EventNet

Asynchronous recursive event processing

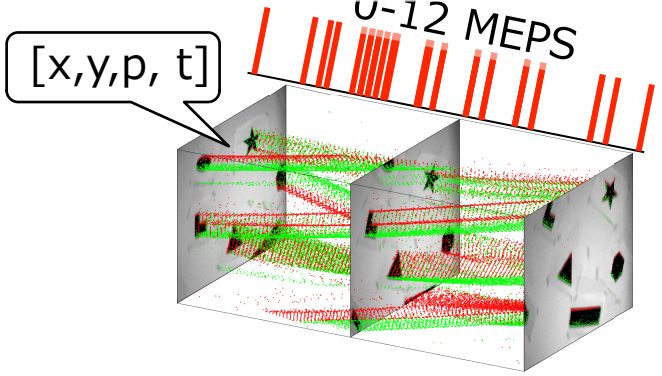
https://arxiv.org/abs/1812.07045
CVPR2019, Poster **#160 1-2P**

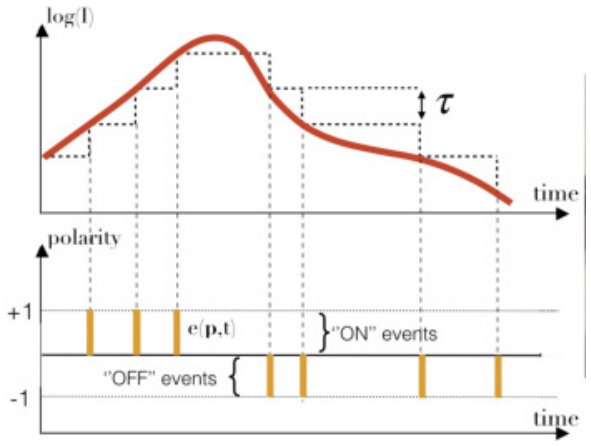
Yusuke Sekikawa
Denso IT Laboratory Inc., Japan

Motivation: Event-based Processing

Event-based camera

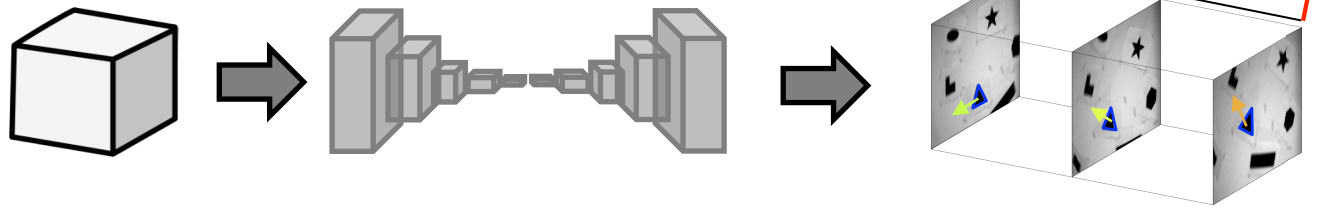
- HDR
- Fast (1μs, 0-12MEPS※)
- Non-redundant data representation





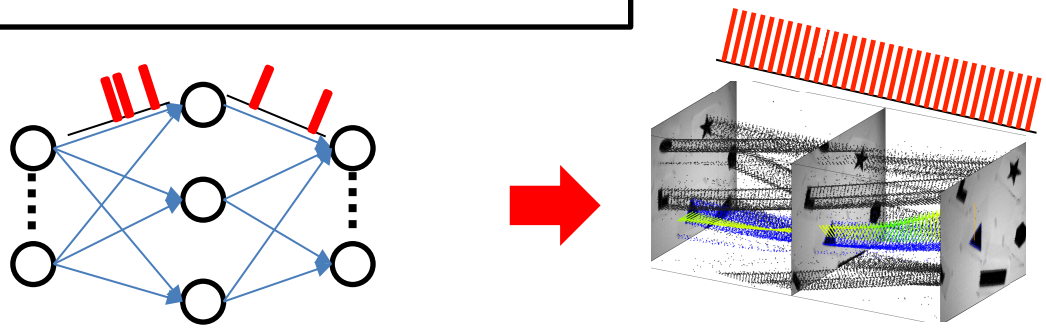
Frame-based

Frame/Event Frame †



† E.x., ”OneEvent-based Vision meets Deep Learning on Steering Prediction for Self-driving Cars”, CVPR2018

