Event Neural Networks
Supplementary Material

Matthew Dutson, Yin Li, and Mohit Gupta

University of Wisconsin–Madison, Madison WI 53715, USA
`{dutson,yin.li,mgupta37}@wisc.edu`

This is the supplement to our main paper. Here we present further results on low-level vision tasks (Sec. A), show additional analysis experiments (Sec. B), show preliminary results on HRNet [12], provide several example model outputs (Sec. D), expand on the details of our main experiments (Sec. E), provide a derivation for Eq. 4 (Sec. F), and discuss the theoretical properties of event networks (Sec. G). For sections, figures, tables, and equations, we use numbers (e.g., Fig. 1) to refer to the main paper and capital letters (e.g., Fig. A) to refer to this supplement.

A Results on Low-Level Tasks

In this section, we describe our experiments for low-level vision tasks. We consider HDRNet for image enhancement [5] and PWC-Net for optical flow [11].

Note that these models include some specialized operations (i.e., the bilateral transform for HDRNet [5] and flow warping in PWC-Net [11]). These operations represent a small portion of the overall computational cost of the models. For simplicity, we exclude them when counting multiply-accumulate operations.

Image Enhancement. HDRNet [5] can be trained to reproduce several image enhancement effects. We use the Local Laplacian [8] version of the model. HDRNet has two subnetworks: a deep, low-resolution feature network and a shallow, high-resolution guidemap network. The guidemap network represents only about 10 % of the overall operations, and converting it to an EvNet has a noticeable effect on the visual quality of the output. Therefore, we only convert the feature network to an EvNet. We report operation savings for both the overall model (both subnetworks) and the feature network (the EvNet portion). We refer to these operation counts as "HDRNet-a" and "HDRNet-f," respectively. We use a threshold of $h = 0.1$ and evaluate using the PSNR metric. We resize all images to 540×960 before applying the model.

We use the original authors' pretrained weights. However, these weights were trained on a non-public dataset. Therefore, instead of evaluating the model against ground truth labels, we compute the agreement between the outputs of the event model and conventional model. We evaluate on a subset of the MPII video dataset [1] (see Sec. E for details on the dataset). See Table A and Table B for results. We also show example model outputs Fig. E.

Optical Flow. We also consider the PWC-Net model [11] for optical flow computation. Unlike the other models (OpenPose, YOLO, HDRNet) which take a

Table A. Results on Low-Level Tasks. Results for image enhancement and optical flow. The tables gives the overall computation savings, agreement between the conventional and event models, and overhead percentages (number of extra operations expended for each operation saved).

Model	Savings	Agreement	Math	Load/Store
HDRNet-a	5.78x	39.4 PSNR	2.57 %	3.79 %
HDRNet-f	23.9x	39.4 PSNR	2.57 %	3.79 %
PWC-Net	2.68x	0.335 EPE	0.44 %	0.74 %

Table B. Camera Motion for Low-Level Tasks. The savings factor for different levels of camera motion, evaluated on our custom MPII dataset (see Sec. E).

Model	None	Minor	Major
HDRNet-a	6.19×	6.02×	5.55×
HDRNet-f	34.9×	29.7×	20.0×
PWC-Net	5.41×	3.29×	2.11×

single frame as input, this model takes a *pair* of frames. We use a threshold of $h = 0.01$ and evaluate using the EPE metric [2]. We resize all images to 288×512 before applying the model. We use the original authors' weights trained on Sintel [3]. Like with HDRNet, we evaluate the agreement between the event and conventional outputs. Results are shown in Table A and Table B. We also evaluate on the ground-truth labels in the Sintel training dataset. On this data, the conventional model achieves EPE 2.86 and the event model achieves EPE 3.33.

B Additional Analysis Experiments

Layer Trends. Fig. A shows the computational cost of the OpenPose model as a function of the layer depth. We show results both on the JHMDB dataset and on our custom-labelled MPII dataset (to allow analysis of the effect of camera motion). Overall, we see a reduction in the relative cost as we go deeper in the network. This highlights the importance of leveraging repetition in the deep layers of the network, not just near the input. We also observe that the early layers transmit more frequently when there is large camera motion. This corresponds to an increased number of changes in low-level features and pixel values.

Temporal Variation. Fig. B shows the per-frame computational cost of the OpenPose EvNet over the course of a video. The video in question has a static background and a moving foreground object (person). Recognizable events in the video (e.g., walking, jumping) correspond to temporary increases in the number of operations. In this way, we see EvNets living up to their promise of "only computing when something interesting is happening."

