Detecting Multimodal Behaviors for Neurodegenerative Disease

by

Antonio Berrones

B.S. Electrical Engineering and Computer Science
Massachusetts Institute of Technology, 2022

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2024

Authored by:	Antonio Berrones Department of Electrical Engineering and Computer Science January 26, 2024
Certified by:	Randall Davis Professor, MIT CSAIL Thesis Supervisor
Certified by:	Dana Penney Director of Neuropsychology, Lahey Hospital & Medical Center Thesis Supervisor
Accepted by:	Katrina LaCurts Chair, Master of Engineering Thesis Committee

Detecting Multimodal Behaviors for Neurodegenerative Disease

by

Antonio Berrones

Submitted to the Department of Electrical Engineering and Computer Science
on January 26, 2024, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

Neurodegenerative diseases such as Parkinson's and Alzheimer's are incurable and affect millions of people worldwide. Early diagnosis is critical for improving quality of life for patients. Current methods rely on the use of tests administered and evaluated by clinicians. The digital Symbol Digit Test (dSDT) is a novel cognitive test that aims to distinguish between individuals with normal and impaired cognitive abilities. This thesis will develop a framework for processing collected participant eye-tracking and handwriting data and show its use in detecting specific multimodal learning behaviors. Furthermore, this thesis will explore recommendations for working with eye-tracking systems and outline future steps towards developing a multimodal classification model to automate early diagnosis of neurodegenerative disease.

Thesis Supervisor: Randall Davis
Title: Professor, MIT CSAIL

Thesis Supervisor: Dana Penney
Title: Director of Neuropsychology, Lahey Hospital & Medical Center

Acknowledgments

I would like to thank my supervisors, Professor Randall Davis and Dr. Dana Penny, for their constant advice and support throughout the duration of this project. Their deep knowledge and passion for their work was always inspiring to witness. I have learned so much from them and this experience.

I also want to thank all of the people from MIT that made my time there special, despite all the stress, chaos, and surprises that came with it.

Finally, I would like to give a very special thank you to my parents. I wouldn't be where I am today without their hard-work, love, and encouragement. Thank you for always being on my side.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

According to the World Health Organization, approximately 50 million people suffer from neurodegenerative disease [4]. Two of the most common types are Parkinson's and Alzheimer's, for which no cures currently exist. Treatment consists primarily of medication and physical therapy to manage symptoms and improve quality of life. Early diagnosis is crucial for ensuring better outcomes for patients [7].

Typically, diagnosis consists of clinical evaluations and diagnostic tests administered by trained physicians. However, these methods tend to be subjective and lack reproducibility, with diagnoses varying from physician to physician [5].

In order to address these limitations, this thesis aims to create a data processing pipeline for detecting specific learning behaviors from participants who have completed the digital Symbol Digit Test (dSDT), administered through the Cognitive Health App on an iPad with a stylus. These learning behaviors are defined in terms of participants' point of gaze and handwriting activity throughout the duration of the test. By acting as descriptors of learning, the detection of these behaviors will aid in the further development of a multimodal classification model for the early detection of neurodegenerative disease.

Eye-tracking data is collected via the ETVision system, developed by Argus Science, which consists of the ETVision Headset and its accompanying desktop software applications known as the ETVision App and the ETAnalysis App. Handwriting data is collected via the Cognitive Health App.

The detection of participant learning behaviors is broken down into four primary data processing pipelines (Figure 1-1). Each modality is processed separately; the eye-tracking data captured by the ETVision system is processed by the ETAnalysis Pipeline and Vision Pipeline while the handwriting data captured by the Cognitive Health App is processed by the Handwriting Pipeline. Finally, the output is processed by the Behavior Detector pipeline. The remainder of this work will explore each step in greater depth.

