

# 知覚ハッシュを利用したVVCの動き推定の検討

阿久津 叶玖<sup>a)</sup> 新田 高庸<sup>b)</sup>

**概要:** 本論文では、映像符号化の動き推定における新たな評価指標として、Average hash (aHash)、Perceptual hash (pHash)、Wavelet hash (wHash) の3種類の知覚ハッシュの適用性を評価する。その有効性を検証するため、2段階の実験を実施した。第1の実験では、方向別の画素シフト量に対するXORハミング距離の変化を測定した。その結果、3手法すべてにおいてシフト量に応じた距離の増加傾向が見られ、動きの変位を捉える知覚特性が確認された。第2の実験では、従来の指標であるSum of Absolute Difference (SAD)を基準とし、VVCのRandom Access Group of Pictures構造に準拠した動き推定実験を行った。その結果、いずれのハッシュ手法においても、SADによる探索結果に対してNeuralHashよりも高い一致率を示し、計測時間においても3つの知覚ハッシュが優位性を示せた。実験で得られた結果は、これら3つの知覚ハッシュが動き推定の代替指標として有効に機能することを示している。

## 1. 導入

インターネット上のストリーミング技術や通信インフラストラクチャーの発展に伴い、高効率な映像符号化技術の重要性は一層高まっている。これに伴い、映像符号化技術には従来以上に高度な圧縮性能と処理効率が求められている。こうした課題に対応するために、Versatile Video Coding (VVC) [1]の導入が必要である。

VVCは2020年に標準化された映像符号化規格であり、前身であるHigh Efficiency Video Coding (HEVC) [2]と比較して、約40%の圧縮効率向上を達成している。(図. 1)

一方で、VVCを広く普及させるためにはハードウェア実装が不可欠であるが、高い計算負荷が障害となっており、実用化に向けては計算量削減が重要な課題となっている。

計算負荷が高い要因の1つとして、動き推定が挙げられる。VVCの動き推定では、ブロック間の類似度評価にSAD (Sum of Absolute Differences) [4]が一般に用いられる。しかし、SADの計算量はブロックサイズに依存するため、大きなブロックサイズを扱うVVCでは演算量の増大が課題である。

この課題に対し、先行研究 [5] では、NeuralHashの知覚特性を示し、VVCの参照ソフトウェアであるVTMにおける動き推定の探索アルゴリズムである、TZsearch内のラスタ探索の評価指標として、ブロックサイズに依存しな



図 1 VVC と HEVC の RD カーブ [3]

い NeuralHash [6] を適用する試みがなされた。その結果、SADを用いた場合と同程度の動画品質が得られることが報告されている。一方で、NeuralHash自体の計算コストは高く、動き推定全体の計算負荷低減には課題が残されている。さらに、軽量化のための量子化を行った場合には、知覚ハッシュとしての識別性能が低下し、動き推定への適用が困難になることが確認された。

そこで本研究では、NeuralHashに加えて、古典的な知覚ハッシュ手法であるAverage Hash (aHash)、Perceptual Hash (pHash)、Wavelet Hash (wHash) [7]を対象とし、2つの実験を行った。

実験1では、知覚ハッシュの特性を調べるために、画像を上下左右の4方向に対して0ピクセルから20ピクセルまで1ピクセル刻みでシフトさせ、各手法のハッシュ値がどのように変化するかを評価した。その結果、4手法すべてでピクセルのシフト量が大きくなるほど元画像とのハッシュ