Varying Granularity. Table C shows the effect of increasing the granularity of the policy. We evaluate the OpenPose model [4] on the JHMDB dataset [7]. We

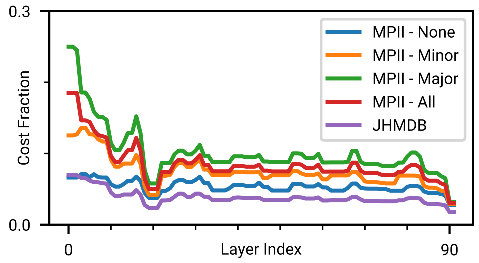

Fig. A. Operation Costs by Layer. Results for the OpenPose model on the JHMDB and custom-labelled MPII datasets. The increasing savings with depth show the importance of leveraging repetition at all levels of the network hierarchy. We have applied a median filter of size 5 (along the layer axis) to the data in this plot.

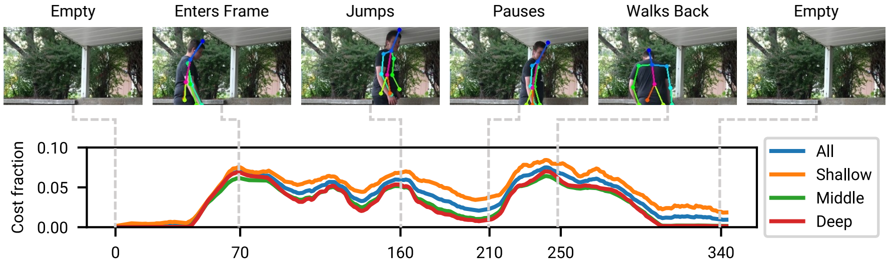

Fig. B. Temporal Variation in Operation Cost. Identifiable events in the video (e.g., jumping) correspond to temporary increases in the number of operations. “Shallow” corresponds to the first 31 layers, “middle” to the next 31, and “deep” to the final 30. We have applied a centered moving average of size 10 (along the time axis) to the data in this plot.

Table C. Varying Granularity. Results for the OpenPose model on the JHMDB dataset. Using larger chunks, especially channel chunks, reduces the computational gains somewhat. However, chunking may have practical benefits on some hardware.

Variant	Threshold	PCK	Operations
Conventional	–	0.7640	7.055×10^{10}
No chunking	0.05	0.7581	6.166×10^{9}
2x2 chunks	$0.05/\sqrt{2}$	0.7575	8.574×10^{9}
4x4 chunks	$0.05/\sqrt{4}$	0.7662	1.191×10^{10}
8x8 chunks	$0.05/\sqrt{8}$	0.7431	1.646×10^{10}
Channel chunks	0.02	0.7600	1.782×10^{10}

test both a spatial chunking policy and a policy that chunks along the channel dimension (e.g., [6]). Because each neighborhood computes a mean of several $|d|$, the thresholds must be reduced to keep the accuracy from dropping. The threshold-setting strategy $0.05/\sqrt{n}$ is a heuristic that we found to give relatively stable accuracy with varying n. The results show that increasing the chunk size reduces the operation savings. However, chunking may, in practice, allow more efficient execution on certain hardware.

Comparison Against Output Interpolation. One alternate strategy for efficient video inference is to run a model once every n frames and interpolate its predictions for the remaining $n-1$ frames. We apply this strategy to OpenPose on JHMDB and compare it to the EvNet approach. We use $n = 16$ and linearly interpolate the joint positions between model predictions (the value 16 was chosen to give a computational cost close to the EvNet in Table 1). The interpolated model expends 6.764×10^9 ops per frame on average and achieves a PCK of 68.52 % (a reduction of 7.88 % from the conventional model). Compare this to the EvNet in Table 1, which expends 6.780×10^9 ops on average while achieving a PCK score of 76.37 % (a reduction of 0.03 % from the conventional model). Compared to output interpolation, the EvNet gives much higher accuracy at a similar computation cost. Note that we trim the inputs for the interpolation model to have a length of $kn+1$ frames, where k is a positive integer. This ensures that the video can be divided into uniform blocks of n frames (with one extra frame at the end). If we trim to the same length for the EvNet, it achieves a PCK of 76.82 % with average cost 7.265×10^9 ops. The conventional model achieves PCK 76.67 % at 7.055×10^{10} ops on the trimmed video.

Temporal Smoothness. We have anecdotally observed improved temporal smoothness in the outputs of EvNets. We hypothesize that this is one of the reasons for the slightly increased accuracy for some models (e.g., Table 1) over the conventional baselines. We quantitatively measure temporal smoothness for OpenPose on JHMDB by measuring the mean L2 joint motion between frames. The average joint motion for the conventional model is 10.3 pixels. For the EvNet with threshold $h = \{0.01, 0.02, 0.04, 0.06, 0.08\}$, the average motion was $\{9.77, 9.26, 7.98, 7.14, 5.81\}$ pixels. This confirms that the EvNet outputs are

Table D. HRNet Results. Results for the HRNet model on JHMDB.

