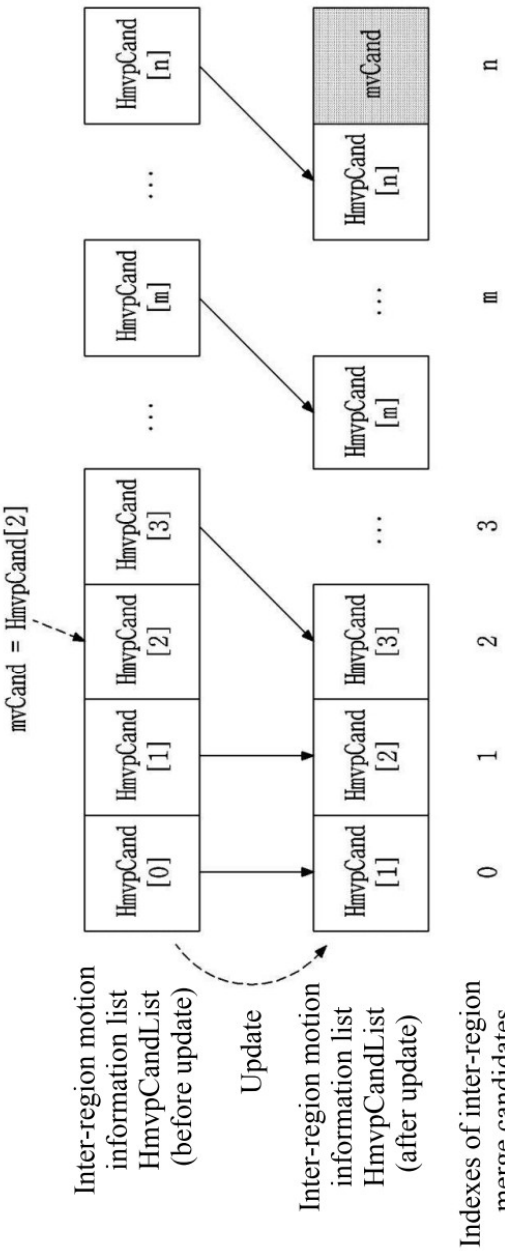

mvCand = HmvpCand[2]
HmvpCand [0]
HmvpCand [1]
HmvpCand [2]
HmvpCand [3]
...
HmvpCand [m]
HmvpCand [n]
Inter-region motion information list HmvpCandList (before update)
Update
HmvpCand [1]
HmvpCand [2]
HmvpCand [3]
...
HmvpCand [m]
HmvpCand [n]
mvCand
Inter-region motion information list HmvpCandList (after update)
Indexes of inter-region merge candidates
0
1
2
3
...
m
n