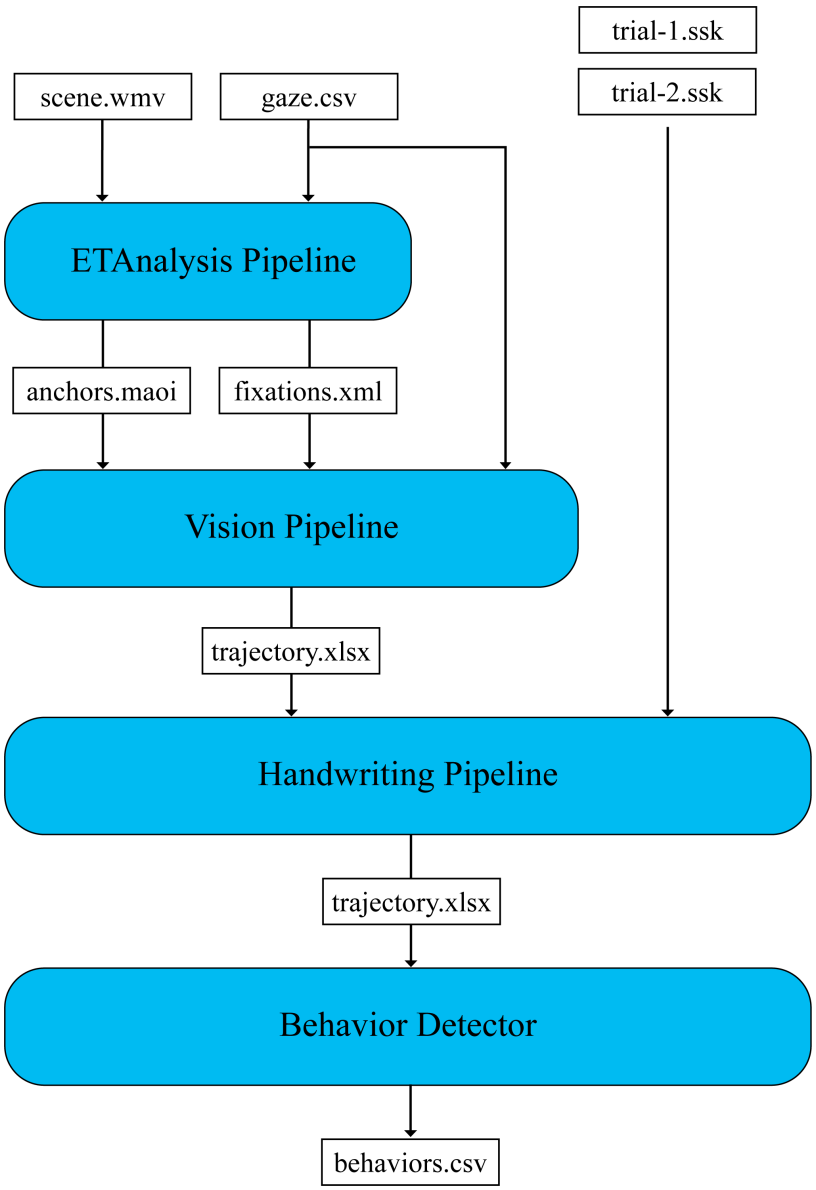

Figure 1-1: Overview of the multimodal data processing pipeline for detecting learning behaviors for a single participant.

Chapter 2

Background

2.1 The Digital Symbol Digit Test

The Digital Symbol Digit Test (dSDT) is a cognitive test developed by Dr. Dana Penny and Professor Randall Davis that highlights patterns of memory and learning to differentiate between individuals with normal and neurologically impaired cognitive states [3]. The test consists of three tasks (translation, copy, and recall) that are administered back to back over the course of two trials.

In the translation task, participants are given a key of six symbol-digit associations (Figure 2-1). A grid of cells is presented below the key, with each cell divided into top and bottom regions. The top half depicts one of the six symbols from the key, while the bottom half is blank. Participants must write the digit associated with the symbol in each cell, according to the key. The first six cells in the first row serve as the sample section, enabling participants to become familiar with the task, while the remaining cells constitute the main section of the test.

The copy task follows the same structure with one exception: the key contains digit-digit associations (Figure 2-2). The key pairs each digit with itself, allowing for participants to disregard the key and directly copy the corresponding digit into each cell. The recall task occurs directly after the copy task and consists of six cells, each containing one of the six symbols from the translation task, but in a different order. Participants are tasked with writing the digit associated with each symbol

from memory.

The copy task acts as a baseline by showing how participants respond to a low effort task that is similar to the translation task. Therefore, we can use a participant's response to the copy task as a control for their response in the translation task.

The recall task in the dSDT acts as a final measure of whether a participant has indeed learned and remembered the symbol-digit associations encountered throughout the test.