<sup>1</sup> 会津大学  
University of Aizu, Aizu-Wakamatsu, Fukushima, Japan

<sup>a)</sup> m5291004@u-aizu.ac.jp

<sup>b)</sup> koyo@u-aizu.ac.jp

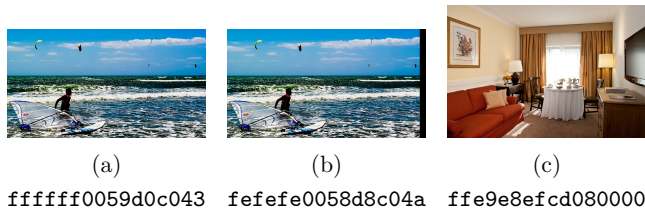


図 2 aHash(hash\_size=8) による 3 画像の比較と各ハッシュ値

(a) と (b) の XOR ハミング距離比較 : 7  
(a) と (c) の XOR ハミング距離比較 : 26

図 3 3 画像間の XOR ハミング距離の比較

値の差が増加し、知覚特性が確認された。

実験 2 では、3 つのデータセットを用いて VTM の実装に近い条件で動き推定を行い、SAD が求めた動きベクトルを正解データとし、NeuralHash 及び知覚ハッシュの一致率また不一致時の探索位置の分布を評価した。その結果、計算時間及び SAD との一致率において aHash, pHash, wHash はいずれも NeuralHash を上回る結果となった。

本論文の構成を以下に示す。第 2 章では手法について記述し、第 3 章では実験について説明を行い、第 4 章では結論を記述する。

## 2. 提案手法

このセクションでは提案手法である、Average Hash (aHash), Perceptual Hash (pHash), Wavelet Hash (wHash) について説明する。

### A. Average Hash

aHash は、画像の輝度情報の平均値を基準として、各画素が平均より明るい暗いかを 1 と 0 で表現する手法である。aHash は、画像をグレースケールに変換し、指定したサイズにリサイズする。次に、全画素の輝度値の平均を計算する。その後、各画素の値が平均値以上なら 1、未満なら 0 と判定する。ハッシュ値を構成するビット数は、指定したリサイズによって決定する。

### B. Perceptual Hash

pHash は、DCT (離散コサイン変換) によって低周波成分を抽出し、そこから画像の特徴を得る手法である。pHash は、画像をグレースケールに変換し、指定のサイズにリサイズする。次に、リサイズ後の画像に対して 2 次元 DCT を適用する。その後、DCT 係数の左上領域にある低周波成分を指定サイズで切り出す。さらに、抽出した係数の中央値を求め、中央値以上を 1、それ未満を 0 として二値化する。ハッシュ値を構成するビット数は、DCT 後の指定サイズによって決定する。

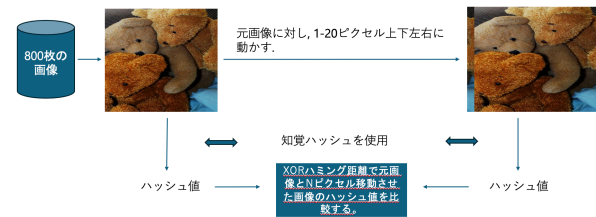


図 4 知覚ハッシュ特性実験イメージ

## C. Wavelet Hash

wHash は、画像を DWT (離散ウェーブレット変換) によって分解し、低周波成分から特徴を抽出する手法である。wHash は、画像をグレースケールに変換し、サイズを 2 のべき乗にリサイズする。次に、DWT の分解回数を決定する。その後、2 次元 DWT を適用し、画像を LL, LH, HL, HH 成分に分解する。さらに、LL 成分を取り出し、その中央値を基準に 1 と 0 へ二値化し、ハッシュ値の生成を行う。

知覚ハッシュの評価には、XOR 演算で求めたハミング距離を用いる。XOR ハミング距離は、2 つの知覚ハッシュ値の各ビットを XOR で比較し、異なるビット数を数えた距離である。値が小さいほど画像同士が類似しており、値が大きいほど異なることを示す。

図 2(a) は、元画像に対して aHash を前述の指定サイズ 8 で適用した結果である。このとき、出力ビット数は  $8 \times 8$  (64 ビット) となる。図 2(b) は、元画像の右端 20 ピクセルを黒色に塗りつぶした画像であり、(a) との XOR ハミング距離は 7 である。図 2(c) は、見た目が大きく異なる画像であり、(a) との XOR ハミング距離は 26 である。さらに、図 3 には (a) と (b)、および (a) と (c) の XOR ハミング距離を比較した結果を示した。これより、類似した画像ほど XOR ハミング距離は小さく、異なる画像ほど大きくなることが確認できる。

