A DYNAMIC MOTION VECTOR REFERENCING SCHEME FOR VIDEO CODING

Jingning Han, Yaowu Xu, and James Bankoski

WebM Codec Team, Google Inc.
1600 Amphitheatre Parkway, Mountain View, CA 94043
Emails: {jingning,yaowu,jimbankoski}@google.com

ABSTRACT

Video codecs exploit temporal redundancy in video signals, through the use of motion compensated prediction, to achieve superior compression performance. The coding of motion vectors takes a large portion of the total rate cost. Prior research utilizes the spatial and temporal correlation of the motion field to improve the coding efficiency of the motion information. It typically constructs a candidate pool composed of a *fixed* number of reference motion vectors and allows the codec to select and reuse the one that best approximates the motion of the current block. This largely disconnects the entropy coding process from the block's motion information, and throws out any information related to motion consistency, leading to sub-optimal coding performance. An alternative motion vector referencing scheme is proposed in this work to fully accommodate the dynamic nature of the motion field. It adaptively extends or shortens the candidate list according to the actual number of available reference motion vectors. The associated probability model accounts for the likelihood that an individual motion vector candidate is used. A complementary motion vector candidate ranking system is also presented here. It is experimentally shown that the proposed scheme achieves about 1.6% compression performance gains on a wide range of test clips.

Index Terms— Reference motion vector, motion compensated prediction, block merging

1. INTRODUCTION

Motion compensated prediction is widely used by modern video codecs to reduce the temporal correlations of video signals. A typical framework breaks the frame into rectangular or square blocks of variable sizes and applies either motion compensated or intra frame prediction to each individual block. All these decisions are made using rate-distortion optimization. In a well-behaved encoder, these blocks along with the associated motion vectors should largely resemble the actual moving objects [1]. Due to the irregularity of the moving objects in natural video content and the on-grid block partition constraint, a large amount of the prediction blocks share the same motion information with their spatial or temporal neighbors. Prior research exploits such correlations to improve coding efficiency [2]-[4].

A derived motion vector mode named *direct mode* is proposed in [5] for bi-directional predicted blocks. Unlike the conventional inter block that needs to send the motion vector residual to the decoder, this mode only infers motion vector from previously coded blocks. To determine the motion vector for each reference frame, the scheme checks the neighboring blocks in the order of above, left, top-right, and top-left, picks the first one that has the same reference frame, and reuses its motion vector. A rate-distortion optimization approach that allows the codec to select between two reference motion vector candidates is proposed in [6].

The derived motion vector approach has been extended to the context of single reference frame, where the codec builds a list of motion vector candidates by searching the previously coded spatial and temporal neighboring blocks in the ascending order of the relative distance. The candidate list is fixed per slice/frame, and the encoder can select any candidate and send its index to the the decoder. If the number of reference motion vectors found is more than the list length, the tail candidates will be truncated. If the candidates found are insufficient to fill the list, zero motion vectors will be appended. It has been successfully incorporated into later generation video codecs including HEVC [4, 7] and VP9 [8, 9].

An alternative motion vector referencing scheme is presented in this work. It is inspired by the observation that the precise neighboring block information is masked by the fixed-length candidate list structure, which can potentially cause sub-optimal coding performance. Instead, we propose a dynamic motion vector referencing mode, where the candidate list can be adaptively extended or shortened according to the number of reference motion vectors found in the search region. The list is then ranked based on their likelihood to be chosen. These likelihood metrics are also employed as the contexts for the index probability modeling. An accompanying ranking system that accounts for both relative distance and popularity factors, while keeping the decoder complexity under check, will be discussed next. The proposed scheme has been integrated into the VP9 framework. Experimental results demonstrate that it achieves considerable compression performance gains over the conventional fixed-length approach.

2. CANDIDATE LIST CONSTRUCTION

An inter coded block can be predicted from either a single reference frame or a pair of compound reference frames. We discuss the candidate motion vector list construction and ranking process in these two settings respectively.