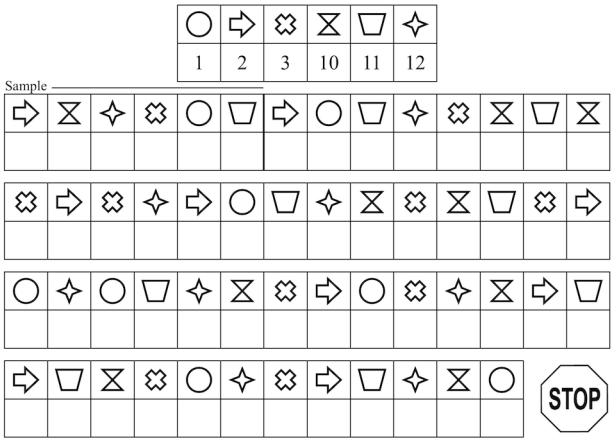

Figure 2-1: Test form for the translation task in the dSDT.

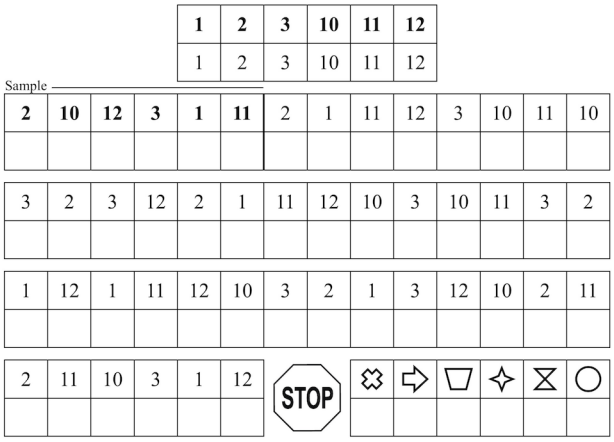

Figure 2-2: Test form for the copy and recall tasks in the dSDT.

2.2 The Cognitive Health App

The Cognitive Health App is an iOS application developed for the iPad and Apple pencil stylus [2, 1]. The purpose of the application is to allow users to self-administer a series of different cognitive tests, one of which is the dSDT. The application uses a generated voice, available in various languages, and guided animations to provide verbal and visual instructions on how to complete the test.

The application is also responsible for capturing the user's handwriting data, which consists of timestamped coordinates of the stylus position. Upon the completion of the dSDT, the captured handwriting data is saved to what we call a *SSK file* (symbol-digit sketch file). It is then encrypted and sent to a secure private server.

2.3 The ETVision System

The ETVision system is an eye-tracking system developed by Argus Science. The system consists of the ETVision Headset and its accompanying hardware, as well as two software applications known as the ETVision App and the ETAnalysis App.

The ETVision Headset uses two pupil-facing cameras and near infra-red LEDs to track the user's point of gaze using the Pupil to Corneal Reflections (CR) technique. It also has a front-facing scene camera, allowing us to determine what they are looking at.

The ETVision App is used in conjunction with the ETVision Headset for recording eye-tracking data, which consists of the coordinates of the user's point of gaze along with other measurements. The system saves this data as a series of files: two pupil-facing videos, a 1280x720px scene video (recorded at 30 frames per second) as a WMV (Windows Media Video) file, and the eye-tracking data as a CSV (comma-separated values) file recorded at a rate of 180 samples per second. The ETAnalysis App provides additional tools for further analyze eye-tracking data. For example, it allows us to detect fixations and track objects in the scene video.

Chapter 3

Prior Work

The work outlined in this thesis involves the detection and visualization of learning behaviors first described in the previous work cited below.

3.1 Learning Patterns and Behaviors

Prior work done by Sarkar outlines a series of behaviors observed by participants undergoing the dSDT [6]. The presence and number of occurrences of these learning behaviors are useful in two ways: they can be used as higher-level features for detection of neurodegenerative disease, and they can be used as a basis for further development for quantifying real-time learning. Table 3.1 provides a summary of the learning behaviors specific to the translation task.

The definitions of these behaviors depend on the knowledge of certain terms. The *stimulus cell* is defined to be the cell in the dSDT that the participant is currently trying to find the correct symbol-digit mapping for. Additionally, *fixations* are brief intervals of time, typically in order of milliseconds, where eye gaze is relatively stationary.