## 3. 実験

このセクションでは、提案手法の有効性を評価するために 2 つの実験を行った。実験 1 では、画像を上下左右にシフトさせたときの知覚ハッシュ値の変化を調べる実験である。実験 2 では、知覚ハッシュを用いた動き推定であり、SAD との完全一致率、不一致時の探索位置の分布を調べる実験である。

### 3.1 実験 1: 知覚ハッシュ特性実験

ここでは NeuralHash, aHash, pHash, wHash を用いて、COCO データセットの 800 枚の画像 [8] に対するハッシュ値の変化を評価した。具体的には、各画像について元画像のハッシュ値を基準に、上下左右の 4 方向へ 0 ピクセルから 20 ピクセルまで 1 ピクセル刻みでシフトを与え、元画像との XOR ハミング距離を算出した。シフト後の画像は、欠

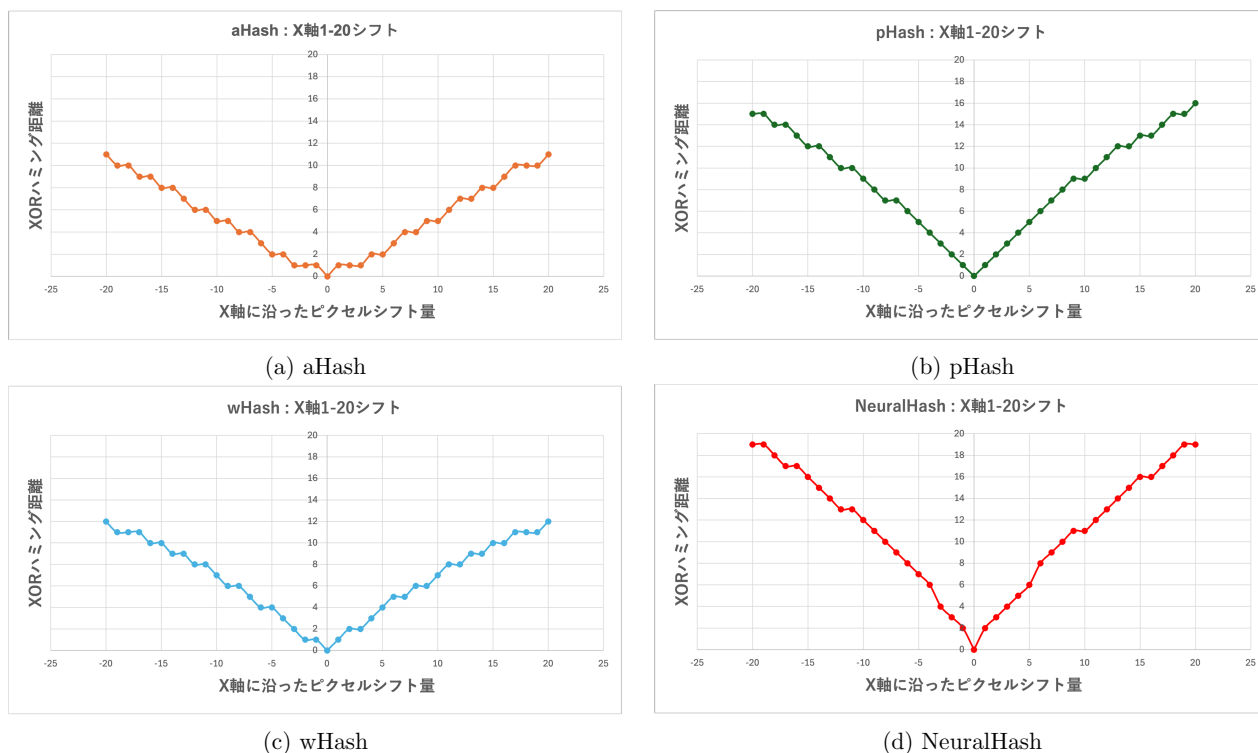


図 5 知覚ハッシュ特性実験結果 (X 方向)