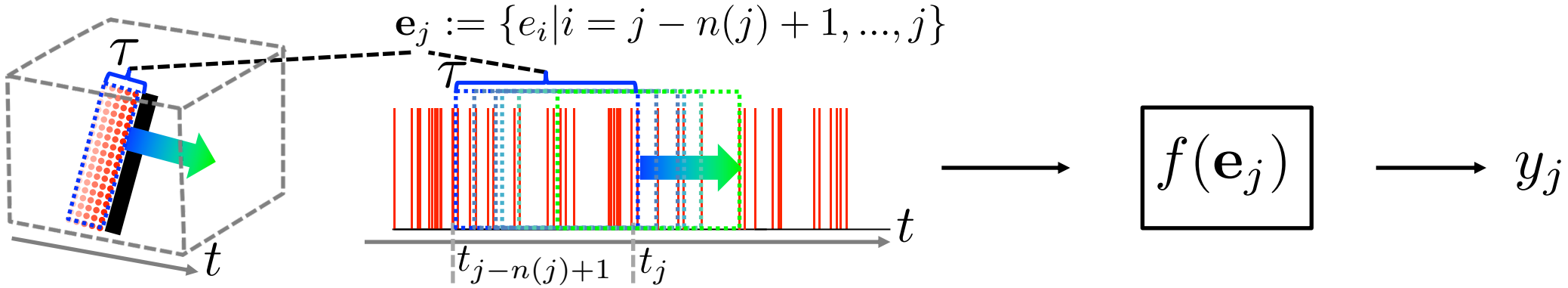
Event-based



Problem Statement

Event-based Camera

- Asynchronous sparse event data(position/polarity/time) $e : (x, y, p, \Delta t)$



Requirements

- E2E Trainable (Supervised learning)
- Sparse Event-wise Processing (No densification)
- Recursive Processing (Real time processing)
- Local Permutation Invariance (Noise/temporal resolution)

Related work: Deep Learning on Unordered Sets

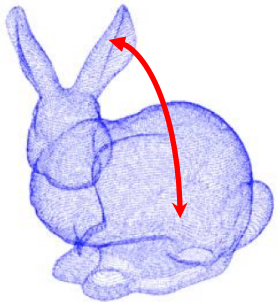
PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation

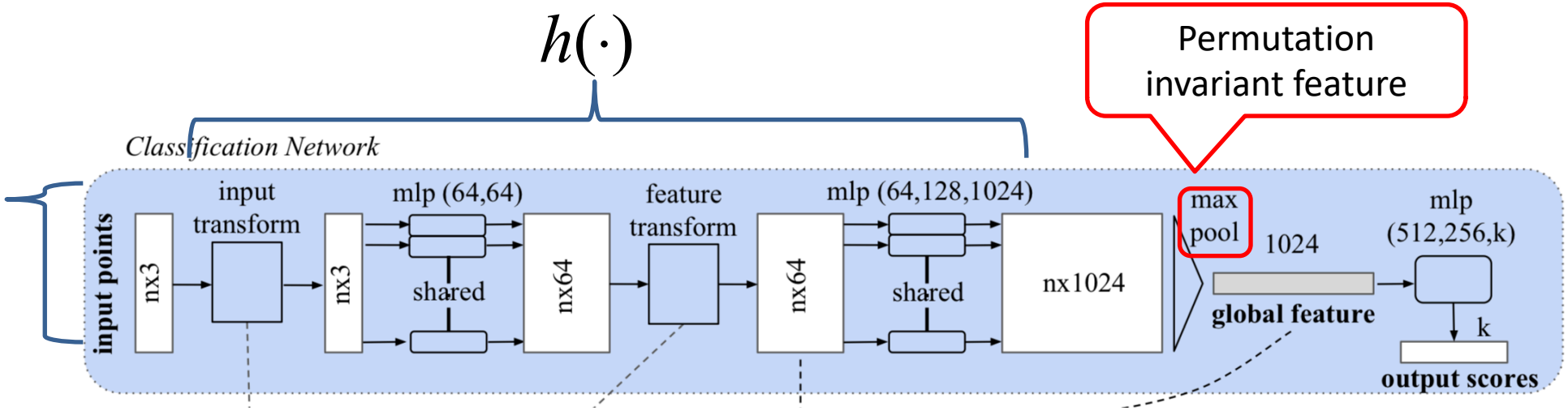
Charles R. Qi* Hao Su* Kaichun Mo Leonidas J. Guibas
Stanford University