Behavior	Description	Types
Visual Match	Fixation on another cell with the same symbol as the cell that has to be filled out.	*key-match*, *cell-match*, *no-match-needed*
Concentrated Match	Specific visual match with respect to other cells.	*concentrated*
Forward and Backward Fixation	Looking back at cells to the left or ahead at cells to the right in the same row.	*backward-fixation*, *forward-fixation*
Scans	Consecutive fixations across three or more adjacent cells.	*right-key-scan*, *right-same-row-scan*, *left-key-scan*, *left-same-row-scan*
Back-and-Forth	Successive fixations in the format: x-y-x, where x and y are different cells.	*key-key*, *cell-cell*, *key-cell*, *confirmation*
Spatial Association	First fixation in same spatial section as cell to be filled out.	*spatial-association*

Table 3.1: Previously defined learning behaviors in translation task.

3.1.1 Visual Match

We define a "Visual Match" to mean the behavior in which the participant looks at a cell that has the same symbol as the current stimulus cell. We have also defined three subtypes of Visual Matches. A *key-match* is a visual match where the cell that the participant looks at lies in the key portion of the test. A *cell-match* is a visual match where the cell the participant looks at is not a key cell. A *no-match-needed* visual match is when no visual match occurred for a particular stimulus cell, presumably because the participant remembers what digit goes with that symbol and as a result, does not need to find a match.

3.1.2 Concentrated Match

We define a "Concentrated Match" to mean the behavior in which a Visual Match occurs immediately when starting a new stimulus cell. We have also defined two subtypes of Concentrated Matches. A *concentrated-key-match* is when the Visual Match in question is a *key-match*. A *concentrated-cell-match* is when the Visual Match in question is a *cell-match*.

3.1.3 Forward and Backward Fixations

We define a "Forward Fixation" to mean the behavior of a participant fixating in a cell that lies in the same row and to the right of the current stimulus cell. In contrast, we define a "Backward Fixation" to mean the behavior of a participant fixating in a cell that lies in the same row and to the left of the current stimulus cell. There are no subtypes for these behaviors.

3.1.4 Scans

We define a "Scan" to mean the behavior of a participant making consecutive glances across three or more adjacent cells in a single direction. We have also defined four subtypes of this behavior. A *right-key-scan* is a scan that occurs in the key in a

rightward direction. A *right-same-row-scan* is a scan that occurs in non-key cells in a rightward direction. A *left-key-scan* is a scan that occurs in the key in a leftward direction. A *left-same-row-scan* is a scan that occurs in non-key cells in the leftward direction.

3.1.5 Back-and-Forth

We define a "Back-and-Forth" behavior as when a participant fixates on one cell, then another cell, and then back to the first cell. We have also defined four subtypes for this behavior. A *key-key* behavior is when all cells involved are in the key. A *cell-cell* behavior is when all cells involved are non-key cells. A *key-cell* behavior is when the participant looks back and forth between a cell in the key and a non-key cell. A *confirmation* behavior is *key-cell* behavior where the non-key cell is the stimulus cell. In this case, we hypothesize that the participant is looking at the stimulus cell in order to confirm their answer.

3.1.6 Spatial Association

We define a "Spatial Association" behavior in terms of the spatial section of a participant's first fixation after completing a cell. A spatial section is one of three regions of a row in the dSDT: *start*, *middle*, and *end* sections. The *start* section consists of the first five cells of a non-key cell row or the first two cells of the key. The *end* section consists of the last five cells of a non-key cell row or the last two cells of the key. The *middle* section are the remaining cells in the non-key cell row or key. There are no subtypes for this behavior.

Chapter 4

Data Collection

In April 2023 a total of 15 healthy participants from the MIT undergraduate population volunteered to undergo a series of different cognitive tests via the Cognitive Health App, one of those being the dSDT. Each of the participants were fitted with the ETVision Headset and eye-tracking data was recorded.