2.1. Single Reference Frame Mode

The scheme searches the candidate motion vectors from previously coded blocks, with a step size of 8×8 block. It defines the nearest spatial neighbors, i.e., immediate top row, left column, and top-right corner, as category 1. The outer regions (maximum three 8×8 blocks away from the current block boundary) and the collocated blocks in the previously coded frame are classified as category 2. The neighboring blocks that are predicted from different reference frames or are intra coded are pruned from the list. The remaining reference blocks are then each assigned a weight. The weight is obtained by calculating the overlap length between the reference block and the current block, where the overlap length is defined as the projection length of the reference block onto the top row or left column of the current block. If the two reference blocks use an identical motion vector, they will be merged as a single candidate, whose weight is the sum of the two individual weights. If these two blocks are in different category regions, the merged one will assume the smaller category index.

The scheme ranks the candidate motion vectors in descending order by their weights within each category. That means a motion vector from the nearest spatial neighbor always has a higher priority than those from the outer region or the collocated blocks in the previous frame. An example of the ranking process for single reference frame case is depicted in Fig. 1. The weight and category information will be used for the entropy coding process.

2.2. Compound Reference Frame Mode

Assume that the current block is predicted from a pair of reference frames (rf0, rf1). The scheme first looks into the neighboring region for reference blocks that share the same reference frame. The corresponding motion vectors are ranked as discussed in Sec. 2.1 and are placed on the top of the candidate list. (See an example in Fig. 2, where the symbol $\underline{mv1}$ denotes a pair of motion vectors.) It then checks the nearest spatial neighbors whose reference frame pair does not match (rf0, rf1), but has a single reference frame that matches either rf0 or rf1. In this situation, we build a list of motion vectors for reference frame rf0 and rf1, respectively. They are combined in an element-wise manner to synthesize a list of motion vector pairs associated with (rf0, rf1), as denoted by the symbol comp$(\underline{mv0}, \underline{mv1})$ in the example, which is then appended to the candidate list.

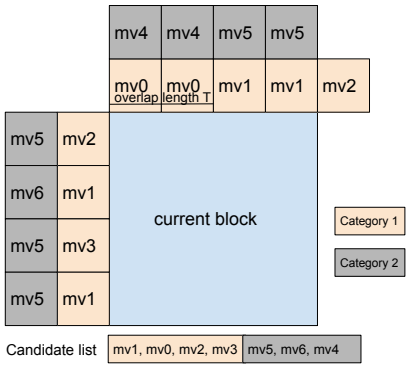

Fig. 1. Candidate motion vector list construction for a single reference frame.

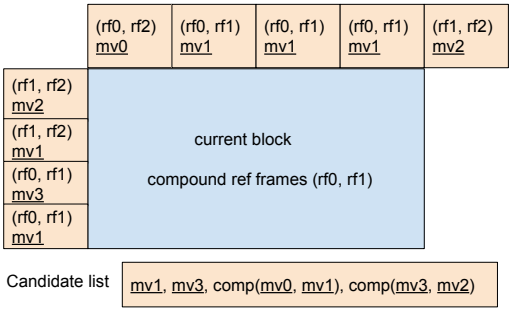

Fig. 2. Candidate motion vector list construction for compound reference frames.

3. DYNAMIC MOTION VECTOR REFERENCING MODE

Having established the candidate list, we now exploit their use in a dynamic motion vector referencing mode to improve the coding performance. The dynamic motion vector referencing mode refers to one of the candidates in the list as the effective motion vector, without the need to explicitly code it. Unlike the *fixed-length* candidate list used in HEVC [7] and VP9 [9], the scheme here builds on a dynamic length candidate list to fully exploit the available motion vector neighborhood information for better motion vector referencing and improved entropy coding performance.