Model	Threshold	PCK	Operations
Conventional	–	90.37 %	1.019×10^{10}
EvNet	0.05	90.43 %	1.112×10^{9}
EvNet	0.1	90.46 %	7.361×10^{8}
EvNet	0.2	86.44 %	4.187×10^{8}
Skip-Conv	0.05	89.17 %	1.035×10^{9}
Skip-Conv	0.1	84.45 %	6.473×10^{8}
Skip-Conv	0.2	78.72 %	3.307×10^{8}

more temporally smooth than those of the conventional model, with smoothness increasing with the policy threshold.

C HRNet Experiments

We test HRNet [12], a state-of-the-art model for various location-based tasks (e.g., object detection) on the JHMDB pose recognition dataset [7]. We use the HRNet-W32 version of the model.

Training Procedure. We initialize with pretrained MPII weights from [12]. We fine-tune the model on JHMDB for 20 epochs using the Adam optimizer and a learning rate of 1×10^{-5}. We set aside 20 % of the training data for validation and save the model at the epoch with the lowest validation loss. JHMDB defines three train/test splits – we train and evaluate a model on each training split and average the results (accuracy and computation costs) over the three splits. Where not otherwise specified, we adopt the training and data augmentation parameters of [6]. All of our training code will be publicly released and included with the supplementary material.

Evaluation. We evaluate three model variants: the conventional model, an EvNet and Skip-Conv (without periodic resets). See Table D for results. We report the PCK metric (as in our experiments on OpenPose). The accuracy and savings we observe are in line with our other experiments.

D Example Outputs

Example Outputs. Fig. C, Fig. D, Fig. E, and Fig. F show several example outputs for OpenPose, YOLO, HDRNet, and PWC-Net, respectively. The videos shown are from our custom MPII dataset (and hence all have 41 frames). In general, we see strong agreement between the conventional and event predictions. In some cases (especially with the YOLO model; Fig. D), we observe greater consistency in the EvNet predictions across frames. This is a consequence of an event network's preference for re-using previous activation values. This greater temporal consistency does not appear to reduce the model's ability to keep up with rapid changes.

E Experiment Details

Custom MPII Dataset. Here we describe the dataset that we use in our camera motion experiments (Table 3 and Table B) and for evaluating HDRNet and PWC-Net (Table A). We take a subset of the MPII video dataset [1] – specifically, the first 246 videos that have exactly 41 frames (most, but not all videos in MPII have 41 frames). We then label each video in this dataset as having "no camera motion" (perfectly stationary camera), "minor camera motion" (slight camera shake), or "major camera motion". These splits contain 59, 46, and 141 videos, respectively.

Overhead Counting. We count overhead operations as follows. An update to an accumulator requires one load (a), one addition ($a + g(\boldsymbol{\Delta}_{\text{in}})$), and one store ($a$). An update to a gate requires two loads (b and d), three additions ($d+f(a)-b$ and $|d| - h$), and two stores (b and d). A transmission requires one load (d), one subtraction ($d - \Delta_{\text{out}}$), and one store (d).

Tables of Results. Table E shows the complete results for OpenPose on JHMDB. These values correspond to the points in Fig. 7 (a). Table E shows the complete results for YOLO [9] on VID [10], corresponding to Fig. 7 (b). Table G shows the overhead operation percentages for all thresholds tested for Fig. 7.

We also show results for larger input images (352 × 480 for OpenPose and 320×544 for YOLO). Results for pose recognition, object detection, and overhead are given in Table H, Table I, and Table J, respectively.

F Derivation of Equation 4

The equation for $a^{(T)}$ is a consequence of the update to $a^{(t)}$ defined in Eq. 3, combined with the linearity of g (g of a sum is equal to the sum of the g). The equation for $b^{(T)}$ is a direct consequence of the update rule in Eq. 3.