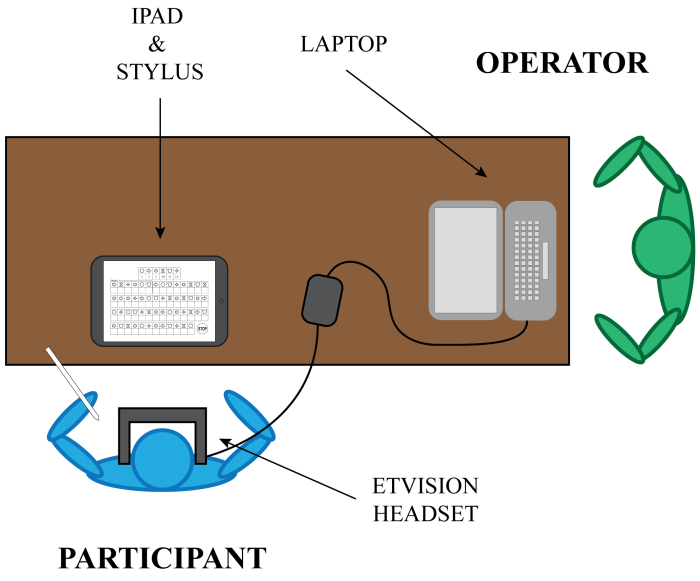

Figure 4-1: Typical experimental setup of a participant complete the dSDT with eye-tracking via the ETVision system.

The general setup for conducting the cognitive tests with eye-tracking involved the participant siting at a table with the iPad and stylus placed in front of them (Figure 4-1). To the side of the table, the operator sits with a laptop running the ETVision

App software. The operator's laptop is connected via cable to the ETVision Headset, which is worn by the participant.

Prior to data collection, the participant must have the ETVision Headset worn and setup properly. The ETVision Headset comes with various different nose-piece attachments to ensure that the participant's pupils are centered in the pupil-facing cameras point-of-view, which is required for proper eye-tracking.

The ETVision system must also be properly calibrated for each participant in order to ensure an accurate correlation between the participant's true point of gaze and the point of gaze measured by the system. The calibration procedure uses a physical plastic target that is placed in front of the participant. They are instructed to stare at its center while the operator clicks on its center in the ETVision App, which displays a live recording of the ETVision Headset's front-facing scene camera. This may need to be repeated multiple times for greater accuracy.

It is important to note that the typical use case of the ETVision system is for the eye-tracking of users that are looking forward at the world with a horizontal line-of-sight. However, the entirety of our participants' time with the ETVision system involves looking downward. Therefore, our use of this system on participants taking cognitive tests on an iPad is atypical.

Due to these difficulties, four out of the fifteen total participants tested contained data in the translation task with reliable eye-tracking for the detection of learning behaviors. All of these participants were healthy controls. Recommendations for better calibration and eye-tracking results are discussed in sections 7.2 and 7.3.

Chapter 5

Data Processing

This chapter describes the data processing pipelines developed for processing the eye-tracking and handwriting data in order to detect the learning behaviors described in section 3.1. This ultimately comes down to determining what cells in the dSDT a participant is looking at and writing in for the duration of the test.

A single recording session of a participant taking the dSDT involves two trials of the translation, copy, and recall tasks. For the purposes of detecting learning behaviors, we consider only the translation task, as neither the copy or recall tasks involve learning.

The overview described in Figure 1-1 depicts four separate data processing pipelines. The following sections describe each in greater detail.

5.1 ETAnalysis Pipeline

After the eye-tracking recording session for a participant taking the dSDT is completed, the ETVision system saves the resulting eye-tracking data to a series of files, including the *scene video* and the *eye-tracking data file*. While the *eye-tracking data file* contains multiple fields for each data sample, the fields most relevant to this work are listed in Table 5.1.

Both the *scene video* and the *eye-tracking data file* serve as the primary inputs to the ETAnalysis Pipeline. The processing done in this stage is done entirely within

FIELD	DESCRIPTION
start_of_record	Time elapsed since start of ETVision recording, in seconds.
sample_number	Current sample number since start of ETVision recording.
horz_gaze_coord	Horizontal coordinate of computed point of gaze within the scene video, in pixels.
vert_gaze_coord	Vertical coordinate of computed point of gaze within the scene video, in pixels.

Table 5.1: Most relevant fields in the *eye-tracking data file* created by the ETVision system after a recording session for a single participant.

the ETAnalysis App, outlined in Figure 5-1.