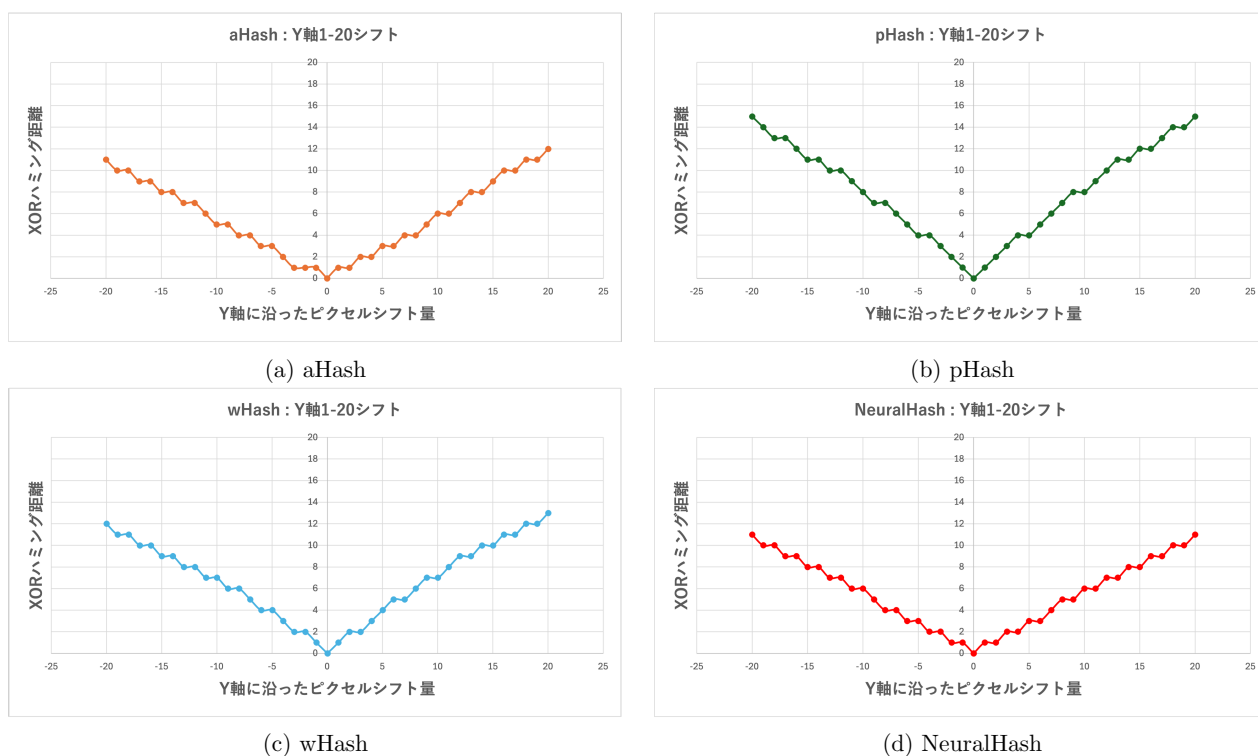


図 6 知覚ハッシュ特性実験結果 (Y 方向)

損画素を反対側へ回り込ませることで画像サイズを保持したまま生成した。その後、同一の方向およびシフト量ごとに 800 枚分の XOR ハミング距離の平均値を算出した。実験の概要を図. 4 に示す。

#### 実験結果

図. 5 および図. 6 に示す。4 手法のいずれも、X、Y 軸の

シフト量が大きくなるほど XOR ハミング距離が増加した。これは、画素のずれを捉える知覚特性を示すこととなる。

#### 3.2 実験 2: 知覚ハッシュを用いた動き推定

実験 2 では、提案手法及び、NeuralHash を用いて動き推定を行った。ここで完全一致率とは、知覚ハッシュが選択



(a) uav0000339\_00001.v (jpg 形式) [9]



(b) Cactus (yuv 形式)



(c) Basketball (yuv 形式) [10]

