https://doi.org/10.2352/ISSN.2470-1173.2021.10.IPAS-260

A Polar Coordinate Perspective on Motion Search in Video Coding

Jingning Han, Paul Wilkins, and Yaowu Xu; Google LLC; Mountain View, CA 94043

Abstract

Video coding heavily relies on motion compensated predictive coding to achieve its compression efficiency. As a central component, the block matching motion search exploits the trade-off between encoder complexity and compression efficiency. The diamond search and its variants are widely used in video encoders. They typically start with the largest search step size and scale down the step size by half at each stage thereafter. To cover an $N \times N$ pixel search region, the diamond search class requires a search complexity of $O(logN)$ check points. In this work, we view the motion search as a sampling problem around the initial reference point in polar coordinates. We modify the sampling pattern at each search stage to better approximate a circle. The search radius is scaled down by $\sqrt{2}$ between stages, in contrast to the factor of 2 used by the diamond search class, such that the sampling density is increased by a factor of 2 between stages. It retains a search complexity of $O(logN)$ over an $N \times N$ pixel region, with a linear increment as compared to the diamond search. It is experimentally shown that the proposed search pattern provides compression gains at the expense of a modest increase in encoder complexity.

Introduction

Video codecs use block based motion compensated prediction to remove temporal correlation. To find a pixel block in a given reference frame that best resembles a current block, one needs to search $N \times N$ motion vectors in full pixel resolution, followed by fine tuning at sub-pixel level. For a 1080p resolution video that is coded with a group of pictures of 16 frames in length, the motion search empirically needs to cover an area of 129×129 full pixels in order to achieve near optimal motion estimation results. Clearly a brute force search at $O(N^2)$ complexity is computationally prohibitive for encoder implementation.

A two-dimensional logarithmic search pattern was initially proposed in [1], which starts with the largest step size and checks 4 vertices in a diamond shape and the center point. It then selects the point which minimizes the sum of absolute difference (SAD) as the new center and repeats until the minima is the center point. The next stage reduces the step size by half and searches around the above center point. This process repeats until the step size reaches 1. A three-step search (TSS) approach [2] extends the pattern to cover 8 points around a square at each stage. The point that yields the lowest SAD value is used as the center point for next stage, with the search radius scaled down by half. Its name - three-step search - comes from the assumption that its maximum radius is 8 pixels, hence it only takes three stages to reach radius at 1 and finish the search. However, the concept can naturally extend to larger search ranges. The four-step search (FSS) [3] allows TSS to have multiple hops at the largest radius until the minima occurs at the center of the square, the rest process is as per the TSS. The new three-step search (NTSS) exploits the observation that the optimal motion vector more often sits close to the predicted motion vector [4]. It hence conducts an additional search over the nearest neighbors of the predicted motion vector in the first stage to evaluate the necessity of search at a large radius. All the above methods allow an $O(logN)$ search complexity.

To enhance the search precision, an unrestricted center-biased diamond search pattern is proposed in [5], where it conducts an 8-point diamond search at each stage similar to the FSS above, but re-centers to the predicted motion vector if prior search stages depart from the center. A hexagon-based search pattern that allows fewer search points than the diamond shape to cover a target search region is proposed in [6]. The center-biased search pattern further promotes the importance of selecting the initial search point (i.e. the center point). In [7], a predictive zonal approach to motion search has been proposed, which evaluates the spatial and temporal neighboring blocks' motion vectors to find the most probable candidate as the initial point for the diamond search.

The advances of single instruction multiple data (SIMD) instruction set over the last decade substantially reduce the CPU cycles needed to compute the block SAD [8]. To compute the SAD between two 16×16 8-bit pixel blocks, the latest AVX-512 instruction set only requires 4 packed additions and 3 regular scalar additions [9], plus several instructions to load the pixels into the vector registers. Such ashift makes the motion search's contribution to the overall complexity of a video encoder less critical. Even though a brute force search at $O(N^2)$ over a large region is still computationally prohibitive, relaxing the constraints on the number of search points in a diamond search with a linear increment is largely feasible. Recent developments propose to conduct a motion search, followed by a full rate-distortion cost evaluation, around each spatial and temporal reference motion vector for improved compression efficiency [10][11].