The equation for $d^{(T)}$ in Eq. 4 comes from combining Eq. 3 and the post-transmission subtraction of Δ_{out}. Let $d^{(0)} = 0$ as stated in Sec. 4.2. With Eq. 3, and noting that $b^{(t)} = f(a^{(t)})$,

$$d^{(T)} = \textstyle\sum_{t=1}^{T} \left(f(a^{(t)}) - b^{(t-1)}\right) = b^{(T)} - b^{(0)}. \tag{A}$$

Adding in the post-transmission subtraction of $\Delta_{\text{out}}^{(t)}$, we have

$$d^{(T)} = b^{(T)} - b^{(0)} - \textstyle\sum_{t=1}^{T} \Delta_{\text{out}}^{(t)}. \tag{B}$$

G Thoughts on Theoretical Guarantees

For certain special cases of transmission policies (e.g., a threshold policy with $h = 0$), we can guarantee that the output of an EvNet will be equal to that of the equivalent conventional network. As we make the policy more selective (e.g., by increasing h), the efficiency of the EvNet improves, but its output increasingly deviates from that of the conventional network. While we currently describe this behavior qualitatively, developing the rigorous theoretical tools necessary for a quantitative description is an important next step.

We can describe a neural network as a composition of functions,

$$\boldsymbol{y} = q_m(\ldots(q_2(q_1(\boldsymbol{x})))\ldots). \tag{C}$$

We can think of an event network as perturbing the output of each q_i by some $\boldsymbol{\epsilon}_i$. That is,

$$\boldsymbol{y}' = q_m(\ldots(q_2(q_1(x) + \boldsymbol{\epsilon}_1) + \boldsymbol{\epsilon}_2)\ldots) + \boldsymbol{\epsilon}_m. \tag{D}$$

If we assume a threshold policy with threshold h, then $\|\boldsymbol{\epsilon}_i\|_\infty < h$. Given these facts and some knowledge of the properties of the q_i (e.g., the distribution of their weights), can we bound the norm of $\boldsymbol{y} - \boldsymbol{y}'$? This question has important implications for applications that require accuracy guarantees and should be studied in future work.

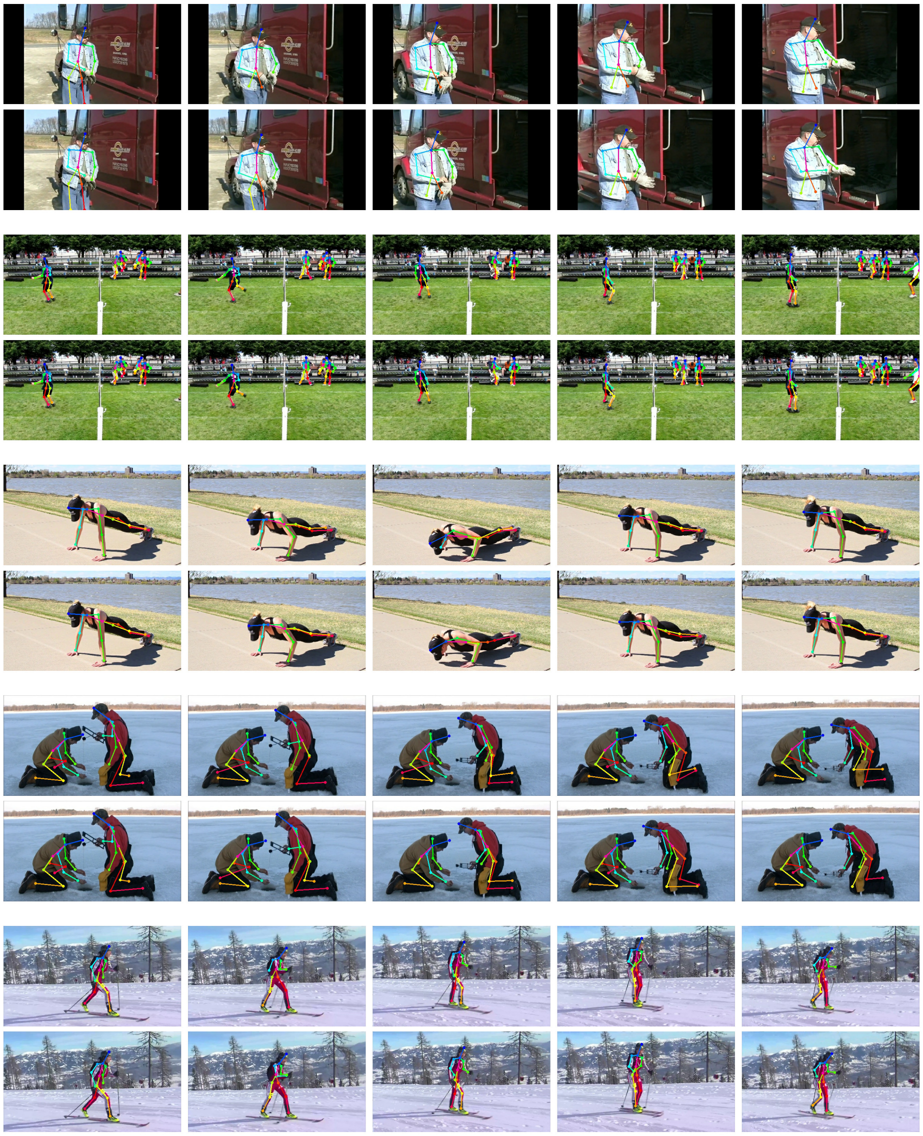

Fig. C. OpenPose Result Samples. Conventional output is on the top and event output is on the bottom. We show frames 0, 10, 20, 30, and 40 from each video. Videos are from the MPII dataset.