図 7 実験 2 で用いたフレーム

した動きベクトルが SAD の動きベクトルと完全一致した割合を指す. 完全一致しなかった場合には, SAD の最適位置との差をユークリッド距離で求め, 0, 1-8, 9-16, 17-32, 33 超の 5 区間に分類して割合を算出し, その時の SAD との 2 乗誤差の総和を求めた. 2 乗誤差の総和の数式は次の通りである.  $mv\_SAD_i$  は SAD の動きベクトルを指し,  $mv\_Hash_i$  は, 3 つの知覚ハッシュと NeuralHash を示す.

$$\sum_{i=1}^N |mv\_SAD_i - mv\_Hash_i|^2$$

実験には 3 つのデータセットを用いた. 図. 7 に示す (a) は jpg 形式, (b) および (c) は yuv 形式である. この 3 つのデータセットは全て 33 枚用いる.

フレーム構造には, ランダムアクセスを想定した GOP サイズ 32 の階層的 B ピクチャ構造を採用した.

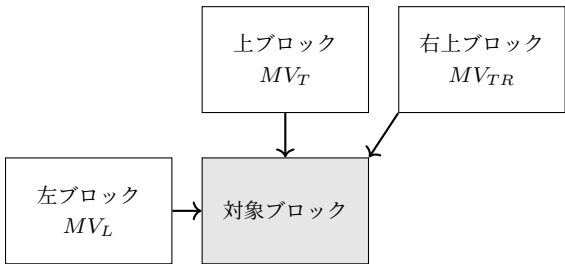
知覚ハッシュとして用いた aHash, pHash, wHash は, いずれも指定のリサイズ処理は行わず, 入力画像をそのまま入力し, グレースケール化を行った. aHash の出力ハッシュ長は 16384 ビットである. pHash と wHash は, 精度比較のため  $8 \times 8$  (64 ビット),  $16 \times 16$  (256 ビット),  $32 \times 32$  (1024 ビット) の 3 条件を設定した. (表 1)

対象画像及び参照画像のブロックサイズとストライドの条件は表 2 である.

探索条件として, 今回はラスタ探索を行い探索範囲は VTM のデフォルト値である 384 に設定した. また, ラスタ探索における初期点 (探索中心点) の決定には, 近傍ブロックの動きベクトルの中央値を用いる空間予測を採用した. 図. 8 に示す左ブロック, 上ブロック, 右上ブロックの 3 つのブロックの動きベクトルを参照する. ここで,  $MV_L$ ,  $MV_T$ ,  $MV_{TR}$  はそれぞれ左ブロック, 上ブロック, 右上ブロックの動きベクトルを表す. これら 3 つの動きベクトルの各成分の中央値を求めることで予測動きベクトル  $MV_{pred}$  を算出し, これをラスタ探索の初期点として用いた.

#### 実験結果

実験結果として, 計算時間を表 3 - 表 5 に示した. また, SAD との動きベクトルの 2 乗誤差の総和を表 6 - 表 8 に示した. さらに, 3 つのデータセットにおける SAD との差分距離の分布を表 9 に示した.



$$MV_{pred} = \text{median}(MV_L, MV_T, MV_{TR})$$

図 8 空間予測に用いる近傍ブロック

計算時間の結果から 提案手法である 3 つの知覚ハッシュは NeuralHash の計算時間を大幅に向上している. aHash においては, 事前計算は単純な平均値のみで計算している

表 1 各ハッシュ手法の設定比較

手法	リサイズ	変換手法	出力条件
aHash	なし	平均値比較	$128 \times 128$ (16384 ビット)
pHash	なし	DCT	$8 \times 8$ (64 ビット) $16 \times 16$ (256 ビット) $32 \times 32$ (1024 ビット)
wHash	なし	DWT	$8 \times 8$ (64 ビット) $16 \times 16$ (256 ビット) $32 \times 32$ (1024 ビット)
NeuralHash	$360 \times 360$	ニューラルネットワーク	96 ビット