This work considers the possibility of modifying the motion search pattern to achieve better compression efficiency with a modest computational complexity increment. We view the motion search as a sampling problem around the initial point in a polar coordinate system. Each search stage corresponds to sampling at a given radius. We depart from the center-biased diamond search scheme and modify the distribution of sampling points to better approximate a circle. The diamond search scales down the radius by a factor of 2 between consecutive stages, which effectively changes the sampling density by a factor of 4. Instead this work adopts a scaling factor of $\sqrt{2}$ for the radius to make the sampling density change by a factor of 2 between consecutive stages. The proposed search pattern retains a complexity of $O(logN)$ with a linear increment from the diamond search.

The perspective from polar coordinate has been previously

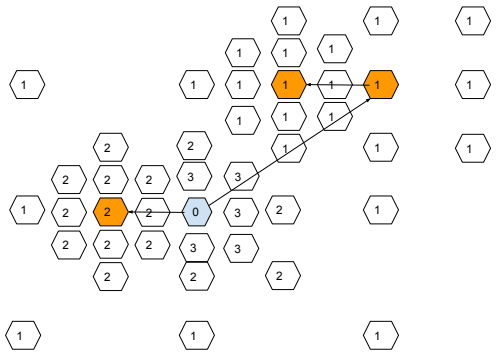


Figure 1. *An example of center-biased diamond search. The initial center is denoted by 0. The first diamond search path is denoted by points marked by 1. The arrows show the shift of search center between stages. A second diamond search is started when the first one steps away from the original center point and then similarly a third search, each time with a reduced radius.*

mentioned in [6] and [12] with a focus on reducing sampling points needed as compared to the conventional diamond search. This work proposes a new sampling pattern in both the radius and circle dimensions and demonstrates the overall compression efficiency gains at a modest encoder complexity increment.

Center-Biased Diamond Search

We briefly recap the center-biased diamond search scheme and extend it to cover a large search region for better compression performance. Though outdated, its performance remains the most competitive and the center-biased diamond search is widely used in modern video encoders. Therefore we use this modified center-biased diamond search scheme as our baseline.

The center-biased diamond search scheme is composed of multiple conventional diamond searches in light of [1]. It starts the search with the maximum radius around an initial center point. At the end of each stage it moves the center to the point that minimizes the weighted cost between the SAD (or sum of squared error) and the rate cost of the motion vector difference [13], where the weight is typically derived from the Lagrangian multiplier used by the rate-distortion optimization framework. The scheme then reduces the radius by half to conduct the next stage.

The center-biased method we are using here is depicted in Algorithm 1. If a diamond search finds a non-center point with minimum total cost at radius R, the scheme will kick off another diamond search at the original search center, at a reduced radius of $R/2$. Hence it guarantees a dense search around the initial center point. The effective search complexity depends on the video signal statistics. In the ideal case, where the SAD cost and the motion vector form a convex hull, and the initial center point provides a minimum SAD, the center biased diamond search subsumes to a conventional diamond search whose complexity is $O(logN)$. In a worst scenario, the search complexity can go up to $O((logN)^2)$.

To illustrate the process, we depict an example in Figure 1. Clearly the mechanism ensures dense search around the original center, regardless of the results at larger radius. It is also noteworthy that when the radius becomes small, i.e. less than 5 pixels, it is highly probable that different search routes would have sampling points that overlap.

Algorithm 1: Center-biased diamond search

Start with the maximum search radius $R = R_{max}$.
Identify a reference motion vector as the search center denoted by ref_mv.
Initialize $best\_sad\_cost$ to the cost associated with center ref_mv.
`Center biased diamond search()`
  while $R >= 1$ **do**
    $r, this\_mv, this\_sad\_cost =$ `Diamond_search(`R, ref_mv`)`
    $R = r/2$
    if $this\_sad\_cost < best\_sad\_cost$
    then
      $best\_mv = this\_mv$
      $best\_sad\_cost = this\_sad\_cost$
  end

Input: Maximum radius R, search center ref_mv
Output:
Maximum radius searched around the original center R
Best motion vector found
Associated weighted cost between the SAD and the motion vector rate cost

`Diamond_search(`R, ref_mv`)`
  while $R >= 1$ **do**
    Check 8 candidate motion vector offsets around the ref_mv:
    $\{-R,-R\}$, $\{-R,0\}$, $\{-R,R\}$, $\{R,-R\}$,
    $\{R,0\}$, $\{R,R\}$, $\{0,-R\}$, $\{0,R\}$.
    Find the one that minimizes the weighted cost.
    Update $ref\_mv = ref\_mv + offset$
    $R = R/2$.
  end

We implemented Algorithm 1 in the AV1 framework [14] as our baseline. Note that we used an 8-point square shape at a given radius, which experimentally provides slightly better compression performance than the 8-point diamond shape.

