VI-Trajectory to Binary Image algorithm is based on the following instructions:

In order to effectively handle the difference between V–I trajectories of PELs within the same category, this paper proposes to first map the V–I trajectory to a grid of cells. Each cell is assigned a binary number. If the V–I trajectory cross though a cell, this cell is occupied by this V–I trajectory, assigned 1, and shown as a solid block as shown in Fig. 2.

The binary mapping algorithm is defined as follows:

1) Load Voltage and Current Waveforms:

Assume that there are a total of K data points of the form (v_k , i_k), where k = 1,...,K. Also, v_k and i_k are the voltage and current values of data point k, respectively.

2) Calculate

$$v_{\max} = \max v_k, v_{\min} = \min v_k$$
$$i_{\max} = \max i_k, i_{\min} = \min i_k$$
$$v_0 = \frac{1}{2}(v_{\max} + v_{\min}), \text{ and } i_0 = \frac{1}{2}(i_{\max} + i_{\min}). \quad (1)$$

$$v_{\max} = \max v_k, v_{\min} = \min v_k$$
$$i_{\max} = \max i_k, i_{\min} = \min i_k$$
$$v_0 = \frac{1}{2}(v_{\max} + v_{\min}), \text{ and } i_0 = \frac{1}{2}(i_{\max} + i_{\min}). \quad (1)$$

3) Read input N, which defines the size of the grid in the horizontal direction.

4) Calculate

$$\Delta v = \frac{v_{\max} - v_0}{N}, \Delta i = \frac{i_{\max} - i_0}{N} \quad (2)$$

and generate two sequences,

$$\{v_0 - N \cdot \Delta v, \ldots, v_0 - \Delta v, v_0 + \Delta v, \ldots, v_0 + N \cdot \Delta v\}$$

and

$$\{i_0 - N \cdot \Delta i, \ldots, i_0 - \Delta i, i_0 + \Delta i, \ldots, i_0 + N \cdot \Delta i\}$$

which both have 2N elements.

5) Define a 2N x 2N grid. Cell (*x*th, *y*th) is assigned a positional value

($v0 + \Delta v \cdot (x - N)$, $i0 + \Delta i \cdot (y - N)$),

and a binary model value Bx,y which is initialized to be 0.

6) Load half-cycle of data points, starting from the zero-crossing point from negative to positive to another zero-crossing point from positive to negative.

7) Start with the first data point (v_1^h, i_1^h) of the data points loaded in 6, and execute the following loop:

for every cell $(N + 1, y)$, $y = N + 1, N + 2, \ldots, 2N$
  if $(v_1^h - v_0) < \frac{\Delta v}{2}$ and $(i_1^h - (i_0 + (y - N) \cdot \Delta i)) < \frac{\Delta i}{2}$
    cell $(N + 1, y)$ is occupied and $B_{\Delta+1,y} = 1$;
   cell $(N + 1, y)$ is stored as the *winner* of(v_1^h, i_1^h);
    break;
  end
end

8) For the remaining data points from 6, repeat 7 by searching the eight adjacent cells of the previous winner/

9) Repeat 7 for a predefined number of times