$$y = f(\{x_1,\ldots,x_n\}) \approx g(\max(h(x_1),\ldots,h(x_n)))$$

Input Point Cloud

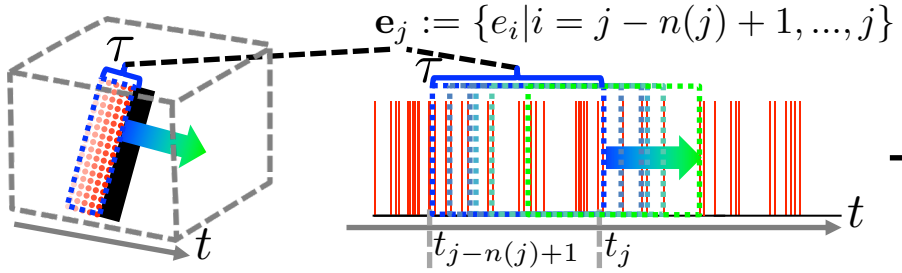
$\{x_1,\ldots,x_n\}$





Approximate point set function using symmetric function

Toward Real-time Processing



$\longrightarrow$ $f(\mathbf{e}_j)$ $\longrightarrow$ y_j

Changes every time new event arrives

PointNet (batch)

MLP

$$y_j = f(\mathbf{e}_j) \approx g(\max(h(e_{j-n(j)+1}), ..., h(e_j)))$$

$$e : (x, y, p, \Delta t)$$

$\boldsymbol{n(j)}$ events (τ ms)

Problem

- Needs to reprocess all $n(j)$ event withs MLP $h(\cdot)$※
- Needs to compute $\max$ for all $n(j)$ high dimensional vector $h(e)$

Infeasible to process millions of events in real-time

EventNet: Key Components for Real-time

Recursive Processing (EventNet)

① Recursive Event-wise Processing

Recursive computation, while keeping temporal information

$$h(e_j) \rightarrow c(h(e^-_j), \Delta t_j) \quad e^- : (x, y, p)$$

$$f(\mathbf{e}_j) = g(\max(c(s_{j-n(j)+1}, \delta t_{j-1}), \ldots, c(h(e^-_j), 0)))$$

$$= g(\max(c(s_{j-1}, \delta t_{j-1}), h(e^-_j)))$$

1 event

② LUT Realization of MLP h

$$h(x, y, p) = \mathrm{LUT}(x, y, p)$$

③ Asynchronously Two Module Architecture

Event-driven(h)/On-demand(g)

① Recursive Event-wise processing

Batch Processing (PointNet) $e:(x,y,p,\Delta t)$

$$f(\mathbf{e}_j)=g(\max(h(e_{j-n(j)+1}),\ldots,h(e_j)))$$

$n(j)$ events (τ ms)

Recursive Processing (EventNet)

① Recursive Event-wise Processing of h and max

Recursive computation, while keeping temporal information

$$h(e_j)\rightarrow c(h(e^-_j),\Delta t_j)\quad e^-:(x,y,p)$$

Avoid re-computation of h

Note: This alone does NOT solve the problem

$$f(\mathbf{e}_j)=g(\max(c(s_{j-n(j)+1},\delta t_{j-1}),\ldots,c(h(e^-_j),0)))$$

$n(j)$ events (τ ms)

$$c(z_i,\Delta t_{j,i})=\left[|z_i|-\frac{\Delta t_{j,i}}{\tau}\right]^+\exp\left(-\mathrm{i}\frac{2\pi\Delta t_{j,i}}{\tau}\right)$$