Fig. 22

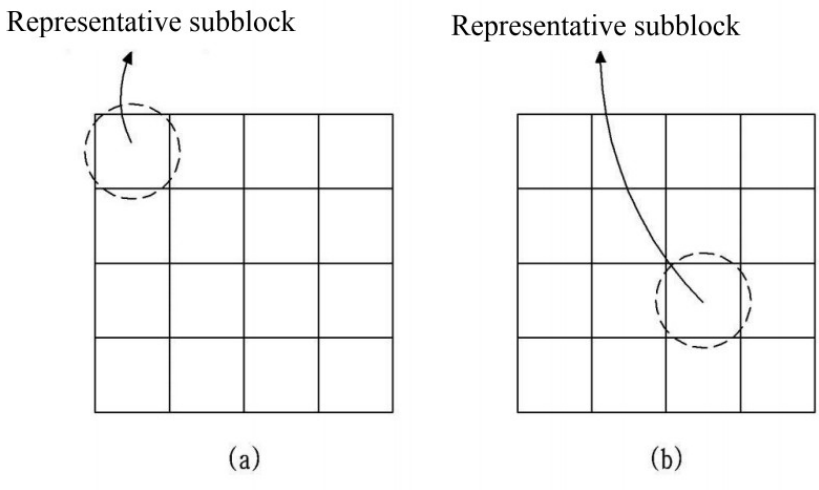


Fig. 23

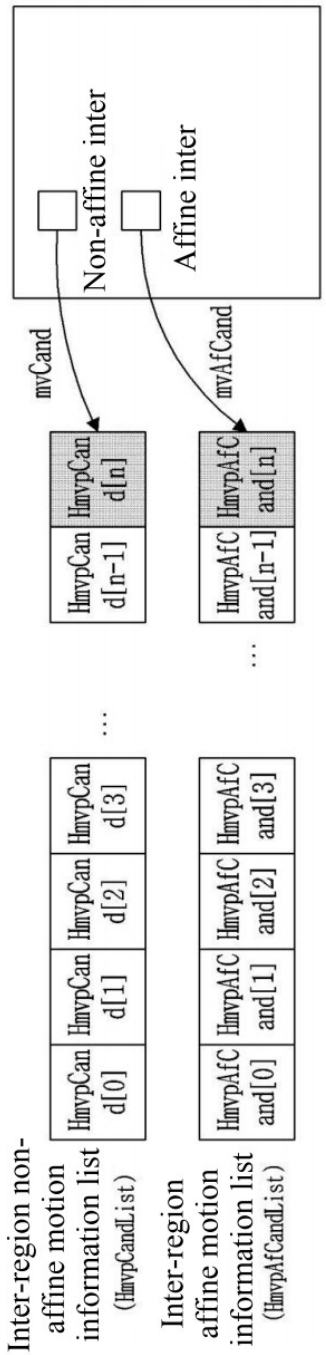

Picture
Non-affine inter
Affine inter
mvCand
mvAfCand
Inter-region non-affine motion information list (HmvpCandList)
Inter-region affine motion information list (HmvpAfCandList)
HmvpCand d[0]
HmvpCand d[1]
HmvpCand d[2]
HmvpCand d[3]
HmvpCand d[n-1]
HmvpCand d[n]
HmvpAfCand[0]
HmvpAfCand[1]
HmvpAfCand[2]
HmvpAfCand[3]
HmvpAfCand[n-1]
HmvpAfCand[n]

Fig. 24

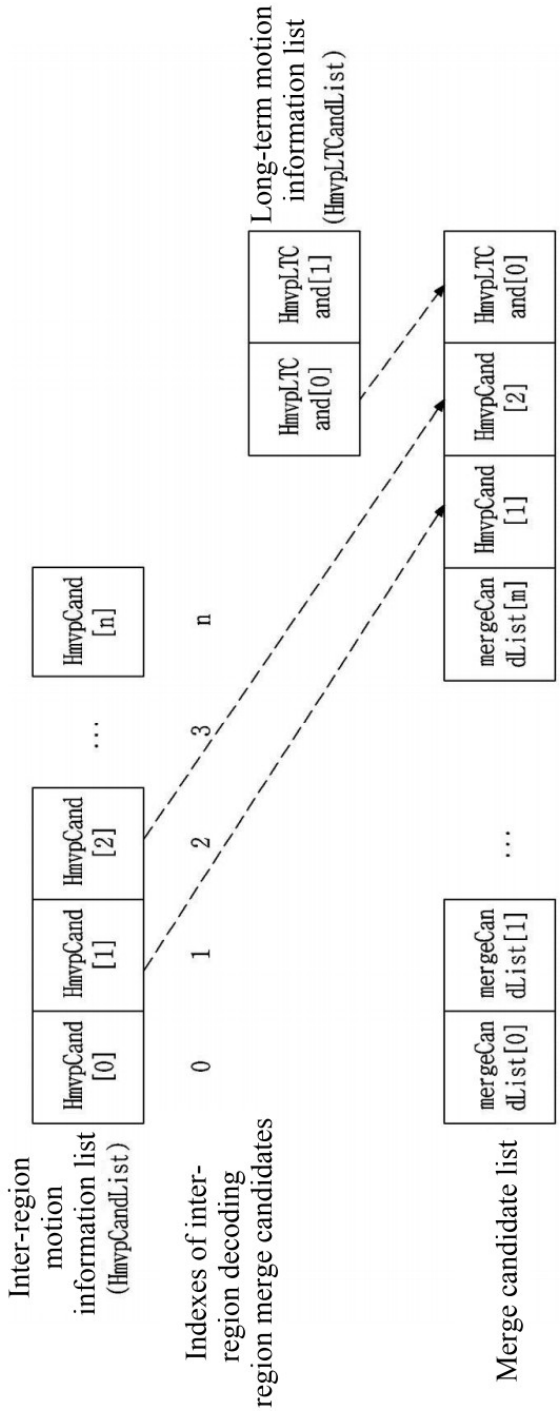

Inter-region motion information list (HmvpCandList)
HmvpCand [0]
HmvpCand [1]
HmvpCand [2]
...
HmvpCand [n]
Indexes of inter-region decoding region merge candidates
0
1
2
3
n
Long-term motion information list (HmvpLTCandList)
HmvpLTC and[0]
HmvpLTC and[1]
Merge candidate list
mergeCan dList[0]
mergeCan dList[1]
...
mergeCan dList[m]
HmvpCand [1]
HmvpCand [2]
HmvpLTC and[0]
HmvpLTC and[1]

