Supplementary Material – A Hybrid ANN-SNN Architecture for Low-Power and Low-Latency Visual Perception

Asude Aydin Mathias Gehrig Daniel Gehrig Davide Scaramuzza
Robotics and Perception Group, University of Zurich, Switzerland

1. Appendix

1.1. Overview

Here, we provide additional information supporting the main manuscript. In what follows we will refer to Figures, Tables, Sections, and Equations from the main manuscript with the prefix “M-”, and use no prefix for new references in the appendix. We start by providing further analysis of our state initialization scheme (Sec. 1.2), then provide additional network details in Sec. 1.3. We also attach a supplementary video to visualize our low-latency human pose estimation network’s output.

1.2. State Initialization Analysis

1.2.1 Initialized State Values

In Fig. 1 we show the distribution of initial membrane potentials predicted by our ANN, grouped by the encoder, residual blocks, decoder, and last initialized layer before the output. Note that the firing threshold is 1, meaning that certain neurons are initialized in a firing state. In particular, the output layer shows a high proportion of these kinds of neurons. We call these states that are initialized close to firing, or even in a firing state meta-stable. This meta-stable state is important to reduce latency since it means that few input events can immediately elicit a network response since the membrane potentials are close to firing. We also see a long tail of inhibited neurons that are initialized with a negative membrane potential.

1.2.2 Effect of Last Layer State Initialization

We test the impact of the membrane potential initialization in the first layers on the SNN results by initializing only the last layer with learned membrane potentials and the rest with zeros. We see that, in fact, the initial layers influence the SNN performance significantly, as can be seen in Fig. 2.

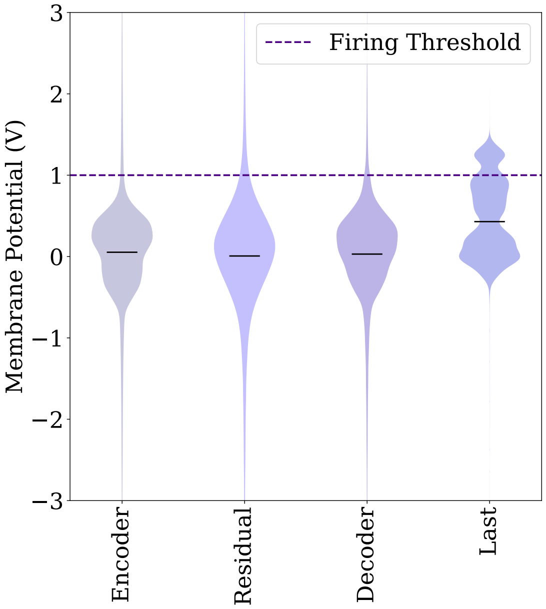

Figure 1. **Violin plots of membrane potential values after state initialization across layers of the SNN.** Black lines indicate the mean state value at every layer. ‘Last’ indicates the last state initialized layer before the output.

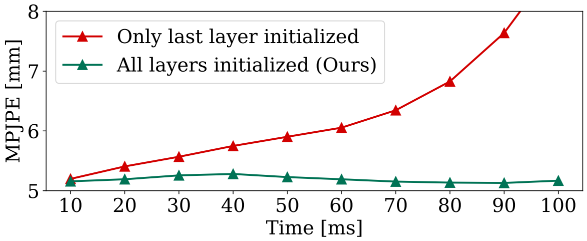

Figure 2. **Effect of last layer state initialization on performance across time steps.**

1.2.3 Visualization of Last Layer State Initialization

In Fig. 3, we plot the initialized membrane potentials of multiple channels of the last layer for additional insights. We observe that each channel raises the membrane potential more in the subject's projection (foreground) than in the background, leading to a greater likelihood of spikes in the foreground. Moreover, while each channel dampens potential keypoint locations, they tend to activate more intensely at certain keypoints, indicating that channels seem to specialize in specific subsets of keypoints.

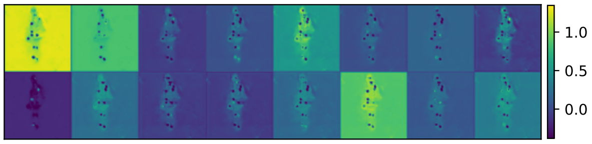

Figure 3. **Visualization of initialized membrane potentials for multiple last layer channels.**

1.3. Network Details

The CNN and SNN architecture details are given in Tables 1 and 2, respectively. For each layer, padding is calculated to preserve spatial dimensions. Both tables are given with respect to the resolution of the DHP19 dataset, 256x256. For the Event-Human3.6M dataset, the resolution is 320x256.

Each convolutional layer in the CNN is followed by a leaky ReLU layer with a negative slope of 0.1. Columns 1-6, are the encoder layers where an average pooling layer is followed by two convolutions. Columns 7-11 are decoder layers, and operations are as follows: (i) interpolation, (ii) convolutional layer, (iii) concatenation with skip connections of the same resolution, and (iv) convolution. Finally, the last column is a simple prediction layer with no activation function.

Each convolutional layer in the SNN is followed by a batch norm, and leaky integrate & fire neuron layer. The first column is the spike encoder, columns 2-4 are encoder, 5-6 are residual, and 7-9 are decoder layers. Decoder blocks perform concatenation with skip connections at the same spatial resolution and are upsampled together. Finally, the last layer is a single convolutional layer.

Table 1. **CNN architecture details.** Changes in spatial resolution are due to 2x2 average pooling or bilinear interpolation by a scale of 2. The input channel is of size 20 for event representations or 3 for RGB images.

Layer	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
Kernel size	7	5	3	3	3	3	3	3	3	3	3	3
Output channel	32	64	128	256	512	512	512	256	128	64	32	13
Output H, W	256	128	64	32	16	8	16	32	64	128	256	256

Table 2. **SNN architecture details.** Changes in spatial resolution are due to convolutions with stride 2 and bilinear interpolation of scale 2. The input channel is of size 2.

Layer	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Kernel size	5	5	5	5	3	3	5	5	5	1
Stride	1	2	2	2	1	1	1	1	1	1
Output channel	32	64	128	256	256	256	128	64	32	13
Output H, W	256	128	64	32	32	32	64	128	256	256