Polar Coordinate Sampling

We take an alternative perspective for the motion search process that uses polar coordinates. The reference motion vector sits on the origin. The search strategy breaks down to sampling in the radius and circle dimensions respectively.

Radius Scaling

Consider the radius scaling factor between stage $n-1$ to n, denoted by α. We have $R_{n-1} = \alpha R_n$, where R_n is the effective radius used by stage n. A lower stage index corresponds to larger radius, hence $\alpha > 1$. In conventional logarithmic search schemes [1]-[7], this is effectively set as $\alpha = 2$.

In the diamond search class, the sampling points around the circle are fixed at each stage. In the above center-biased diamond search, there are 8 points around the circle (square). To evaluate the search density, we consider the additional area covered from stage (n + 1) to stage n and from stage n to stage (n - 1). From

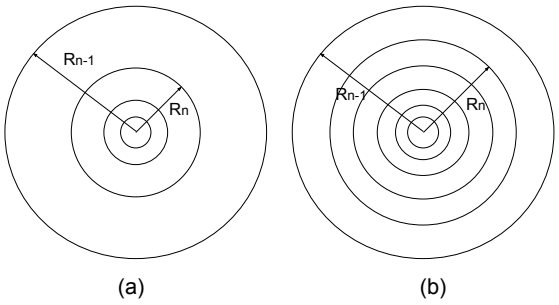


Figure 2. *Radius sampling. (a) Conventional $R_{n-1} = 2R_n$ sampling. The sampling density decreases by 4 times between stage n and stage n-1. (b) The proposed $R_{n-1} = \sqrt{2}R_n$. The sampling density decreases by half between stage n and stage n-1.*

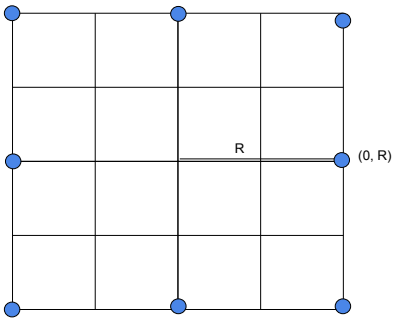


Figure 3. *Square search pattern.*

stage (n + 1) to stage n, the additional area covered is

$$\Delta S_n = \pi(\alpha^2 - 1)R_{n+1}^2. \quad (1)$$

From stage n to stage (n - 1), the additional area covered is

$$\Delta S_{n-1} = \pi(\alpha^2 - 1)R_n^2 = \pi(\alpha^2 - 1)\alpha^2 R_{n+1} = \alpha^2 \Delta S_n. \quad (2)$$

Clearly the sampling density changes between consecutive stages by α^2. A conventional approach setting $\alpha = 2$ effectively drops the search density by 4 times from stage n to (n - 1). We hence propose a simple modification, i.e. setting $\alpha = \sqrt{2}$, such that the search density goes down by 2 times between stages. To attain the same maximum radius, this change approximately doubles the number of search points. The sampling densities along the radius dimension that correspond to $\alpha = 2$ and $\alpha = \sqrt{2}$ are shown in Figure 2.

Sampling Around Circle

As discussed in the Section , the center-biased search scheme would likely provide dense sampling around the original center. To better tailor to the center-biased search scheme, we propose two search patterns to be used at different radii. When the radius is less than 6 pixels, we stay with the conventional square pattern as shown in Figure 3.