Make c and max recursive

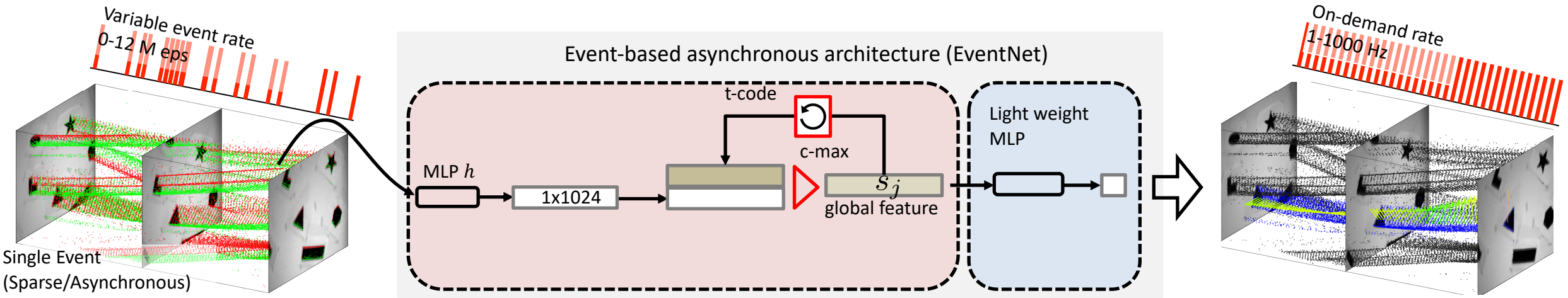
$$f(\mathbf{e}_j)=g(\max(c(s_{j-1},\delta t_{j-1}),h(e^-_j)))$$

1 event

EventNet

$$f(\mathbf{e}_j) \approx g(\max(c(s_{j-1}, \delta t_{j-1}), h(e^-_j)))$$

Recursive processing



Asynchronous recursive event-wise processing

Composition of c and $\max$ is recursive $\rightarrow n(j) \times$ faster than batch

General max

$$\max(\mathbf{a}_j) \neq \max(\max(\mathbf{a}_{j-1}, a_j))$$

Temporal coding/max in EventNet

$$\max(\mathbf{a}_j) = \max(c(\max(\mathbf{a}_{j-1}), \delta t_j), a_j),$$

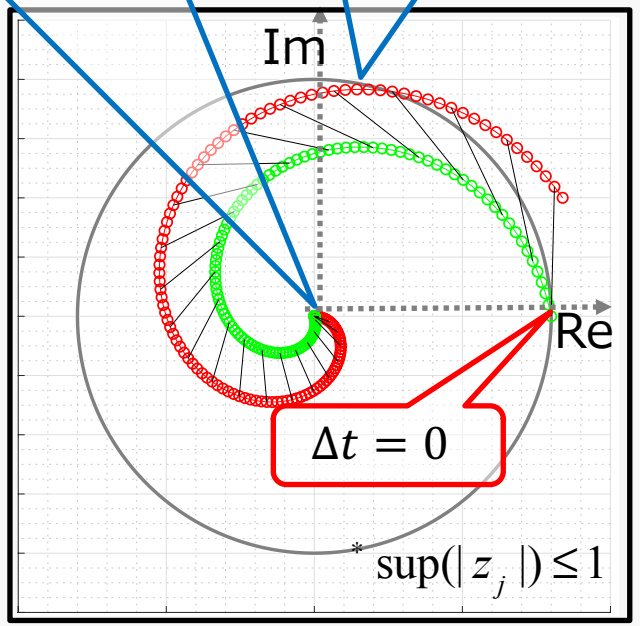
$$c(z_i, \Delta t_{j,i}) = \left[|z_i| - \frac{\Delta t_{j,i}}{\tau} \right]^+ \exp\left(-\mathrm{i} \frac{2\pi \Delta t_{j,i}}{\tau} \right)$$