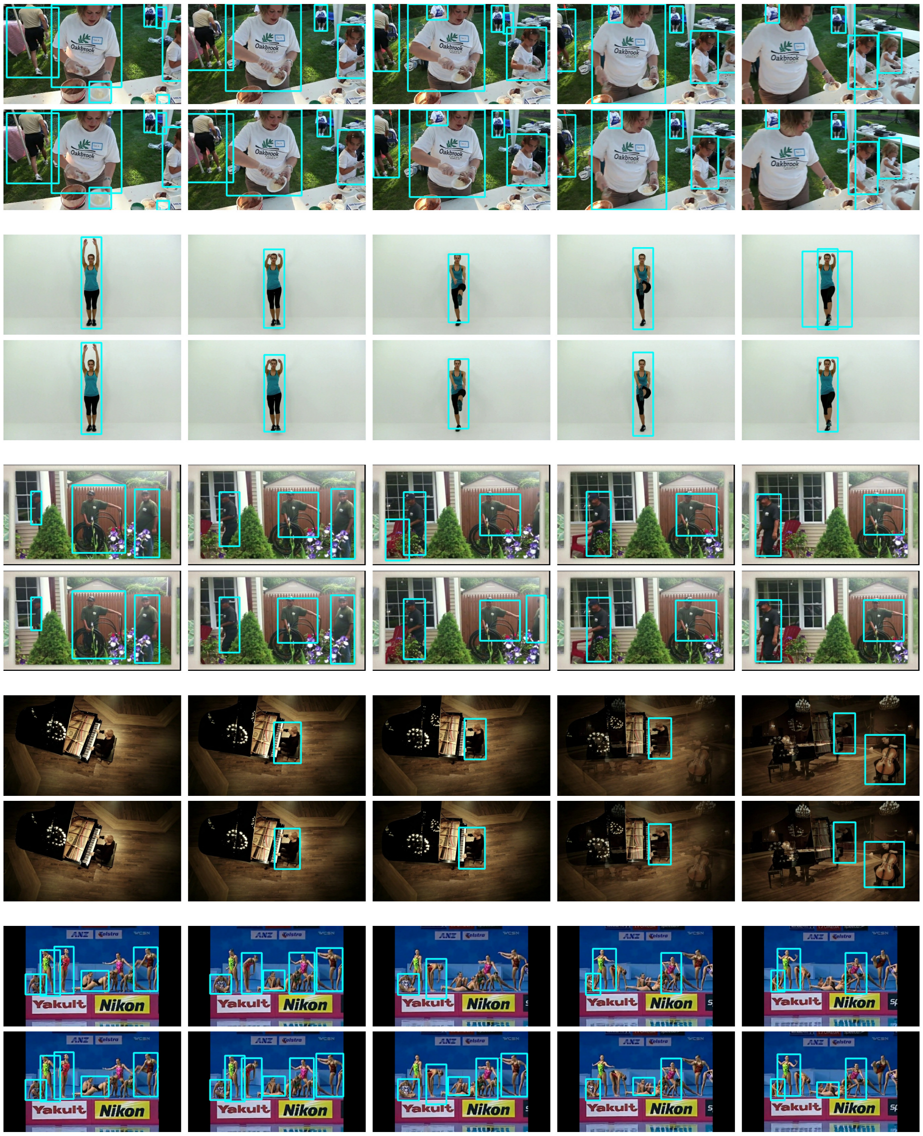

Fig. D. YOLO Result Samples. Conventional output is on the top and event output is on the bottom. We show frames 0, 10, 20, 30, and 40 from each video. Videos are from the MPII dataset.