表 2 対象画像と参照画像におけるブロック生成条件

画像種別	ブロックサイズ	ストライド	計算処理
対象画像	$128 \times 128$	128	その場でハッシュ計算
参照画像	$128 \times 128$	8, 4	候補位置の事前計算



ため事前計算は速いが、ビット数が  $128 \times 128$  (16,384 ビット) であるため動き推定にかかる時間が長いことが確認できる。

次に、3 データセットにおける SAD との差分距離の分布である。0 は SAD との動きベクトルが完全一致していることを指す。このことを踏まえ、結果を見ると 3 つの知覚ハッシュは NeuralHash よりも完全一致してる割合が高いことが示された。また、p,wHash に関しては、基本的にビット数が多くなるごとに精度の向上が見られる。SAD との差分距離分布を見ると、完全一致率が低いほど 1-8 ピクセルの近い範囲での誤差の割合が高く見られた。

最後に、SAD との動きベクトルの二乗誤差の総和を示す。データセット uav0000339\_00001\_v では、完全一致率の高い aHash や pHash  $32 \times 32$  においても、比較的大きな値を示した。これは、完全一致しなかった場合に、動きベクトルが SAD から大きく外れることを示唆している。一方、他の 2 つのデータセットでは、aHash の二乗誤差の総和は比較的小さかった。これらの結果から、動き推定における知覚ハッシュの有効性は、画像列の特性に依存することがわかる。

## 4. 結論

本研究では、aHash, pHash, wHash を対象に、VVC の動き推定における知覚ハッシュの有効性を 2 つの実験で検討した。実験 1 では、画像を上下左右にシフトさせた際の

表 3 uav0000339\_00001\_v における各ハッシュ手法の事前計算時間と動き推定時間 (stride  $8 \times 8$  /  $4 \times 4$ )

手法	$8 \times 8$		$4 \times 4$	
	事前計算 [s]	動き推定 [s]	事前計算 [s]	動き推定 [s]
aHash $128 \times 128$	96	364	378	1,522
pHash $8 \times 8$	175	163	707	622
pHash $16 \times 16$	173	157	696	617
pHash $32 \times 32$	179	179	721	689
wHash $8 \times 8$	343	157	1,377	607
wHash $16 \times 16$	318	157	1,279	610
wHash $32 \times 32$	293	186	1,180	722
NeuralHash	6,056	246	24,230	700
SAD	-	3,817	-	15,144

表 4 cactus における各ハッシュ手法の事前計算時間と動き推定時間 (stride  $8 \times 8$  /  $4 \times 4$ )

手法	$8 \times 8$		$4 \times 4$	
	事前計算 [s]	動き推定 [s]	事前計算 [s]	動き推定 [s]
aHash $128 \times 128$	96	345	372	1,397
pHash $8 \times 8$	157	143	702	644
pHash $16 \times 16$	166	148	708	628
pHash $32 \times 32$	185	155	735	719
wHash $8 \times 8$	352	170	1,411	652
wHash $16 \times 16$	307	150	1,192	584
wHash $32 \times 32$	303	189	1,209	733
NeuralHash	6,505	258	23,145	700
SAD	-	1,301	-	5,093

XOR ハミング距離を評価し、いずれの手法でも X,Y のシフト量の増加に応じて距離が増加することを確認した。

実験 2 では、SAD の動きベクトルを正解とした動き推定を行い、uav0000339\_00001\_v では、a,p,wHash 全てが約 70% 程度の完全一致率を示した。cactus では、3 つの知覚ハッシュが 90% 以上を示し、basketball では一致率が他のデータセットと比較すると低かったものの、NeuralHash を超える完全一致率を示すことができた。また、事前ハッシュの計算速度も大幅に NeuralHash の計算時間を上回った。この結果から古典的な知覚ハッシュ手法は、軽量の動き推

表 5 basketball における各ハッシュ手法の事前計算時間と動き推定時間 (stride  $8 \times 8$  /  $4 \times 4$ )