Fig. 25

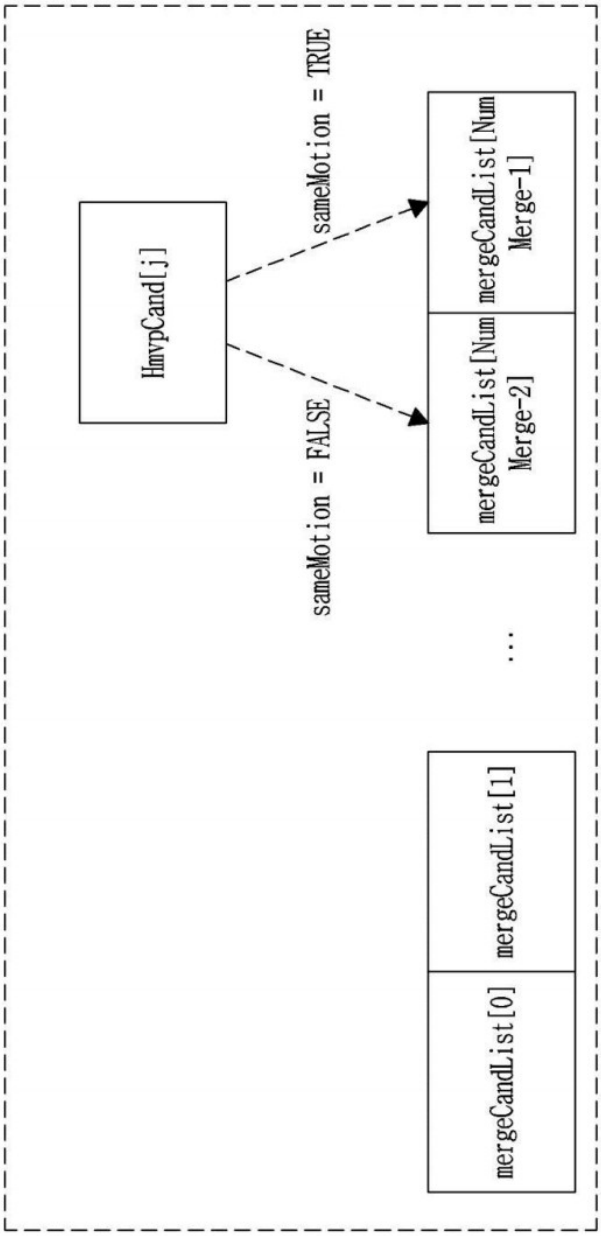

HmvpCand[j]
sameMotion = TRUE
sameMotion = FALSE
mergeCandList[Num Merge-1]
mergeCandList[Num Merge-2]
...
mergeCandList[0]
mergeCandList[1]