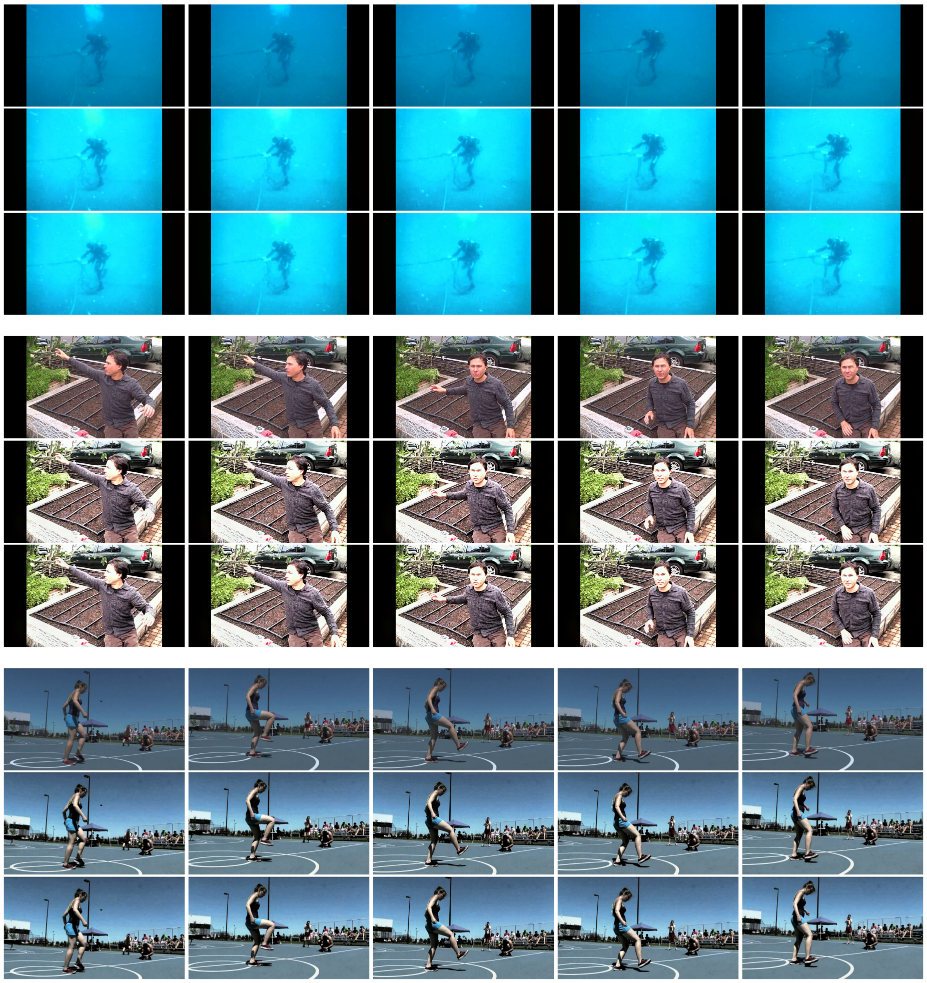

Fig. E. HDRNet Result Samples. The model input is on the top, conventional output is in the middle, and event output is on the bottom. We show frames 0, 10, 20, 30, and 40 from each video. Videos are from the MPII dataset.