For radius above 6 pixels, we propose to increase the sampling points from 8 to 12 around each circle. The first 8 points are arranged to be uniformly distributed around the circle in an

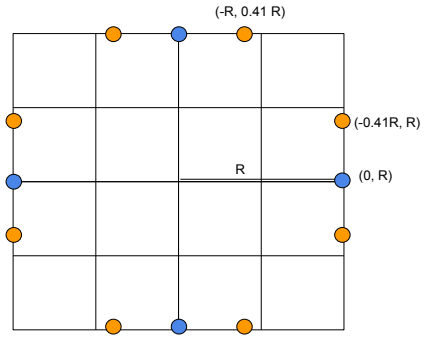


Figure 4. *Octagon-cross search pattern.*

octagon shape as shown by the orange points in Figure 4. The motion vector entropy coding system uses predictive coding for each component. A bit is first coded to indicate whether to directly re-use the predicted motion vector component, or to further code a difference. This approach tends to have a structural bias that assigns a higher probability to zero difference, which translates into a lower rate cost for zero difference. Hence offsets $\{0,R\},\{0,-R\},\{R,0\}$, and $\{-R,0\}$ as shown by the blue points in Figure 4 tend to have a higher probability of achieving the minimum weighted cost. Therefore we choose the search pattern such that 8 points uniformly sample the circle and an additional 4 points cover the offsets with zero difference in motion vector component. We refer to it as octagon-cross search pattern.

These two simple modifications in the radius and circle dimensions form our proposed center-biased octagon-cross search scheme as summarized in Algorithm 2.

Experimental Results

The proposed scheme Algorithm 2 was implemented in the libaom AV1 framework [14]. We evaluated the overall compression performance across different frame resolutions and target bitrates as compared to the center-biased diamond search in Algorithm 1. The baseline used libaom AV1 encoder at speed 1 (i.e. –cpu-used=1), which was the setting tuned for better compression efficiency with moderately high encoder complexity. The default 2-pass encoding mode was used. The rate control system used the default variable bit-rate mode.

All the video clips were in YUV420 format and were tested with 150 frames. A wide range of the operating bit-rates were evaluated that covered the average PSNR between 35 dB to 45 dB. The compression performance was evaluated in PSNR and SSIM. The BD-rate reduction [15] is shown in Table 1-2. A negative number means better compression performance. It is evident that the proposed center-biased octagon-cross search provides consistent compression performance gains, and is particularly advantageous to the video sequences that contained complex motion activities, e.g., *soccer*, *racehorse*, etc.

We next evaluated its impact on the encoder complexity due to the increased number of search points. We used a clip *BasketballDrill* at 480p to demonstrate typical observations at different operating bit-rates. We ran a single thread encoding and recorded the overall instruction counts. The actual encoding time change was closely aligned with the instruction count change. The results are shown in Table 3. The percentage of encoder com-

Algorithm 2: Center-biased octagon-cross motion search

Start with the maximum search radius $R = R_{max}$.
Identify a reference motion vector as the search center denoted by ref_mv.
Initialize $best\_sad\_cost$ to the weighted cost associated with center ref_mv.
`Center biased octagon-cross search()`
  while $R >= 1$ **do**
    $r, this\_mv, this\_sad\_cost =$ `Octagon_Cross_search(`R, ref_mv`)`
    $R = r/\sqrt{2}$
    if $this\_sad\_cost < best\_sad\_cost$ **then**
      $best\_mv = this\_mv$
      $best\_sad\_cost = this\_sad\_cost$
  end

Input: Maximum radius R, search center ref_mv
Output:
Maximum radius searched around the original center R
Best motion vector found
Associated weighted cost between the SAD and the rate cost of the motion vector

`Octagon_Cross_search(`R, ref_mv`)`
  while $R >= 1$ **do**
    if $R < 6$ **then**
      Check 8 candidate offsets in the square pattern
    else
      Check 12 candidate offsets in the octagon-cross pattern
    end
    Find the one that minimizes the SAD cost.
    Update $ref\_mv = ref\_mv + offset$
    $R = R/\sqrt{2}$.
  end

plexity increment is slightly lower in a high bit-rate setting, since the rate-distortion optimization process typically takes more instructions to handle non-zero transform coefficient blocks, which are more common in the high bit-rate cases, whereas the encoding complexity due to motion search remains largely the same at various operating bit-rates.

Conclusions

We consider the motion search process as a sampling problem in the polar coordinate. A center-biased octagon-cross search pattern is proposed based on design modifications in the radius and circle dimension respectively. The scheme is experimentally shown to improve the compression efficiency at moderate encoder complexity increase.

Compression performance gains for 480p resolution dataset in terms of BD-rate reduction.