We denote the dynamic motion vector referencing mode by REFMV. In this setting, the encoder evaluates the rate-distortion cost associated with each candidate motion vector, picks the one that provides minimum cost, and sends its index to the decoder as the effective motion vector for motion compensated prediction. Another inter mode where one needs to send the motion vector difference in the bit-stream is referred to as NEWMV mode. In this setting, the encoder runs a regular motion estimation to find the best motion vector and looks up a predicted motion vector closest to the effective motion vector from the candidate reference motion vector list. It then sends the index of the predicted motion vector as well as the

difference from the effective motion vector to the decoder. A special mode where it forces zero motion vector is named ZEROMV. An accompanying entropy coding system is devised here to improve the coding efficiency of transmitting the syntax elements in the bitstream.

The entropy coding of the flag that identifies the NEWMV mode from the derived motion vector mode requires a probability model conditioned on two factors: (1) the number of reference motion vectors found; (2) the number of reference blocks that are coded in NEWMV mode. The context is denoted by *ctx0* in Fig. 3. When the candidate list is empty or very short, it is unlikely to find a good match from the reference motion vector pool, which inversely makes it more likely to use NEWMV mode. Alternatively, if most of the candidate motion vectors are from reference blocks that are coded in NEWMV mode, one would assume that the region consists of intense motion activity and hence increase the likelihood to select NEWMV mode. The context *ctx0* can be retrieved according to the mapping function defined in Table 1.

Table 1. Probability model context *ctx0* mapping table.

mv candidate count	NEWMV count	ctx0
0	0	0
1	1	1
1	0	2
≥ 2	≥ 2	3
≥ 2	1	4
≥ 2	0	5

The coding for syntax that differentiates between ZEROMV and REFMV employs a probability model based on whether the candidate motion vectors are mostly zero vector or close to zero. The probabilities for the candidate index within the REFMV mode are distributed such that the category 1 indexes have higher probability than the category 2 ones. Within each category, if the weights are identical, the two indexes have the same probability; otherwise, the one with higher weight has higher probability. In practice, the probability models are conditioned on relative weight and category difference and are updated per frame. A similar mapping approach as Table 1 is used to translate these factors into the probability model contexts accordingly. All the probability models are updated per frame. In our implementation, this context based probability model for motion vector entropy coding requires the codec to maintain a probability table of 23 bytes.

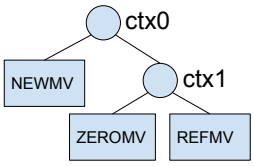

Fig. 3. Inter mode entropy coding tree.

4. EXPERIMENTAL RESULTS

We implemented the proposed dynamic motion vector referencing scheme in the VP9 framework. Baseline VP9 employs a fixed-length candidate list for motion vector referencing, where the codec searches the spatial and temporal neighboring reference blocks in a spiral order to find the nearest *two* reference motion vectors. Experiments show that increasing the fixed length to three provides very limited coding gains, i.e., around 0.2%. Similar observations have been found in the context of HEVC as well [4]. The baseline VP9 encoder supports recursive coding block partition ranging from 64×64 down to 4×4. Variable transform size ranging from 32×32 to 4×4 is selected per block. All the intra and inter prediction modes (including NEWMV and ZEROMV modes) are turned on by default. The coding decisions are made in a rate-distortion optimization framework.

We replaced the motion vector referencing mode based on fixed-length candidate list in VP9 with the proposed dynamic referencing approach. The entropy coding system for inter modes is modified accordingly as discussed in Sec. 3. Its compression performance is evaluated on a large set of test clips and over a wide range of target bit-rates, which typically cover the PSNR range between 30 dB to 46 dB. All the sequences are coded in 2-pass mode with an instantaneous refresh frame inserted every 150 frames. The coding gains over VP9 baseline are shown in Table 2-4. Note that a positive number means better compression performance. We further evaluate the coding performance at low, median, and high bit-rates as shown in the right three columns in Table 2-4, by evenly breaking the operating points of a test clip into three groups and computing the BD-rate savings respectively.