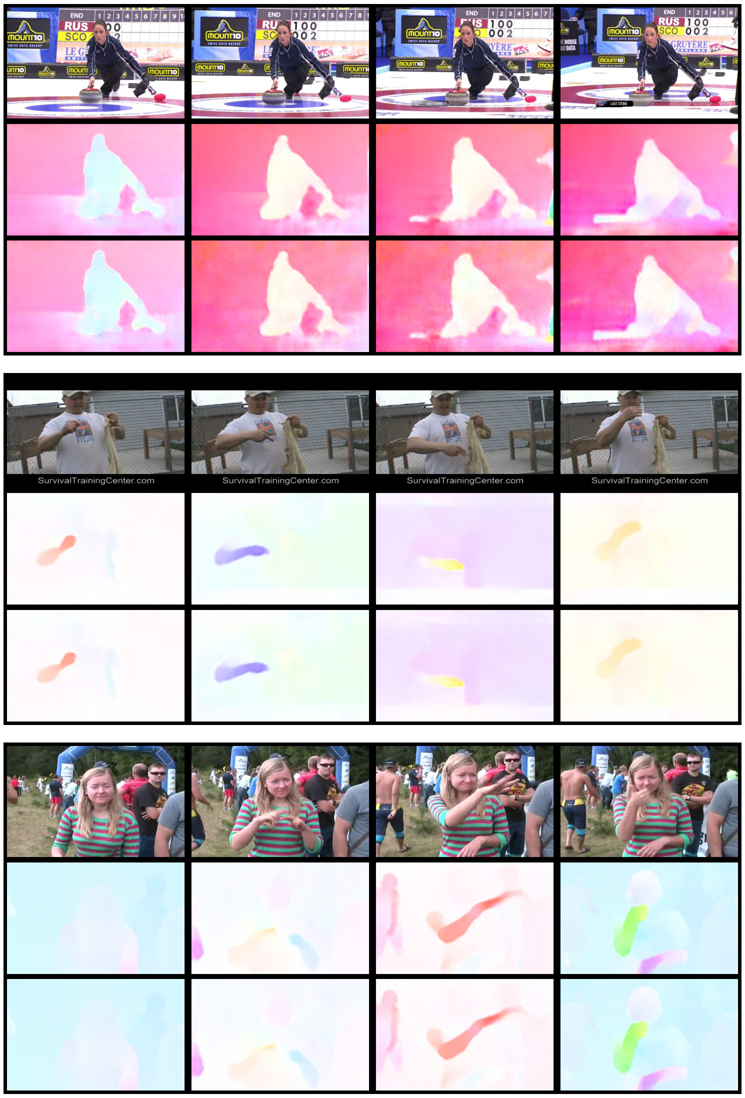

Fig. F. PWC-Net Result Samples. The model input is on the top, conventional output is in the middle, and event output is on the bottom. We show frames 0, 10, 20, and 30 from each video. Videos in this dataset have 41 frames, and we predict flow for each pair of frames. There, we cannot show output for frame 40. Videos are from the MPII dataset.

Table E. Video Pose Estimation. Detailed results for the OpenPose model on JHMDB. The "Skip-Conv Reset" model re-flushes the network (i.e., sets all thresholds to zero) every 8 frames. See Fig. 7.

Model	Threshold	PCK	Operations
Conventional	–	0.7640	7.055×10^{10}
EvNet	0.01	0.7656	1.071×10^{10}
EvNet	0.02	0.7696	8.825×10^{9}
EvNet	0.04	0.7637	6.780×10^{9}
EvNet	0.06	0.7448	5.640×10^{9}
EvNet	0.08	0.7310	4.890×10^{9}
Skip-Conv	0.01	0.7603	1.027×10^{10}
Skip-Conv	0.02	0.7277	7.873×10^{9}
Skip-Conv	0.04	0.6644	5.837×10^{9}
Skip-Conv Reset	0.01	0.7621	1.092×10^{10}
Skip-Conv Reset	0.02	0.7311	8.816×10^{9}
Skip-Conv Reset	0.04	0.6635	7.054×10^{9}

Table F. Video Object Detection. Detailed results for the YOLO model on VID. The "Skip-Conv Reset" model re-flushes the network (i.e., sets all thresholds to zero) every 8 frames. See Fig. 7.

Model	Threshold	mAP50	Operations
Conventional	–	0.5538	1.537×10^{10}
EvNet	0.01	0.5545	7.472×10^{9}
EvNet	0.02	0.5563	6.074×10^{9}
EvNet	0.04	0.5618	4.517×10^{9}
EvNet	0.08	0.5619	3.061×10^{9}
EvNet	0.12	0.5463	2.306×10^{9}
EvNet	0.16	0.5024	1.812×10^{9}
Skip-Conv	0.01	0.5413	7.340×10^{9}
Skip-Conv	0.02	0.4581	5.705×10^{9}
Skip-Conv	0.04	0.3098	3.819×10^{9}
Skip-Conv Reset	0.01	0.5406	8.111×10^{9}
Skip-Conv Reset	0.02	0.4544	6.692×10^{9}
Skip-Conv Reset	0.04	0.2737	5.054×10^{9}

Table G. Operation Overhead. The amount of overhead operations required for computing EvNet updates. "Math" refers to arithmetic operations (additions and subtractions) and "load/store" refers to memory access operations. Percentages are the ratio of additional operations expended for each arithmetic operation saved. For example, a memory overhead of 1 % indicates that one extra load/store is expended for each 100 arithmetic operations saved. See Table 2.

Model	Threshold	Load/Store	Memory
OpenPose	0.01	0.16 %	0.28 %
OpenPose	0.02	0.14 %	0.25 %
OpenPose	0.04	0.12 %	0.21 %
OpenPose	0.06	0.10 %	0.18 %
OpenPose	0.08	0.09 %	0.16 %
YOLO	0.01	0.88 %	1.57 %
YOLO	0.02	0.74 %	1.28 %
YOLO	0.04	0.62 %	1.04 %
YOLO	0.08	0.52 %	0.85 %
YOLO	0.12	0.47 %	0.74 %
YOLO	0.16	0.43 %	0.67 %

Table H. Video Pose Estimation for Larger Images. The corrolary of Table E, but for larger input images (352 × 480 instead of 320 × 240).