Fig. 26

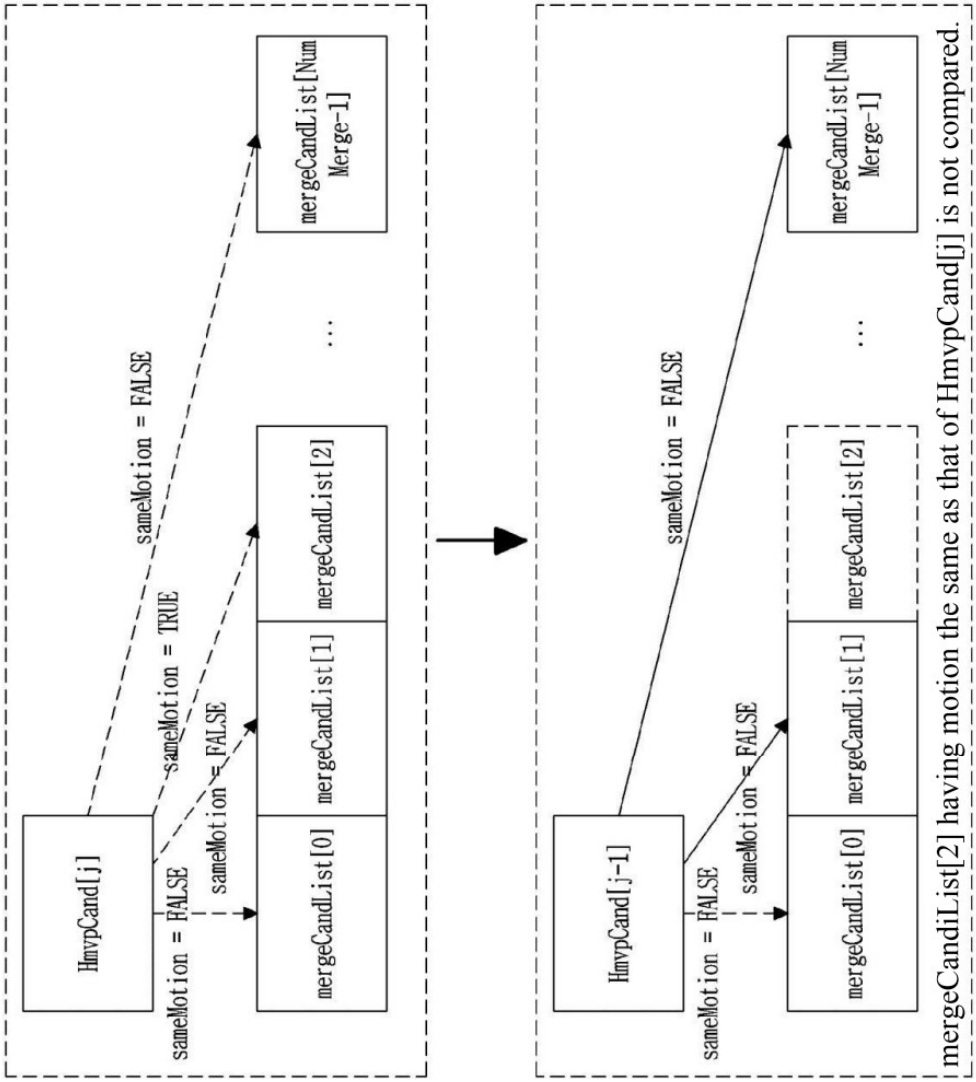

HmvpCand[j]
sameMotion = FALSE
sameMotion = TRUE
sameMotion = FALSE
sameMotion = FALSE
mergeCandList[0]
mergeCandList[1]
mergeCandList[2]
...
mergeCandList[NumMerge-1]
HmvpCand[j-1]
sameMotion = FALSE
sameMotion = FALSE
mergeCandList[0]
mergeCandList[1]
mergeCandList[2]
...
mergeCandList[NumMerge-1]
mergeCandiList[2] having motion the same as that of HmvpCand[j] is not compared.

Fig. 27

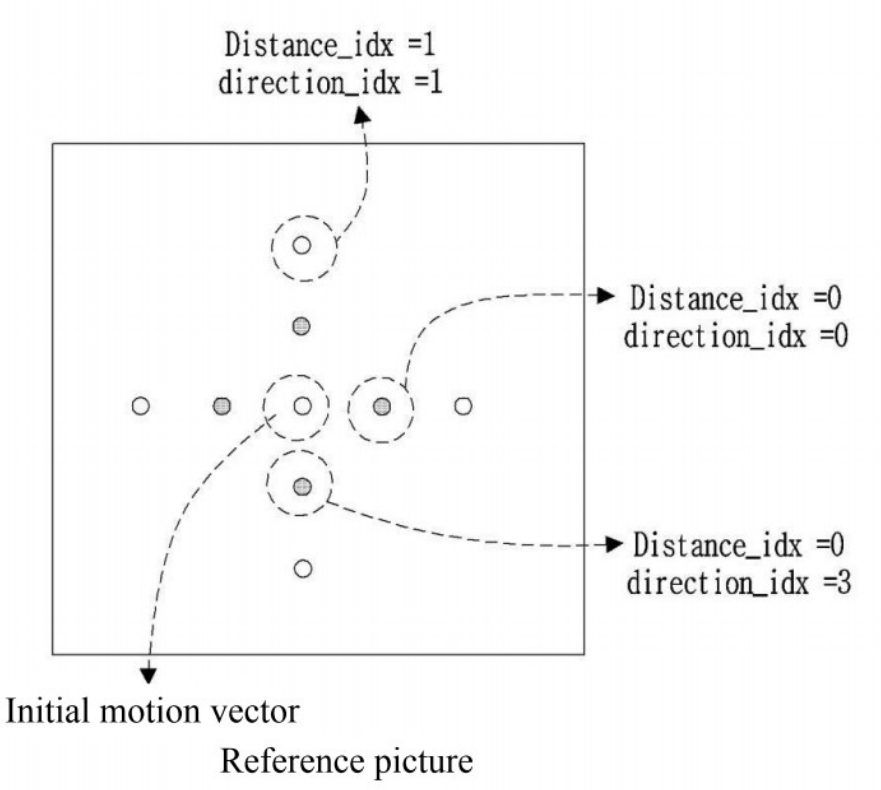

Distance_idx =1
direction_idx =1
Distance_idx =0
direction_idx =0
Distance_idx =0
direction_idx =3
Initial motion vector
Reference picture

