Algorithm 1: Find candidate disks with plane sweeping technique

Input: $\mathcal{T}[t_i]$: positions in timestamp t_i, sorted by x-axis values, ϵ: flock diameter, μ: minimum size of flock
Output: $\mathcal{C}$: candidate disks for timestamp t_i, $\mathcal{B}$: active boxes in timestamp t_i

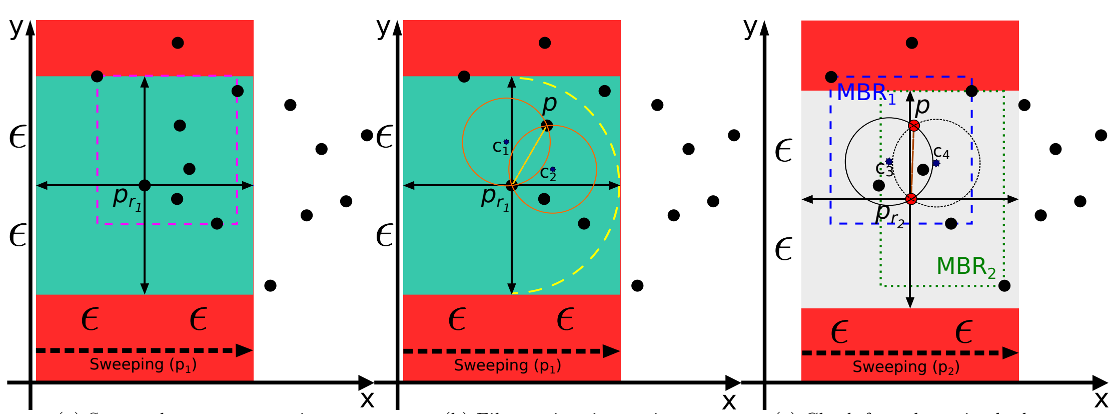
1 $\mathcal{C} \leftarrow \emptyset$, $\mathcal{B} \leftarrow \emptyset$
2 **foreach** $p_r \in \mathcal{T}[t_i]$ **do** `// analyze elements in increasing x-values`
3     $\mathcal{P} \leftarrow \emptyset$ `// list of elements of current box defined by` p_r
4   **foreach** $p_s \in \mathcal{T}[t_i] : |\ p_s.x - p_r.x\ | \leqslant \epsilon$ **do** `// test only elements inside` 2ϵ `x-band`
5   **if** $|p_s.y - p_r.y| \leqslant \epsilon$ **then** `// check if` p_s `is inside` 2ϵ `y-band`
6             $\mathcal{P} \leftarrow \mathcal{P} \cup p_s$ `// add element` p_s `to box`
7   **foreach** $p \in \mathcal{P} : p.x \geqslant p_r.x$ **do** `(// elements inside right half of box`
8   **if** $dist(p_r, p) \leqslant \epsilon$ **then** `// calculate pair distance`
9   let $\{c_1, c_2\}$ be disks defined by $\{p_r, p\}$ and radius $\epsilon/2$
10   **foreach** $c \in \{c_1, c_2\}$ **do**
11   **if** $|c \cap \mathcal{P}| \geqslant \mu$ **then** `// check the number of entries in disk`
12                     $\mathcal{C} \leftarrow \mathcal{C} \cup c$ `// add` c `to candidate disks`
13                     $\mathcal{B} \leftarrow \mathcal{B} \cup box(p_r)$ `// add box to active boxes`
14 **return** $\mathcal{C}, \mathcal{B}$

4.1 Plane Sweep-based Disk Detection

As previously described, the BFE algorithm (and its extensions) first constructs a grid-based index and then generates candidate disks for each timestamp. This process of building and searching the index can be time consuming. Thus, to reduce this cost we propose a new approach based on plane sweeping to find flock disks without index construction. Our proposed approach is described in Algorithm 1.

Algorithm 1 first sweeps the plane (from left to right in x-axis) using a band of size 2ϵ along the x-axis centered at a point p_r (red box in Figure 3(a)). The algorithm selects all the points inside the band that are in the range $[p_r.y - \epsilon, p_r.y + \epsilon]$ (blue box in Figure 3(a)). These steps are presented in lines 4–6 of Algorithm 1.

After selecting the points in the $2\epsilon \times 2\epsilon$ box defined by p_r, we then check for pairs of points that qualify for new flock disks (refer to Theorem 3.1). Thus, we generate disks defined by p_r and any point p inside the right half of box such that the distance between p_r and p is at most ϵ (yellow-dashed semicircle in Figure 3(b)). Points in the left half of box were checked in previous steps. If a candidate disk contains at least μ entities inside it, then the underlying entity set is reported as a candidate set and the box is set active in the timestamp. Every active box is represented through the Minimum Bounding Rectangle (MBR) enclosing its elements (dashed/dotted rectangles in figures). These last steps are represented by lines 7–13 of Algorithm 1.



(a) Sweep plane across x-axis. (b) Filter points in y-axis. (c) Check for subsets in the boxes.

Fig. 3. Steps needed to find disks in one timestamp.