Our results indicate that gains are largely consistent across all the resolutions and frame rates. The lower bit-rate settings tend to have higher gains than higher bit-rates due to the fact that the rate cost on motion vector syntax takes much less percentage than the quantized coefficients in the high bit-rate settings. Video sequences with intense motion activities (i.e., those hard to compress) tend to gain more than those with static content, since the extended candidate list can provide more motion vector options for reference. Our local tests suggest the use of the dynamic motion vector referencing scheme increases the encoding complexity by 8% on average.

5. CONCLUSIONS

An advanced motion vector referencing scheme is proposed to capture the dynamic nature of a neighborhood of motion vectors. Accompanied by a motion vector ranking system, it allows the codec to fully utilize all the available motion information from previously coded neighboring blocks to improve the coding efficiency. It is experimentally demonstrated that the proposed approach provides considerable compression performance gains over the conventional motion vector referencing system based on fixed-length candidate list.

Table 2. Compression performance comparison of the dynamic motion vector referencing scheme with respect to the VP9 baseline. The gains are in terms of BD-rate reduction percentage.

	res	fps	avg (%)	low (%)	mid (%)	high (%)
akiyo	288p	25	1.014	1.081	1.004	0.975
bowing	288p	30	0.454	0.287	-0.013	1.310
bus	288p	30	2.002	2.871	1.444	0.839
cheer	240p	30	0.828	1.147	0.782	0.453
city	288p	25	2.089	2.265	1.983	1.753
coastguard	288p	30	1.244	2.049	0.869	0.554
container	288p	30	1.552	2.188	1.268	0.787
crew	288p	30	1.082	1.616	0.733	0.456
deadline	288p	30	1.370	1.915	0.826	0.699
flower	288p	30	1.518	2.241	1.236	0.737
football	288p	30	0.801	1.053	0.707	0.406
foreman	288p	30	2.288	3.039	2.044	1.126
hall	288p	30	0.934	1.492	0.844	0.390
harbour	288p	30	1.299	2.044	1.021	0.591
highway	288p	25	0.877	1.426	0.817	0.409
husky	288p	50	1.153	1.649	1.164	0.743
ice	288p	30	2.261	3.016	1.356	1.341
mobile	288p	30	1.475	1.846	1.297	0.984
mother	288p	25	1.523	1.996	1.128	0.729
news	288p	25	1.426	2.137	0.976	0.178
pamphlet	288p	25	2.695	1.949	-0.158	1.107
paris	288p	30	1.233	1.691	1.058	0.578
sign irene	288p	30	1.362	1.898	0.972	0.568
silent	288p	30	1.241	1.539	1.027	0.554
soccer	288p	30	1.454	1.852	1.212	0.891
stefan	288p	30	0.896	1.091	0.821	0.536
students	288p	30	1.413	1.948	0.925	0.749
tempete	288p	30	0.743	0.903	0.795	0.548
tennis	240p	30	0.653	0.963	0.541	0.283
waterfall	288p	50	1.433	2.167	0.917	0.557
OVERALL			1.344	1.779	0.987	0.728

Table 3. Compression performance comparison of the dynamic motion vector referencing scheme with respect to the VP9 baseline. The gains are in terms of BD-rate reduction percentage.