Fig. 28

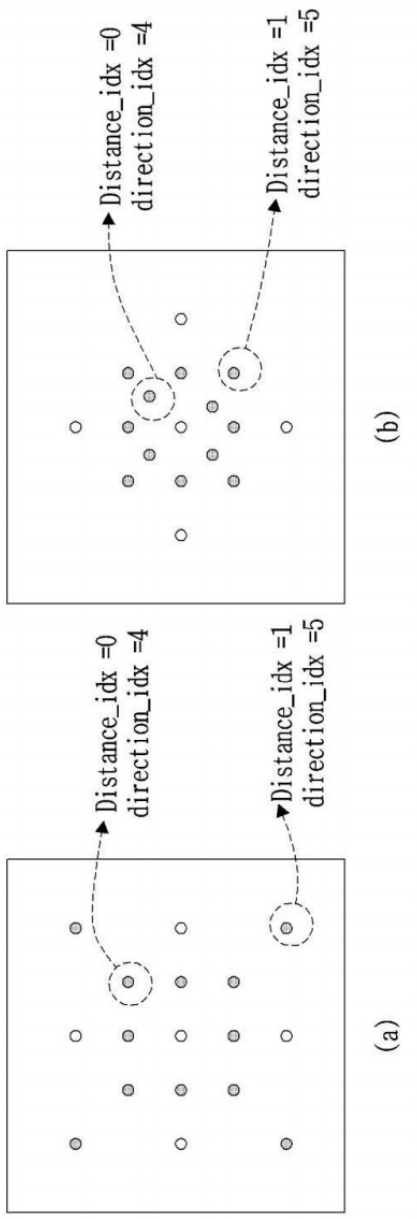

Distance_idx =0
direction_idx =4
Distance_idx =1
direction_idx =5
(a)
Distance_idx =0
direction_idx =4
Distance_idx =1
direction_idx =5
(b)

Fig. 29

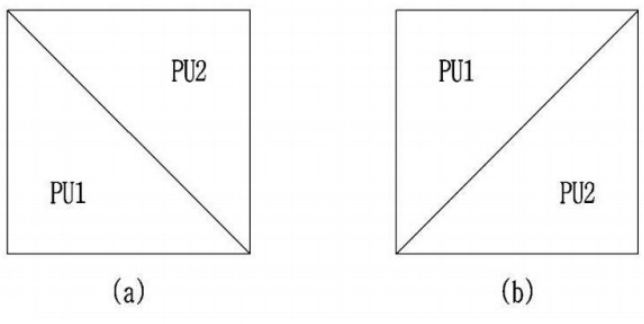

PU2
PU1
(a)
PU1
PU2
(b)

Fig. 30

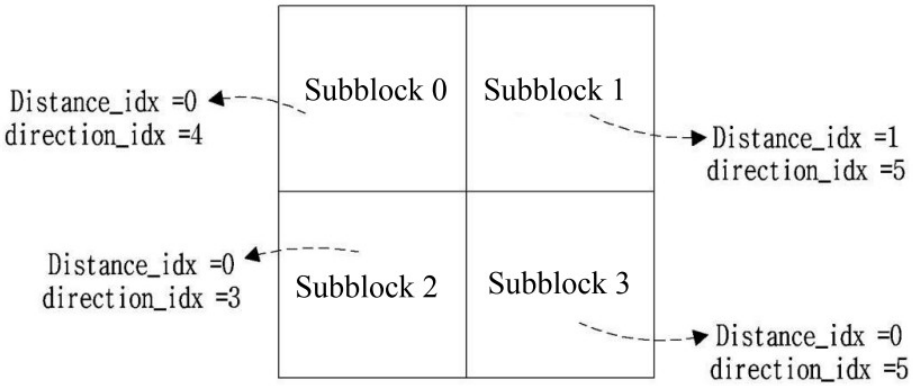

Distance_idx =0
direction_idx =4
Subblock 0
Subblock 1
Distance_idx =1
direction_idx =5
Distance_idx =0
direction_idx =3
Subblock 2
Subblock 3
Distance_idx =0
direction_idx =5