手法	$8 \times 8$		$4 \times 4$	
	事前計算 [s]	動き推定 [s]	事前計算 [s]	動き推定 [s]
aHash $128 \times 128$	78	316	322	1,280
pHash $8 \times 8$	151	139	593	550
pHash $16 \times 16$	150	149	602	602
pHash $32 \times 32$	155	167	633	667
wHash $8 \times 8$	295	141	1,174	757
wHash $16 \times 16$	328	160	1,316	628
wHash $32 \times 32$	255	164	1,018	627
NeuralHash	5,770	258	23,065	698
SAD	-	1,181	-	4,687

表 6 uav0000339\_00001\_v における SAD との動きベクトルの 2 乗誤差の総和

手法	stride 8	stride 4
aHash $128 \times 128$	163,116,288	154,280,528
pHash $8 \times 8$	42,870,016	73,121,568
pHash $16 \times 16$	36,901,632	29,031,456
pHash $32 \times 32$	177,965,888	38,926,672
wHash $8 \times 8$	51,643,776	43,532,624
wHash $16 \times 16$	53,277,312	46,737,584
wHash $32 \times 32$	76,443,776	68,272,800
NeuralHash	46,524,032	45,649,376

表 7 Cactus における SAD との動きベクトルの 2 乗誤差の総和

手法	stride 8	stride 4
aHash $128 \times 128$	2,841,984	2,949,040
pHash $8 \times 8$	181,472,000	135,977,872
pHash $16 \times 16$	130,385,344	166,707,408
pHash $32 \times 32$	135,970,496	202,442,960
wHash $8 \times 8$	164,527,680	206,595,088
wHash $16 \times 16$	86,172,800	72,516,928
wHash $32 \times 32$	5,336,832	14,846,880
NeuralHash	129,108,928	121,897,168

表 8 Basketball における SAD との動きベクトルの 2 乗誤差の総和

手法	stride 8	stride 4
aHash $128 \times 128$	22,022,976	39,548,368
pHash $8 \times 8$	209,536,192	181,269,200
pHash $16 \times 16$	100,496,768	141,227,680
pHash $32 \times 32$	72,704,384	141,544,224
wHash $8 \times 8$	149,413,568	175,532,928
wHash $16 \times 16$	84,177,472	134,684,640
wHash $32 \times 32$	51,647,104	73,364,176
NeuralHash	231,780,288	234,108,992