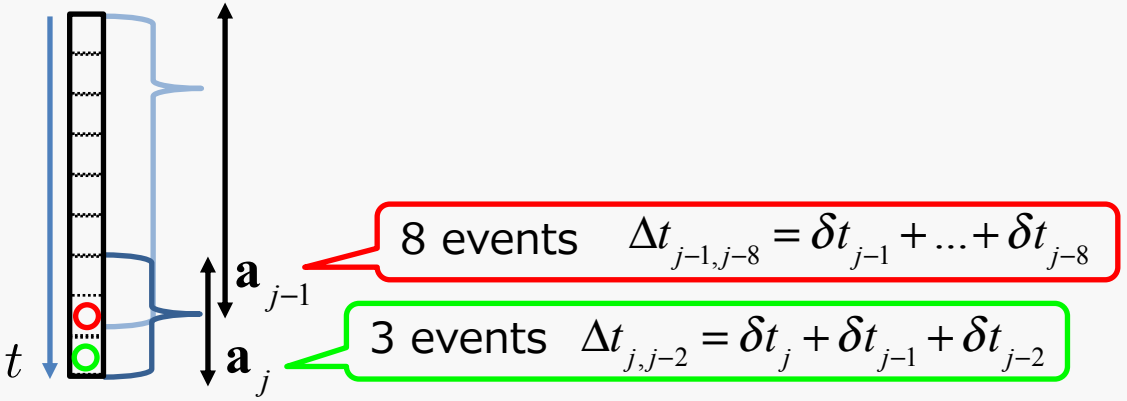
$$\Rightarrow c(z_i, \Delta t_{j,i}) = c(...c(z_i, \delta t_i)..., \delta_j)$$

† $c(\cdot) \in \mathbb{C}$

Only new event can be max

Event older than τ die ($\Delta t \leq \tau$)





Length of a_j is variable for fixed τ

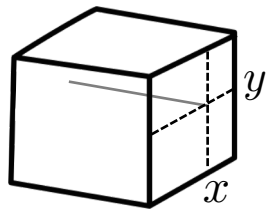
Note: Exp/linear decay can NOT

② LUT Realization of MLP h

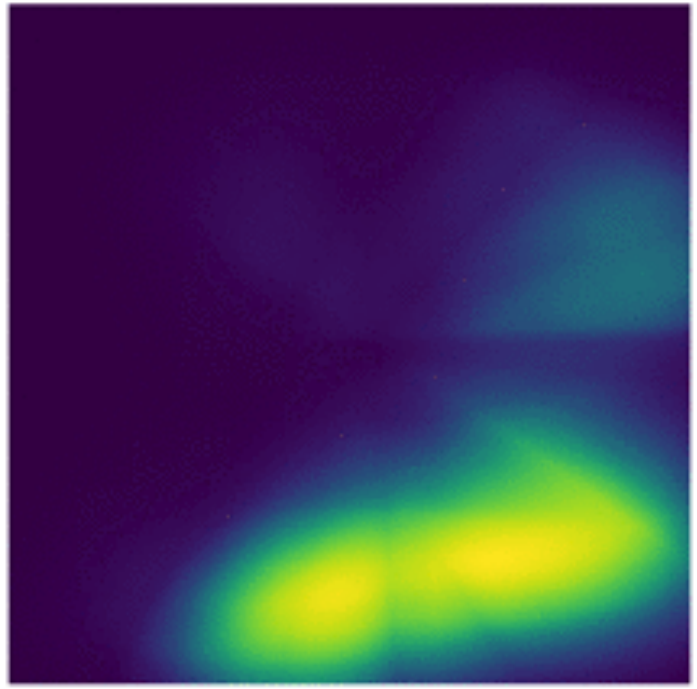
$$f(\mathbf{e}_j) \approx g(\max(c(s_{j-1}, \delta t_{j-1}), h(e^-_j)))$$

$h(x, y, p) =$ $a_j \in \mathbb{R}^K$

mlp (64,64,64,128,1024)