Model	Threshold	PCK	Operations
Conventional	–	0.8181	1.591×10^{11}
EvNet	0.01	0.8171	2.324×10^{10}
EvNet	0.02	0.8200	1.903×10^{10}
EvNet	0.04	0.8073	1.446×10^{10}
EvNet	0.06	0.7785	1.196×10^{10}
EvNet	0.08	0.7489	1.033×10^{10}
Skip-Conv	0.01	0.8065	2.183×10^{10}
Skip-Conv	0.02	0.7656	1.660×10^{10}
Skip-Conv	0.04	0.6852	1.212×10^{10}
Skip-Conv Reset	0.01	0.8084	2.344×10^{10}
Skip-Conv Reset	0.02	0.7689	1.886×10^{10}
Skip-Conv Reset	0.04	0.6940	1.497×10^{10}

Table I. Video Object Detection for Larger Images. The corrolary of Table F, but for larger input images (320 × 544 instead of 224 × 384).

Model	Threshold	mAP50	Operations
Conventional	–	0.5655	3.164×10^{10}
EvNet	0.01	0.5658	1.527×10^{10}
EvNet	0.02	0.5679	1.246×10^{10}
EvNet	0.04	0.5726	9.289×10^{9}
EvNet	0.08	0.5785	6.284×10^{9}
EvNet	0.12	0.5616	4.714×10^{9}
EvNet	0.16	0.5007	3.686×10^{9}
Skip-Conv	0.01	0.5525	1.498×10^{10}
Skip-Conv	0.02	0.4716	1.164×10^{10}
Skip-Conv	0.04	0.3140	7.726×10^{9}
Skip-Conv Reset	0.01	0.5552	1.656×10^{10}
Skip-Conv Reset	0.02	0.4624	1.366×10^{10}
Skip-Conv Reset	0.04	0.2829	1.026×10^{10}

Table J. Operation Overhead for Larger Images. The corrolary of Table G, but for larger input images (352 × 480 for OpenPose and 320 × 544 for YOLO).

Model	Threshold	Load/Store	Memory
OpenPose	0.01	0.15 %	0.26 %
OpenPose	0.02	0.13 %	0.23 %
OpenPose	0.04	0.11 %	0.19 %
OpenPose	0.06	0.09 %	0.16 %
OpenPose	0.08	0.08 %	0.14 %
YOLO	0.01	0.85 %	1.51 %
YOLO	0.02	0.71 %	1.23 %
YOLO	0.04	0.60 %	1.00 %
YOLO	0.08	0.50 %	0.81 %
YOLO	0.12	0.44 %	0.71 %
YOLO	0.16	0.40 %	0.64 %

References

1. Andriluka, M., Pishchulin, L., Gehler, P., Schiele, B.: 2D human pose estimation: New benchmark and state of the art analysis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3686–3693 (2014)
2. Baker, S., Scharstein, D., Lewis, J.P., Roth, S., Black, M.J., Szeliski, R.: A database and evaluation methodology for optical flow. International Journal of Computer Vision **92**(1), 1–31 (Mar 2011). https://doi.org/10.1007/s11263-010-0390-2
3. Butler, D.J., Wulff, J., Stanley, G.B., Black, M.J.: A naturalistic open source movie for optical flow evaluation. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) Computer Vision – ECCV 2012. pp. 611–625. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33783-3_44
4. Cao, Z., Simon, T., Wei, S.E., Sheikh, Y.: Realtime multi-person 2D pose estimation using part affinity fields. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7291–7299 (2017)
5. Gharbi, M., Chen, J., Barron, J.T., Hasinoff, S.W., Durand, F.: Deep bilateral learning for real-time image enhancement. ACM Transactions on Graphics **36**(4), 118:1–118:12 (Jul 2017). https://doi.org/10.1145/3072959.3073592
6. Habibian, A., Abati, D., Cohen, T.S., Bejnordi, B.E.: Skip-convolutions for efficient video processing. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2695–2704 (2021)
7. Jhuang, H., Gall, J., Zuffi, S., Schmid, C., Black, M.J.: Towards understanding action recognition. In: International Conference on Computer Vision (ICCV). pp. 3192–3199 (Dec 2013)
8. Paris, S., Hasinoff, S.W., Kautz, J.: Local Laplacian Filters: Edge-aware image processing with a Laplacian pyramid. ACM Transactions on Graphics p. 11 (2011)
9. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Conference on Computer Vision and Pattern Recognition (CVPR). pp. 779–788 (2016)
10. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet large scale visual recognition challenge. International Journal of Computer Vision (IJCV) **115**(3), 211–252 (2015). https://doi.org/10.1007/s11263-015-0816-y
11. Sun, D., Yang, X., Liu, M.Y., Kautz, J.: PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 8934–8943 (2018)
12. Sun, K., Xiao, B., Liu, D., Wang, J.: Deep high-resolution representation learning for human pose estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5693–5703 (2019)