	res	fps	avg (%)	low (%)	mid (%)	high (%)
mobcal	720p	50	1.224	0.714	1.686	1.563
shields	720p	50	2.364	2.662	1.794	1.902
blue_sky	1080p	25	0.528	0.424	0.320	1.251
city	720p	50	2.348	3.406	1.267	1.218
crew	720p	50	1.133	1.553	0.985	0.556
crowd_run	1080p	50	2.188	3.198	1.361	0.992
cyclists	720p	50	1.774	2.235	1.034	1.416
jets	720p	50	3.048	3.211	1.902	1.953
night	720p	50	1.983	3.209	1.991	-0.544
old_town	720p	50	1.434	1.140	1.931	1.324
park_joy	1080p	50	2.218	2.760	1.928	1.552
pedestrian	1080p	30	1.917	2.256	1.520	1.696
riverbed	1080p	25	0.155	0.223	0.055	0.056
sheriff	720p	30	0.865	0.969	0.967	0.734
sunflower	1080p	25	3.036	3.162	1.954	3.468
OVERALL			1.748	2.075	1.380	1.276

Table 4. Compression performance comparison of the dynamic motion vector referencing scheme with respect to the VP9 baseline. The gains are in terms of BD-rate reduction percentage.

	res	fps	avg (%)	low (%)	mid (%)	high (%)
BQTerrace	1080p	60	2.346	2.666	2.259	2.469
BasketballDrive	1080p	50	1.317	2.263	1.066	0.883
Cactus	1080p	50	2.031	2.462	2.141	2.020
ChinaSpeed	720p	30	0.755	1.435	1.239	-0.804
FourPeople	720p	60	1.273	1.212	1.707	0.499
Johnny	720p	60	0.793	0.950	0.730	0.161
Kimono1	1080p	24	1.647	1.984	1.448	1.022
KristenAndSara	720p	60	0.915	1.262	1.748	-0.250
ParkScene	1080p	24	2.287	2.557	2.368	1.574
PeopleOnStreet	2k	30	2.386	3.644	1.762	1.029
SlideEditing	720p	30	1.403	1.955	0.658	0.491
SlideShow	720p	20	0.430	0.297	0.651	0.313
Tennis	1080p	20	1.659	1.937	1.601	1.004
Traffic	2k	30	2.130	3.666	1.246	0.719
vidyo1	720p	60	1.932	2.290	1.702	0.565
vidyo3	720p	60	2.254	2.931	1.245	0.809
vidyo4	720p	60	1.633	2.084	-0.502	2.343
OVERALL			1.600	2.094	1.357	0.873

6. REFERENCES

[1] G. J. Sullivan and R. L. Baker, "Efficient quadtree coding of images and video," *IEEE Transactions on Image Processing*, vol. 3, no. 3, pp. 327–331, 1994.

[2] R. Shukla, P. L. Dragotti, M. N. Do, and M. Vetterli, "Rate-distortion optimized tree-structured compressio algorightms for piecewise polynomial images," *IEEE Transactions on Image Processing*, vol. 14, no. 3, pp. 343–359, 2005.

[3] R. Mathew and D. S. Taubman, "Quad-tree motion modeling with leaf merging," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20, no. 10, pp. 1331–1345, 2010.

[4] P. Helle, S. Qudin, B. Bross, D. Marpe, M. O. Bici, K. Ugur, J. Jung, G. Clare, and T. Wiegand, "Block merging for quadtree-based partitioning in HEVC," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1720–1731, 2012.

[5] A. M. Tourapis, F. Wu, and S. Li, "Direct mode coding for bi-predictive slices in the H.264 standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, no. 1, pp. 119–126, 2005.

[6] G. Laroche, J. Jung, and B. Pesquet-Popescu, "RD optimized coding for motion vector predictor selection," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 9, pp. 1247–1257, 2008.

[7] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding HEVC standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, 2012.

[8] J. Bankoski, P. Wilkins, and Y. Xu, "VP8 data format and decoding guide," *http://www.ietf.org/internet-drafts/draft-bankoski-vp8-bitstream-01.txt*, 2011.

[9] D. Mukherjee, J. Han, J. Bankoski, R. Bultje, A. Grange, J. Koleszar, P. Wilkins, and Y. Xu, "A technical overview of vp9 - the latest open-source video codec," *SMPTE*, vol. 2013, no. 10, pp. 1–17, 2013.