Optimality of the Distance Dispersion Fixation Identification Algorithm

Beibin Li, Quan Wang, Laura Boccanfuso, Frederick Shic
Child Study Center, Yale School of Medicine, Yale University

Motivation

The goal of fixation identification is to reduce the complexity of eye-tracking data while maintaining the essential components for cognitive and visual processing analyses. Fewer fixations (reduce complexity) but longer fixation time (essential components) is good criteria.

I-DD: Distance Dispersion Fixation Identification Algorithm. Any two points inside the same fixation should be at most $\epsilon°$ apart.

Goal G1: maximize total fixation time. In theory, a fixation should be identified when the $\epsilon°$ criterion is met. A maximal total fixation time criterion is closely aligned with this principle.

Goal G2: minimize the number of fixations. This is a parsimony principle: points should be considered part of the same $\epsilon°$-limited cluster whenever possible.

Without Time Rejection

Because no rejection, every point belongs to a fixation. (saccade is fixation itself). How about # fixations?

Problem: Given an array of n gaze points $(p_1, p_2, p_3, \dots, p_n)$, we want to find the minimum number of fixations (clusters) according to the I-DD definition that cover all gaze points.

Algorithm: We will use set S to record all fixations in our algorithm, where set S contains the start points of every fixation. Once we know where a fixation starts, we can know which points are covered by the fixation, because the greedy I-DD will group all of the possible points that the first point can cover.

Pseudo Code:

1. $S = \{\}$; // empty set
2. $i = 1$;
3. $S = S \cup \{ p_i \}$
4. Using distance dispersion threshold, find all the consecutive points (i.e. $p_{i+1}, p_{i+2}, \dots, p_{i+j}$) that can be covered by the same fixation as p_i .
5. $i = i + j + 1$;
6. If $i < n$, Goto Line 3; else, output set S.

Claim: The Greedy I-DD described above produces an optimal solution for G2, i.e. minimum number of fixations.

Proof: Prove by contradiction. Assume optimal solution is $S*$, and $|S*| < |S|$, where S is the output of our algorithm.

Sort all points in S and $S*$ in time order. $S* = \{p_a*, p_b*, p_c*, \dots,\}$, $S = \{p_a, p_b, p_c, \dots,\}$.

Without loss of generality, assume the first point differs in S and $S*$ are p_d* and p_d.

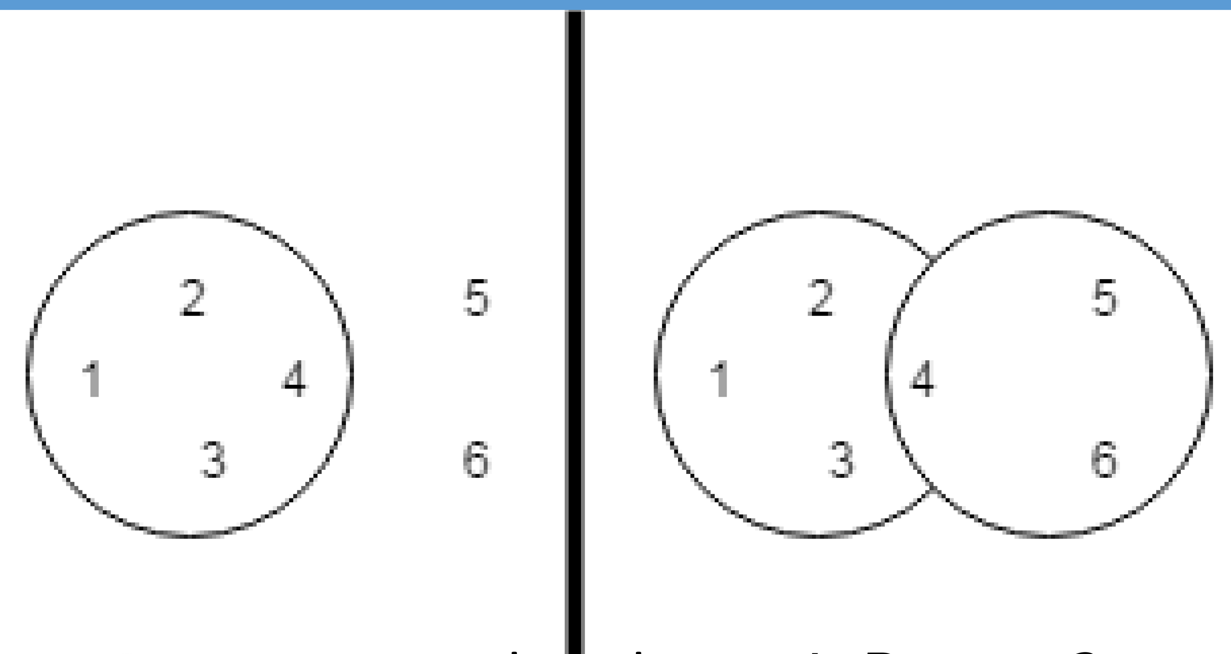
It's easy to see $p_d* < p_d$; otherwise, some points between p_d* and p_d are not covered in $S*$.

Therefore, p_d's fixation covers more than p_d*'s fixation, which means $p_e* < p_e$.

For the same reason, $p_f* < p_f$. S will cover faster than $S*$, and $S*$ is no better than S.

The assumption is false. Thus, our algorithm produces the optimal result.

With Rejection (Dynamic Programming)



Counter example, when $minPts = 3$

Greedy I-DD is not optimal now, as shown. However, the following DP algorithm is.

Algorithm Idea: Reject all groups of gazes that have less than λ points and identify them as saccades in our algorithm.

Define $f(i)$ = the index of first point that point p_i can cover reversely (run the original greedy algorithm in reverse).

Define $S(i)$ = the optimal solution (number of points covered by all fixations) for points $p_1, p_2, p_3, \dots, p_i$.

Then, we have the relationship:

$$S(i) = \max_j \left[S(j) + \begin{cases} i - j & if (i - j \geq \lambda) \\ 0 & otherwise \end{cases} \right],$$

where j satisfies $f(i) - 1 \leq j < i$

Runtime: build $f(n)$ takes $O(n^2)$ time, and build $S(n)$ also takes $O(n^2)$ time. So, the total runtime is $O(n^2 + n^2) = \boldsymbol{O(n^2)}$ time. This algorithm will find fixations that maximized the total fixation time under time rejection criteria.

Conclusion

Without time rejection, greedy I-DD algorithm satisfies both G1 and G2. With time rejection, the dynamic programming version could be optimal for G1.

Further studies should try to discover more standards to evaluate fixation identification algorithms. Further studies can also combine different standards together.

Most fixation identification algorithms can be classified as either dispersion-based or velocity-based algorithms. Researchers can try to find other types of algorithms.

Acknowledgement: K01 MH104739, R21 MH102572, CTSAUL1 RR024139, R03 MH092618, NIH R01 MH100182, R01 MH087554, U19 MH108206; NSF #1139078, #0835767, DOD W81XWH-12-ARP-IDA, The Nancy Taylor Foundation for Chronic Diseases , and the Associates of the Child Study Center.

Special thanks to Erin Barney for proofreading the manuscript.

Beibin.li@yale.edu Frederick.shic@yale.edu