		PSNR BD-rate	SSIM BD-rate
BQMall	832x480	-0.610	-0.659
PartyScene	832x480	-0.121	-0.218
old_town_cross	480p	-0.259	-0.268
snow_mnt	480p	-0.224	-0.041
soccer	4cif	-1.135	-1.345
speed_bag	480p	-0.061	0.205
station2	480	0.034	-0.089
tears_of_steel	480p	-0.370	-0.332
touchdown_pass	480p	-0.649	-0.641
crowd_run	480p	-0.056	-0.255
Flowervase	832x480	-0.100	-0.244
Mobisode2	832x480	-0.024	0.107
sintel_shot	854x480	-0.846	-1.238
BasketballText	832x480	-0.319	-0.278
BasketballDrill	832x480	-0.309	-0.323
RaceHorses	832x480	-1.801	-2.040
Keiba	832x480	-2.592	-3.488
harbour	4cif	-0.077	-0.156
west_wind_easy	480p	-0.059	-0.006
rush_field_cuts	480p	-0.006	-0.194
ducks_take_off	480p	0.071	0.005
park_joy	480p	-0.144	0.006
red_kayak	480p	0.027	-0.016
aspen	480p	-0.266	-0.313
city	4cif	-0.790	-0.893
controlled_burn	480p	-0.310	-0.319
crew	4cif	-0.372	-0.379
ice	4cif	-0.569	-0.149
into_tree	480p	-0.075	-0.149
OVERALL		-0.413	-0.467

References

[1] Jaswant Jain and Anil Jain, "Displacement measurement and its application in interframe image coding," *IEEE Transactions on Communications*, vol. 29, no. 12, pp. 1799–1808, 1981.

[2] Toshio Koga, "Motion compensated interframe coding for video-conferencing," *Proc. Nat. Telecommun. Conf*, 1981.

[3] Lai-Man Po and Wing-Chung Ma, "A novel four-step search algorithm for fast block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 3, pp. 313–317, 1996.

[4] Reoxiang Li, Bing Zeng, and Ming L. Liou, "A new three-step search algorithm for block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 4, no. 4, pp. 438–442, 1994.

[5] Jo Yew Tham, Surendra Ranganath, Maitreya Ranganath, and Ashraf A. Kassim, "A novel unrestricted center-biased diamond search algorithm for block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, no. 4, pp. 369–377, 1998.

[6] Ce Zhu, Xiao Lin, and Lap-Pui Chau, "Hexagon-based search pattern for fast block motion estimation," *IEEE*

Compression performance gains for high definition resolution dataset in terms of BD-rate reduction.

		PSNR BD-rate	SSIM BD-rate
crew	720p	-0.493	-0.586
ped	1080p	-0.476	-0.441
johnny	720p	0.279	0.388
kimono1	1080p	-0.516	-0.570
kristenandsara	720p	0.005	0.047
parkscene	1080p	-0.263	-0.264
rush_hour	1080p	-0.133	-0.039
station2	1080p	-0.564	-0.638
tennis	1080p	-1.075	-1.426
tractor	1080p	-1.331	-1.706
sheriff	720p	-0.000	-0.005
vidyo1	720p	0.001	0.125
vidyo3	720p	-0.013	0.055
vidyo4	720p	-0.210	-0.249
city	720p	-0.282	-0.098
night	720p	-0.246	-0.302
cyclists	720p	-1.274	-1.829
sunflower	720p	-1.525	-1.793
jets	720p	-0.190	-0.381
chinaspeed	xga	-1.310	-1.243
basketballdrive	1080p	-0.929	-1.282
blue_sky	1080p	-1.214	-2.128
bqterrace	1080p	-0.720	-1.012
cactus	1080p	-0.133	-0.090
riverbed	1080p	0.111	0.136
crowd_run	1080p	-0.259	-0.415
parkrun	720p	-0.014	-0.012
parkjoy	1080p	-0.256	-0.505
ducks_take_off	1080p	0.037	-0.047
in_to_tree	1080p	-0.102	-0.085
dinner	1080p	-0.791	-0.980
factory	1080p	-5.146	-5.699
life	1080p	-0.522	-0.544
mobcal	720p	-0.109	0.057
shields	720p	-0.104	-0.055
old_town_cross	720p	0.008	0.083
stockholm_ter	720p	-0.289	-0.285
fourpeople	720p	-0.099	-0.128
OVERALL		-0.530	-0.630

The encoder complexity increment in terms of the instruction counts in billion.

bit-rate (kbps)	baseline	octagon-cross	increment
800	1555	1691	8.7%
1600	2251	2416	7.3%
2400	2840	3042	7.1%

Transactions on Circuits and Systems for Video Technology, vol. 12, no. 5, pp. 349–355, 2002.