Fig. 31

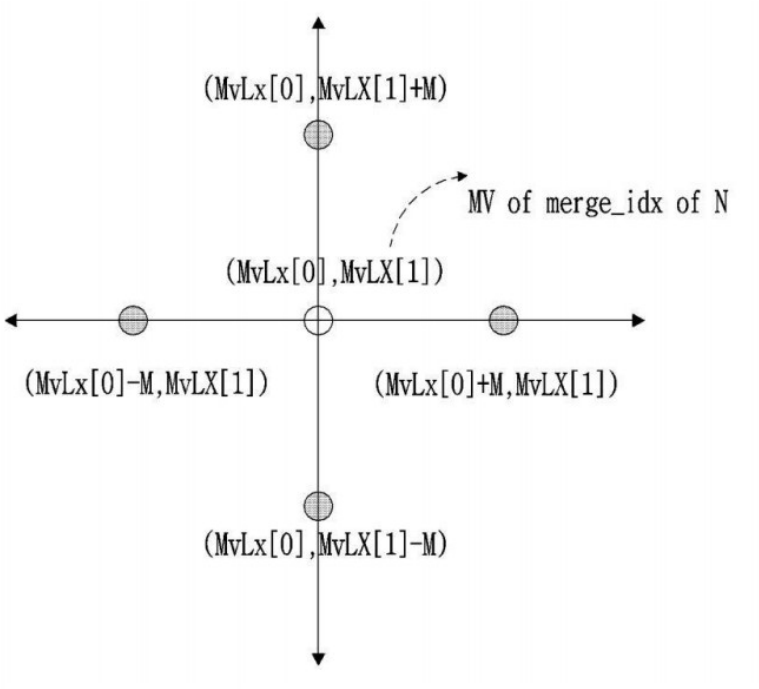

(MvLx[0],MvLX[1]+M)
MV of merge_idx of N
(MvLx[0],MvLX[1])
(MvLx[0]-M,MvLX[1])
(MvLx[0]+M,MvLX[1])
(MvLx[0],MvLX[1]-M)

Fig. 32

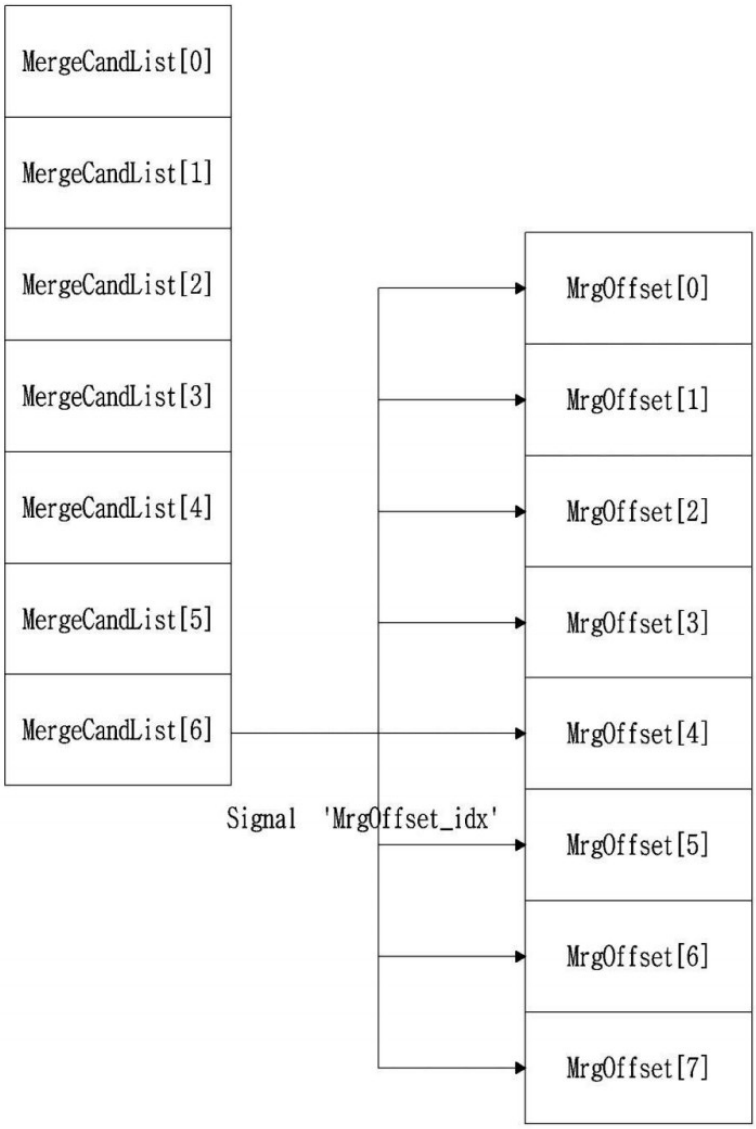

MergeCandList[0]
MergeCandList[1]
MergeCandList[2]
MergeCandList[3]
MergeCandList[4]
MergeCandList[5]
MergeCandList[6]
Signal 'MrgOffset_idx'
MrgOffset[0]
MrgOffset[1]
MrgOffset[2]
MrgOffset[3]
MrgOffset[4]
MrgOffset[5]
MrgOffset[6]
MrgOffset[7]