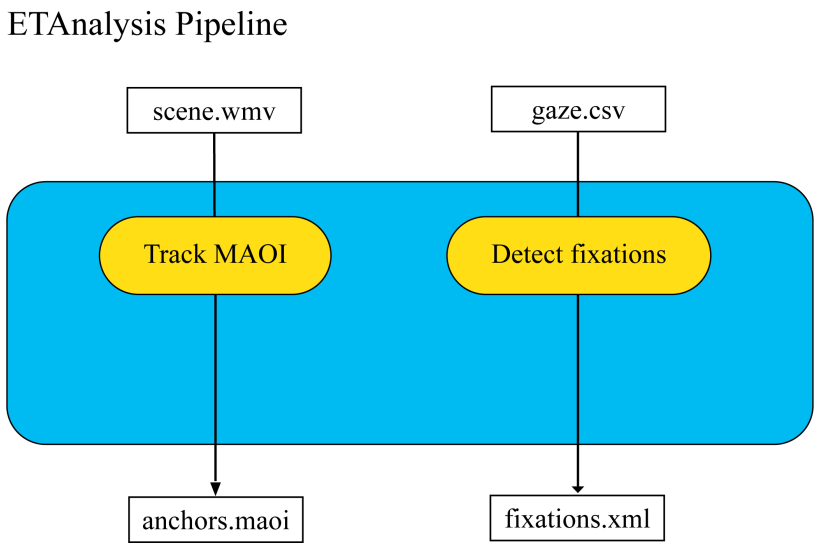

Figure 5-1: An overview of the processing done in the ETAnalysis Pipeline.

5.1.1 Computing Fixations

One of the features provided in the ETAnalysis App is a fixation detector. By using fixations (intervals of time where the point of gaze is relatively stationary) rather than point of gaze for determining participant dSDT cell locations, we prioritize instances where the participant's point of gaze is focused and intentional.

Fixation detection algorithms involve the use of various manually determined thresholds. This work uses the default thresholds and algorithm provided in the

ETAnalysis App, implemented by Argus Science. The detected fixations are saved as a separate *fixations file* in XML format. The fields in the *fixations file* most relevant to this work are listed in Table 5.2.

FIELD	DESCRIPTION
FixNumber	Current fixation number since start of ETVision recording.
StartTime	`start_of_record` field value of the first sample in the current fixation sequence.
Duration	Difference between stop and start time of current fixation sequence, in seconds.
StopTime	`start_of_record` field value of the last sample in the current fixation sequence.
HorzPos	Horizontal coordinate of average point of gaze during fixation within the scene video, in pixels.
VertPos	Vertical coordinate of average point of gaze during fixation within the scene video, in pixels.
StartSampleNumber	`sample_number` field value of the first record in the current fixation sequence.
StopSampleNumber	`sample_number` field value of the last record in the current fixation sequence.

Table 5.2: Most relevant fields in the *fixations file* created by the ETAnalysis fixation detector.

5.1.2 Tracking Moving Areas of Interest

Another important feature in the ETAnalysis App is the ability to define and track a moving area of interest (MAOI). This is done by defining the corner points of a convex polygon on any frame in the scene video. The application can then track the movement of these points across every four frames for the duration of the *scene video*. This data can be saved to a *MAOI file*, which contains the coordinates of the corner points of the MAOI along with its associated frame number.

These tracked MAOIs act as natural fiducial markers that will be used for transforming the reference frame of the fixation coordinates (see section 5.2.2). Certain structures in the dSDT (e.g. the symbol-digit key, the first cell in the sample section,

etc.) tend to remain visible during the relevant portions of the scene video. These are the structures that are selected for MAOI tracking.

As a consequence of using the ETAnalysis App for MAOI tracking, the coordinates are generated only every four frames. For every pair of frames with MAOI coordinates, data for the frames in between are interpolated in order to ensure coverage for every frame. The exact information saved for each frame in the *MAOI file* is can be viewed in Table 5.3.

FIELD	DESCRIPTION
frame_number	Current frame number since start of ETVision recording.
name	Type of MAOI to use for tracking.
maoi_x1	Horizontal coordinate of top-left corner of MAOI within the scene video, in pixels.
maoi_y1	Vertical coordinate of top-left corner of MAOI within the scene video, in pixels.
maoi_x2	Horizontal coordinate of top-right point of MAOI within the scene video, in pixels.
maoi_y2	Vertical coordinate of top-right corner of MAOI within the scene video, in pixels.
maoi_x3	Horizontal coordinate of bottom-right corner of MAOI within the scene video, in pixels.
maoi_y3	Vertical coordinate of bottom-right corner of MAOI within the scene video, in pixels.
maoi_x4	Horizontal coordinate of bottom-left corner of MAOI within the scene video, in pixels.
maoi_y4	Vertical coordinate of bottom-left corner of MAOI within the scene video, in pixels.