$a_j \in \mathbb{R}^K$

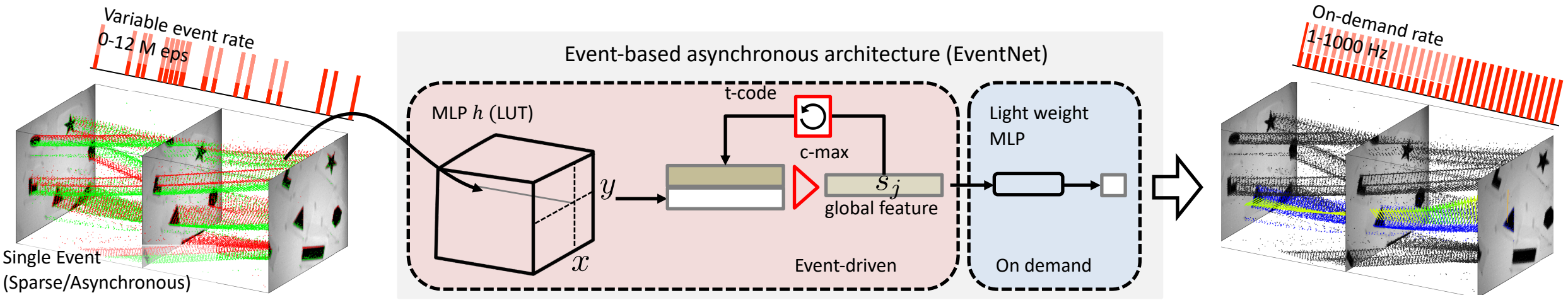


Inputs to h are discrete → 45× faster than MLP

③ Asynchronously Two Module Architecture

$$f(\mathbf{e}_j) \approx g(\max(c(s_{j-1}, \delta t_{j-1}), h(e^-_j)))$$

$$s_j = \max(c(s_{j-1}, \delta t_{j-1}), h(e^-_j)); \quad y_j = g(s_j)$$



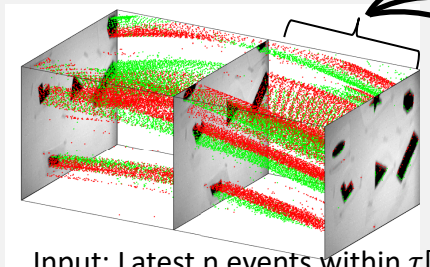
APP rate (60Hz) << Event rate (1MEPS)
Compute output on demand

Asymmetric Training/Inference

PointNet (train/inference)

$f(\mathbf{e}_j) \approx g(\max(h(e_{j-n(j)+1}),\ldots,h(e_j)))$

$(x, y, p, \Delta t)$



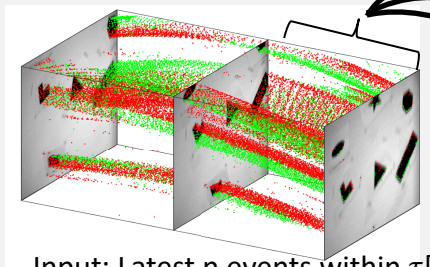
Input: Latest n events within τ[ms]

mlp1 (64,64) | shared | nx64 | mlp2 (64,128,1024) | shared | nx1024 | max | 1x1024 global feature | mlp3 (512,256,k) | 1xk

EventNet (train)

$f(\mathbf{e}_j) \approx g(\max(c(s_{j-1}, \delta t_{j-1}), h(e^-_j)))$

(x, y, p)

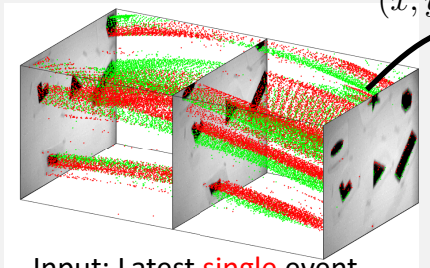


Input: Latest n events within τ[ms]

mlp1 (64,64) | shared | nx64 | mlp2 (64,128,1024) | shared | nx1024 | t-code | c-max | s_j | 1x1024 global feature | mlp3 (512,256,k) | 1xk

EventNet (inference)

(x, y, p)