Fig. 33

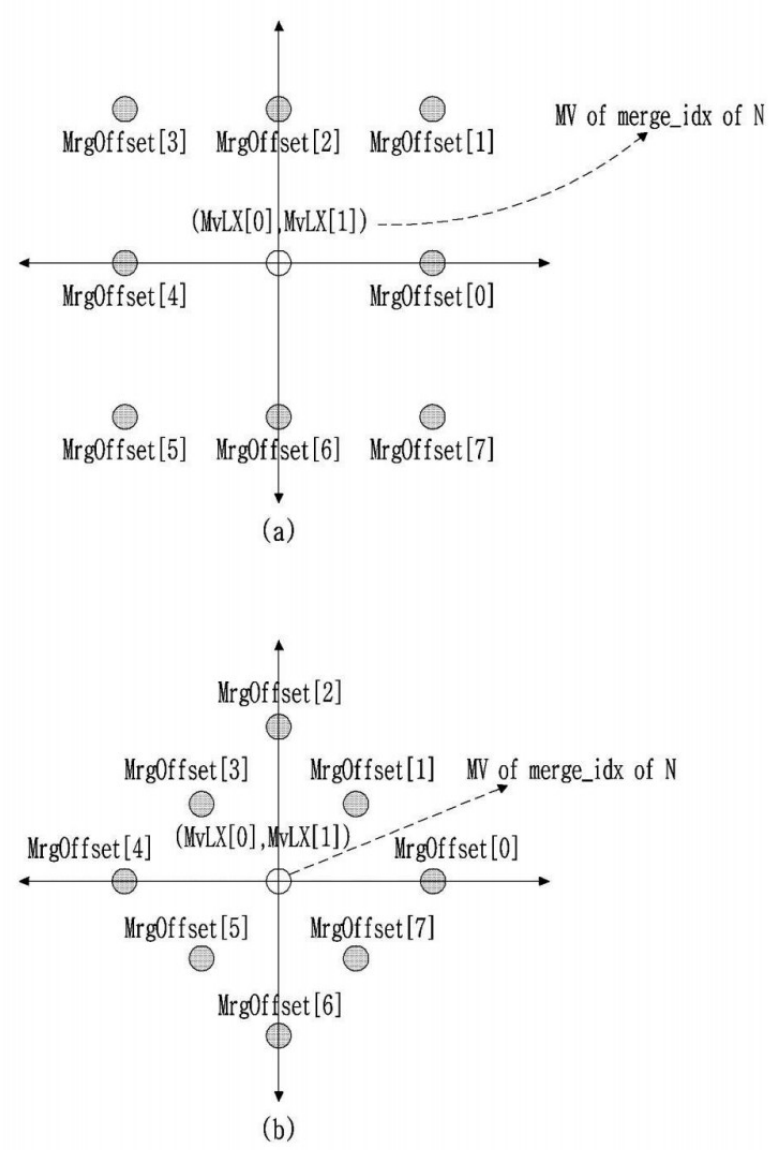

MrgOffset[3]
MrgOffset[2]
MrgOffset[1]
MV of merge_idx of N
(MvLX[0],MvLX[1])
MrgOffset[4]
MrgOffset[0]
MrgOffset[5]
MrgOffset[6]
MrgOffset[7]
(a)
MrgOffset[2]
MrgOffset[3]
MrgOffset[1]
MV of merge_idx of N
(MvLX[0],MvLX[1])
MrgOffset[4]
MrgOffset[0]
MrgOffset[5]
MrgOffset[7]
MrgOffset[6]
(b)

