GUIDe SACCADE DETECTION AND SMOOTHING ALGORITHM

Manu Kumar

Stanford University, HCI Group

sneaker@stanford.edu

Data from an eye tracker is noisy and includes jitter due to errors in tracking and because of the physiology of the eye. To smooth the data from the eye tracker, it is necessary to determine whether the most recent data point is the beginning of a saccade or whether it is an aberration relative to the current fixation. Before returning a data point, our algorithm looks at the subsequent data point to make decisions about whether the current data point was the beginning of a saccade or an aberration, in which case it is adjusted.

The algorithm maintains two sets of points: the current fixation window and a potential fixation window. If a point is close to (within a saccade threshold) the current fixation, then it is added to the current fixation window. The new current fixation is calculated by a weighted mean which favors more recent points (see below). If the point differs from the current fixation by more than a saccade threshold, then it is added to the potential fixation window and current fixation is returned. When the next data point is available, if it was closer to the current fixation, then we add it to the current fixation and throw away the potential fixation as an aberration/noise. If it is closer to the potential fixation, then we add the point to the potential fixation window and make this the new current fixation.

The fixation point is calculated as a weighted mean of the set of points in the fixation window. The weight assigned to each point is based on its position in the window. For a window with n points (p_0, p_1... p_{n-1}) the mean fixation would be calculated by the formula:

$$p_{mean} = \frac{1p_0 + 2p_1 + \ldots + nP_{n-1}}{(1 + 2 + \ldots + n)}$$

The graph in Figure 1 shows the output from the smoothing algorithm for the x-coordinate of the eye-tracking data. The blue line shows a trace of the raw data from the eye-tracker and the red line shows the smoothed result based on the algorithm above. It should be noted that when a saccade does occur, the smoothed data lags the raw data by one data point since it needs to wait for the next datapoint to confirm whether the previous point was an aberration or the beginning of a saccade.

While there is still room for improvement in the algorithm above by taking into account the directionality of the incoming data points, we found that this smoothing algorithm significantly improved the reliability of the results for applications which rely on the real-time use of eye-tracking data.
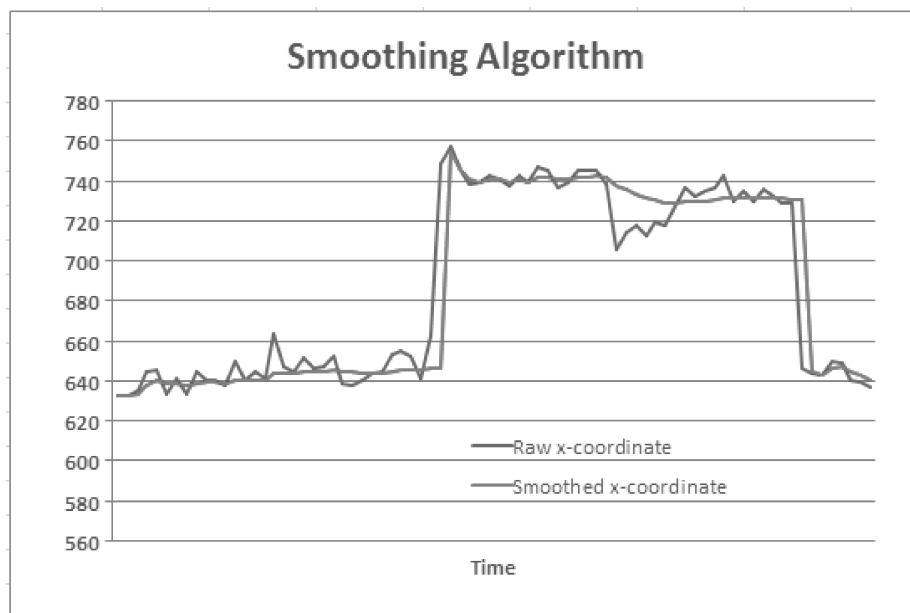

Figure 1. A graphical view of the x-axis data and smoothing from the eye tracker.

```
point FilterData(point)
   if current fixation is empty add the point to current fixation and return
   if current fixation has points and potential fixation has points
   calculate the distance of the point from each fixation
   if the new point is closer to the current fixation
      if the distance is less than the saccade threshold
         add the point to the current fixation
         clear the potential fixation
         return the new current fixation
      else
         clear the potential fixation
         add the point to the potential fixation
         return the current fixation
   else
      if the distance is less than the saccade threshold
         add the point to the potential fixation
         make potential fixation the new current fixation
         clear the potential fixation
         return the current fixation
      else
         save the potential fixation
         clear the potential fixation
         add the point to the potential fixation
return the saved potential fixation
   else
      if the point is beyond the saccade threshold from the current fixation
         add the point to the potential fixation
         return the current fixation
      else
         add the point to the current fixation
         return the current fixation
End FilterData
```

Listing 1. Pseudocode for Saccade detection and smoothing algorithm