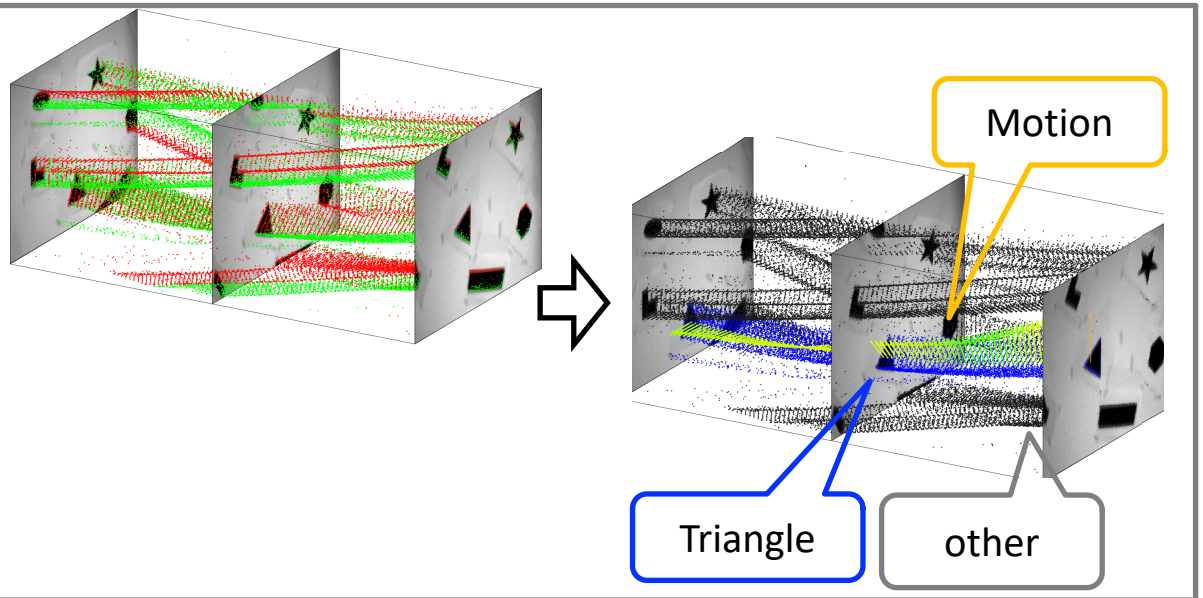
Input: Latest single event

table (mlp1) | table (mlp1/2) | t-code | c-max | s_j | mlp4 (512,256,128,128,l) | 1x64 | 1x1024 | 1xl

Experiments

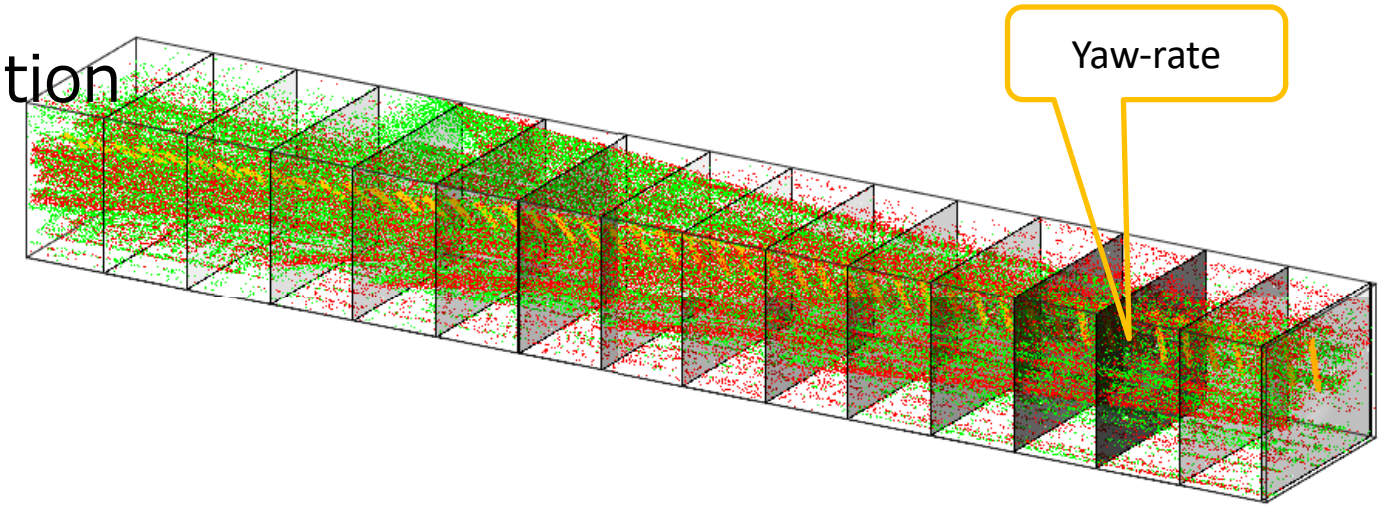
1 (ETHTED+)