Fig. 34

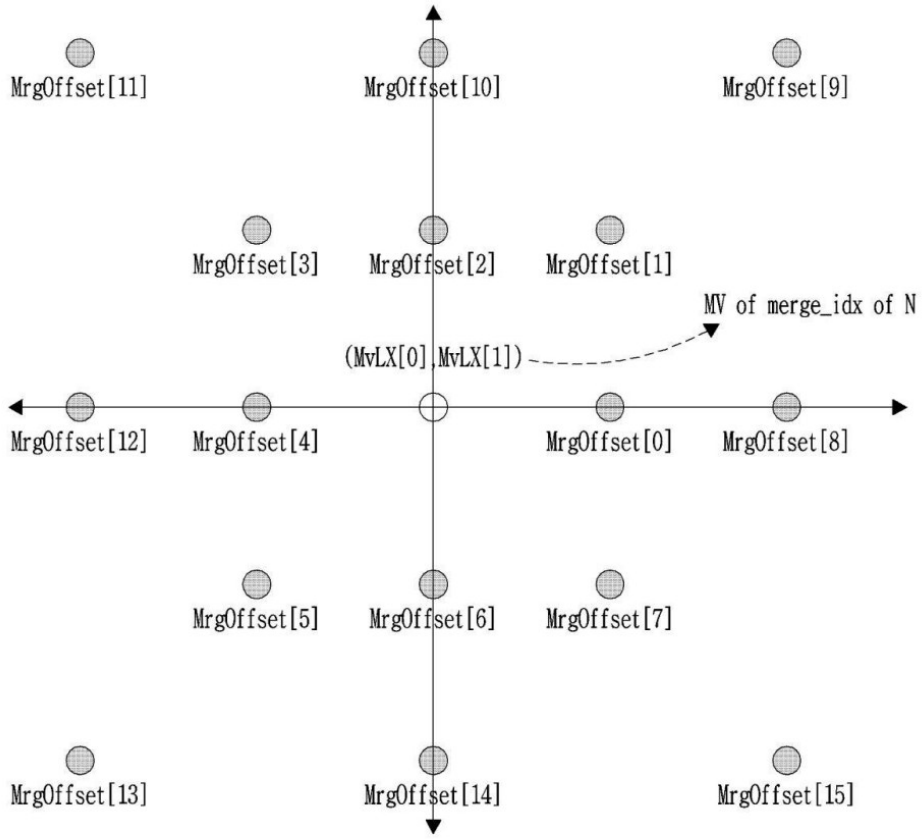

MrgOffset[11]
MrgOffset[10]
MrgOffset[9]
MrgOffset[3]
MrgOffset[2]
MrgOffset[1]
MV of merge_idx of N
(MvLX[0], MvLX[1])
MrgOffset[12]
MrgOffset[4]
MrgOffset[0]
MrgOffset[8]
MrgOffset[5]
MrgOffset[6]
MrgOffset[7]
MrgOffset[13]
MrgOffset[14]
MrgOffset[15]

Fig. 35

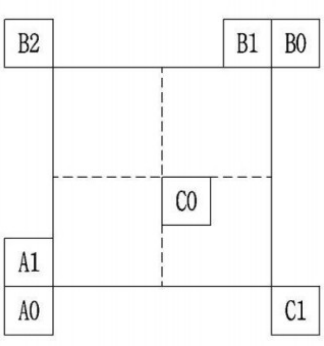

B2
B1
B0
C0
A1
A0
C1

Fig. 36

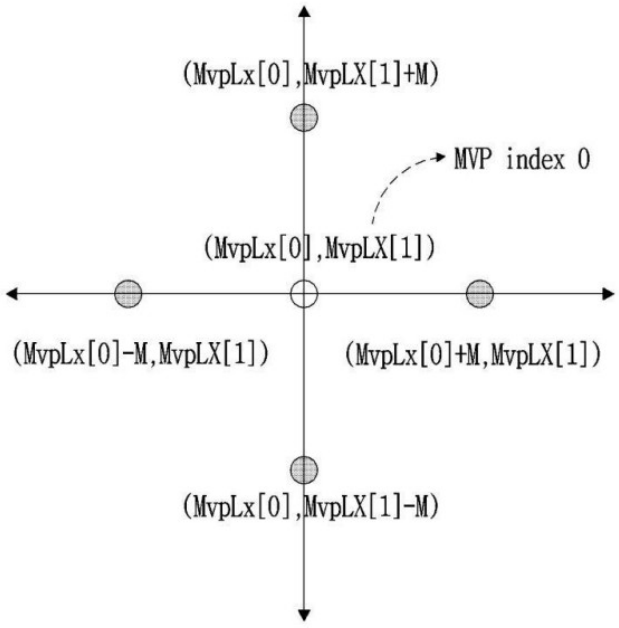

(MvpLx[0],MvpLX[1]+M)
MVP index 0
(MvpLx[0],MvpLX[1])
(MvpLx[0]-M,MvpLX[1])
(MvpLx[0]+M,MvpLX[1])
(MvpLx[0],MvpLX[1]-M)

Fig. 37

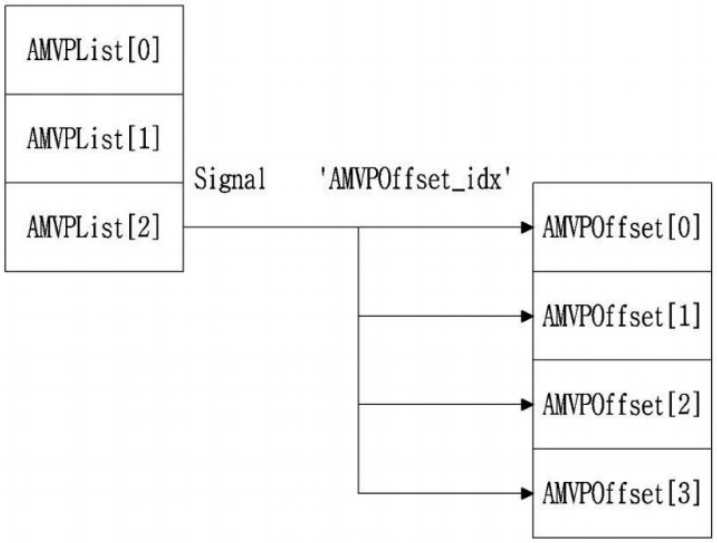

AMVPList[0]
AMVPList[1]
AMVPList[2]
Signal 'AMVPOffset_idx'
AMVPOffset[0]
AMVPOffset[1]
AMVPOffset[2]
AMVPOffset[3]

Fig. 38