[7] Alexis M. Tourapis, Oscar C. Au, and Ming L. Liou, "Highly efficient predictive zonal algorithms for fast block-matching motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 10, pp. 934–947, 2002.

[8] John L. Hennessy and David A. Patterson, "Computer architecture: a quantitative approach," *Elsevier*, 2011.

[9] James Reinders, "Avx-512 instructions," *Intel*, 2013.

[10] 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.

[11] J. Han, Y. Xu, and J. Bankoski, "A dynamic motion vector referencing scheme for video coding," in *International Conference on Image Processing*. IEEE, 2016, pp. 2032–2036.

[12] Alexis Michael Tourapis, Oscar Chi Lim Au, and Ming Lei Liou, "Fast motion estimation using circular zonal search," *Visual Communications and Image Processing*, vol. 3653, 1998.

[13] Markus Flierl, Thomas Wiegand, and Bernd Girod, "Rate-constrained multi-hypothesis prediction for motion compensated video compression," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 11, pp. 957–969, 2002.

[14] "Alliance of open media - source code repository," *https://aomedia.googlesource.com/*.

[15] Gisle Bjontegaard, "Calculation of average psnr differences between rdcurves," *Technical Report VCEG-M33, ITU-T SG16/Q6, Austin, Texas, USA*, 2001.

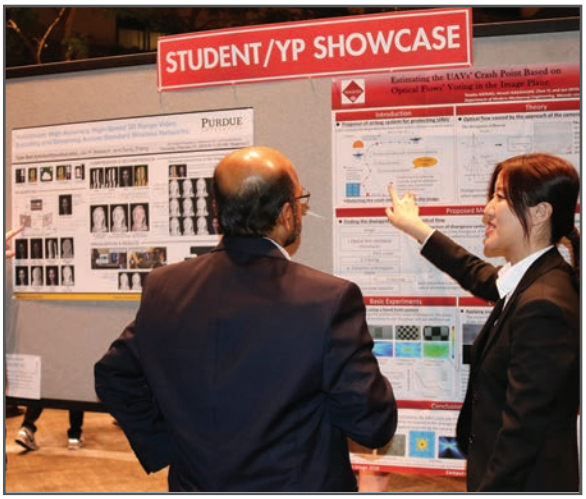
JOIN US AT THE NEXT EI!

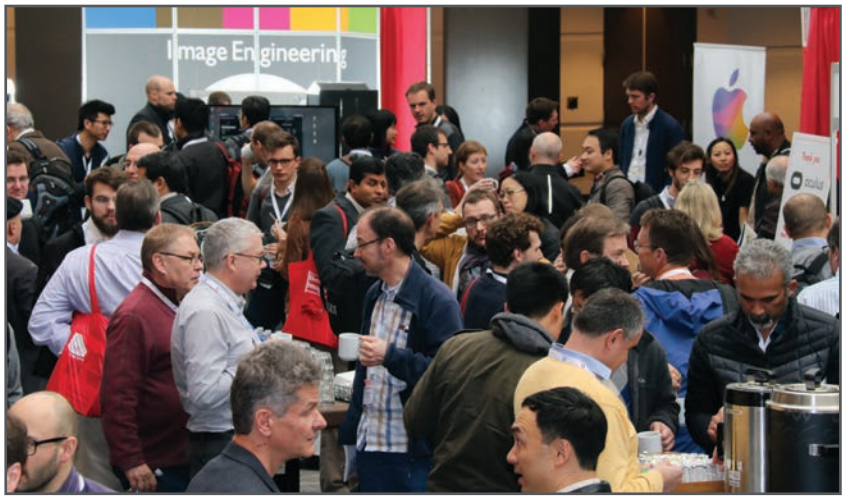
IS&T International Symposium on

Electronic Imaging

SCIENCE AND TECHNOLOGY

Imaging across applications . . . Where industry and academia meet!





• SHORT COURSES • EXHIBITS • DEMONSTRATION SESSION • PLENARY TALKS •
• INTERACTIVE PAPER SESSION • SPECIAL EVENTS • TECHNICAL SESSIONS •

www.electronicimaging.org

IS&T
imaging.org