表 9 3 データセットにおける aHash 128x128, pHash/wHash 8x8-32x32, NeuralHash の SAD との差分距離分布

Method	Stride (N)	0	1-8	9-16	17-32	33 以上
<b>uav0000339_00001.v</b>						
aHash 128x128	8	70.3%	3.3%	2.0%	2.1%	22.3%
	4	69.8%	4.9%	2.1%	2.3%	21.0%
pHash 8x8	8	63.2%	8.2%	4.7%	3.8%	20.1%
	4	50.4%	21.2%	5.2%	3.8%	19.5%
pHash 16x16	8	70.3%	4.7%	3.1%	4.0%	17.9%
	4	66.9%	10.2%	2.7%	4.1%	16.2%
pHash 32x32	8	69.4%	2.7%	2.1%	2.9%	22.9%
	4	72.4%	6.1%	2.2%	3.1%	16.2%
wHash 8x8	8	53.3%	11.7%	5.9%	5.6%	23.6%
	4	41.9%	22.4%	7.8%	6.1%	21.9%
wHash 16x16	8	63.1%	6.9%	3.8%	4.9%	21.4%
	4	55.0%	16.3%	3.9%	5.0%	19.8%
wHash 32x32	8	70.5%	3.9%	2.5%	3.5%	19.7%
	4	67.7%	7.0%	3.0%	3.8%	18.5%
NeuralHash	8	33.8%	19.4%	11.1%	11.6%	24.1%
	4	21.8%	29.2%	14.3%	11.1%	23.6%
<b>cactus</b>						
aHash 128x128	8	98.0%	0.3%	0.3%	0.4%	0.9%
	4	96.8%	1.3%	0.4%	0.4%	1.1%
pHash 8x8	8	72.8%	2.8%	0.8%	0.5%	23.1%
	4	56.2%	15.4%	0.7%	0.6%	27.1%
pHash 16x16	8	87.4%	0.1%	0.8%	0.1%	11.6%
	4	79.9%	1.9%	0.5%	0.2%	17.5%
pHash 32x32	8	90.9%	0.2%	0.5%	0.2%	8.2%
	4	84.5%	0.7%	0.2%	0.3%	14.3%
wHash 8x8	8	51.7%	5.7%	2.0%	1.7%	38.9%
	4	36.6%	14.6%	3.3%	1.5%	43.9%
wHash 16x16	8	86.7%	1.2%	1.5%	1.1%	9.5%
	4	79.3%	6.1%	1.1%	1.4%	12.1%
wHash 32x32	8	94.5%	1.2%	1.0%	0.7%	2.5%
	4	92.2%	2.8%	0.7%	0.9%	3.4%
NeuralHash	8	20.8%	13.0%	8.4%	7.8%	49.9%
	4	12.8%	19.7%	11.7%	7.7%	48.1%
<b>basketball</b>						
aHash 128x128	8	67.3%	7.9%	8.1%	6.9%	9.8%
	4	63.0%	9.4%	3.5%	12.5%	11.6%
pHash 8x8	8	30.3%	8.6%	8.2%	10.0%	42.9%
	4	20.9%	15.1%	4.9%	14.4%	44.6%
pHash 16x16	8	46.2%	5.5%	7.2%	9.2%	31.9%
	4	39.2%	7.7%	2.8%	13.2%	37.2%
pHash 32x32	8	47.4%	10.3%	4.4%	11.2%	26.7%
	4	43.6%	8.8%	5.4%	13.5%	28.7%
wHash 8x8	8	24.1%	10.6%	7.1%	10.8%	47.4%
	4	16.9%	14.5%	5.9%	15.5%	47.2%
wHash 16x16	8	42.8%	10.5%	8.1%	9.1%	29.5%
	4	35.4%	12.8%	4.4%	13.4%	34.0%
wHash 32x32	8	56.2%	8.4%	9.5%	8.8%	17.2%
	4	50.7%	11.5%	4.0%	13.8%	20.0%
NeuralHash	8	6.9%	9.3%	7.3%	12.4%	64.1%
	4	3.4%	10.2%	8.8%	14.0%	63.7%

定の代替指標として有効であることが示された。

## 参考文献

- [1] Bross, et al., “Overview of the Versatile Video Coding (VVC) Standard and its Applications,” IEEE Trans. on Circuits and Systems for Video Technology, vol. 31, no. 10, pp. 3736–3764, October 2021.
- [2] J. Lainema, et al., “Intra Coding of the HEVC Standard”
- [3] OTTVerse. (2024). “What is VVC (H.266) - Versatile Video Coding and How Does It Compare to HEVC (H.265)?”
- [4] N. D. Vu, et al., “High speed SAD architecture for variable block size motion estimation in HEVC encoder”
- [5] Ryota Igarashi. (2024). “Motion Estimation Using NeuralHash for VVC Hardware Implementation,” graduation thesis, University of Aizu.
- [6] Apple, “CSAM Detection Technical Summary” [https://www.apple.com/child-safety/pdf/CSAM\\_Detection\\_Technical\\_Summary.pdf](https://www.apple.com/child-safety/pdf/CSAM_Detection_Technical_Summary.pdf)
- [7] Johannes Buchner, “imagehash” <https://github.com/JohannesBuchner/imagehash?>

`tab=readme-ov-file`

- [8] Common Objects in Context  
<https://cocodataset.org/#download>
- [9] P. Zhu, L. Wen, D. Du, X. Bian, H. Fan, Q. Hu, H. Ling, “Detection and Tracking Meet Drones Challenge,” IEEE Transactions on Pattern Analysis and Machine Intelligence, 2021.
- [10] ITE/ARIB Hi-Vision Test Sequence 2nd Edition,  
[https://www.ite.or.jp/contents/chart/  
manual-rev1.3.pdf](https://www.ite.or.jp/contents/chart/manual-rev1.3.pdf)