- Object motion estimation
- Event-wise Semantic Segmentation



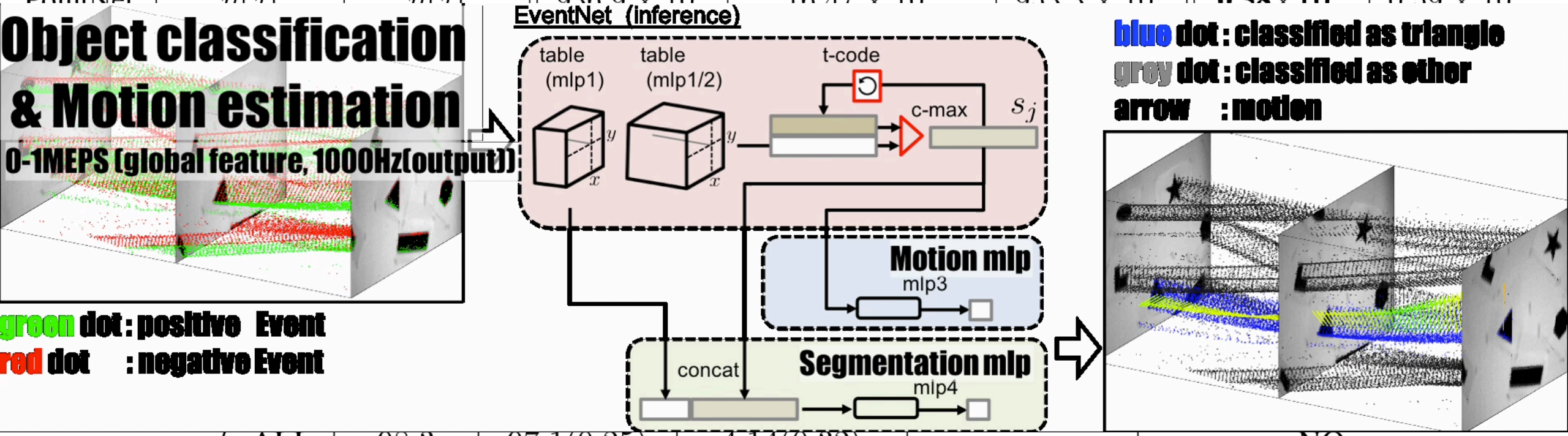
2 (MVSEC)

- Ego-motion (yaw-rate) estimation



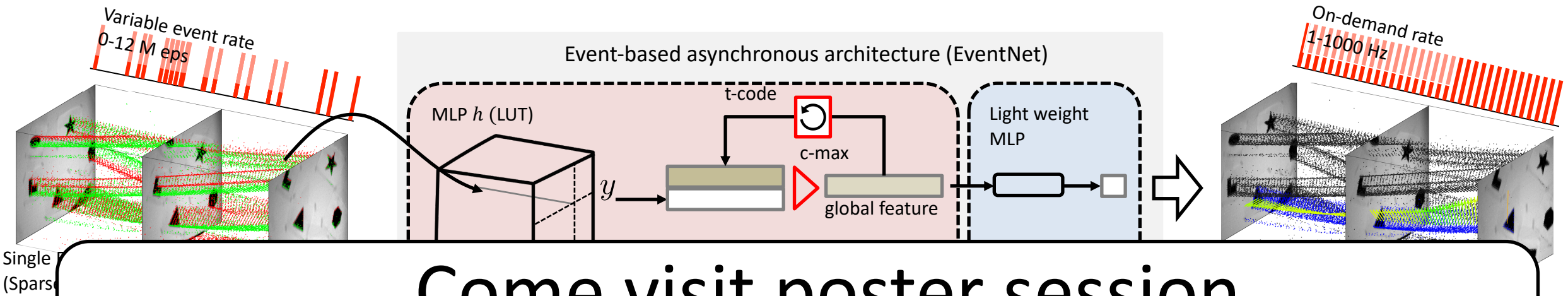
Results

	#input mlp1	#input max	mlp1/2	max pool(+t-code)	total	mlp3	mlp4
PointNet	$n(j)$	$n(j)$	936.9×10^3	16.47×10^3	953.3×10^3	$\mathbf{0.58 \times 10^3}$	0.59×10^3
w/o ALL	98.3	97.1(0.25)	4.14(0.32)			NO	



$\approx n(j) \cdot 45\times$ faster than batch/MLP processing
→ Realize 1us/event (1MEPS) with comparable performance

EventNet: Contribution Summary



Come visit poster session

#160 1-2P (Tue PM)

A neural network designed for real-time processing of asynchronous event streams in a recursive and event-wise manner

① Recursive Event-wise processing

② LUT Realization of MLP

③ Asynchronously Two Module Architecture