Table 5.3: Data represented in a *MAOI file* created by the ETAnalysis App.

5.1.3 Configurations

Because the test-taking process involves a number of different cognitive tests that must be visually and verbally explained to the participant, the screen of the Cognitive Test App changes during these instructional periods. As a consequence, the MAOI structures used for tracking are not displayed on the screen. Therefore the session

must be divided into each trial and task's sample and main sections. The capability to do this is provided by the ETAnalysis App. As a result, for a single participant eye-tracking session there are a total of four MAOI files generated for the translation task. Each (trial, section) pair is referred to as a "config" or configuration, which serves as the unit of data that all data processing pipelines operate on.

5.2 Vision Pipeline

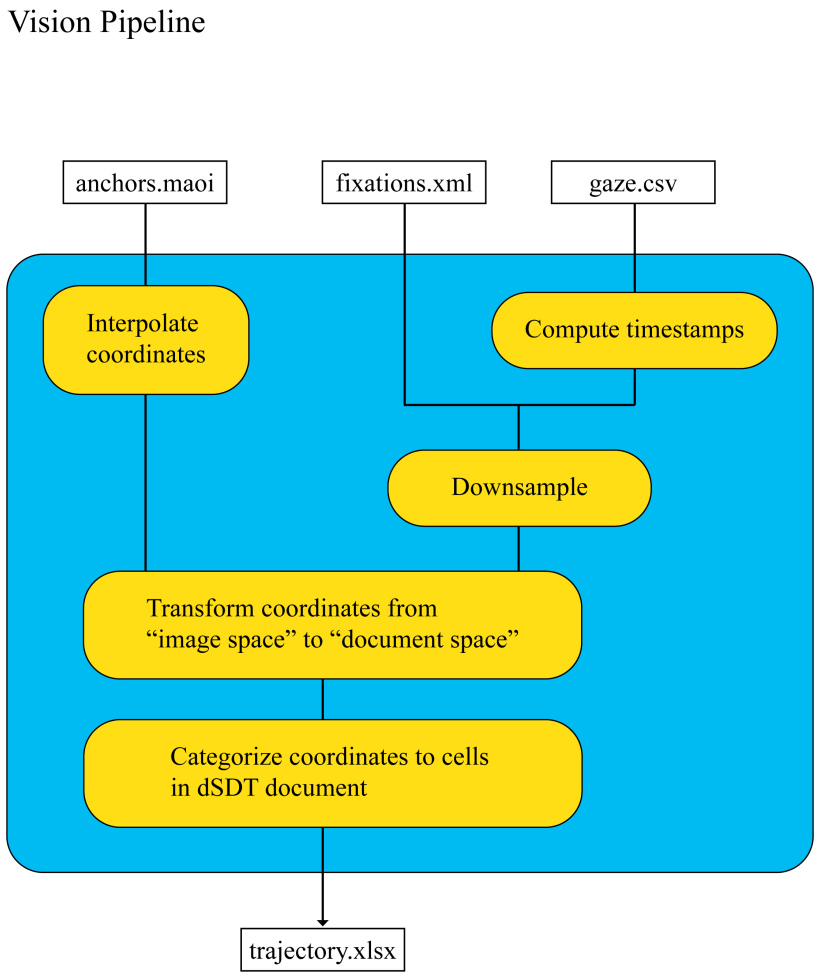

Figure 5-2: An overview of the processing done in the Vision Pipeline. The pipeline receives as input the *eye-tracking data file* (gaze.csv), *fixations file* (fixations.xml), the *MAOI file* (anchors.maoi) and outputs a single *trajectory file* (trajectory.xlsx).

This section describes the processing done to the eye-tracking data that will result in the categorization of each fixation to a particular cell on the dSDT. An overview

of this pipeline is depicted in Figure 5-2

The *eye-tracking data file* provided by the ETVision App does not contain a timestamp for each sample. However, the file does contain a timestamp indicating the moment when the ETVision system starts recording. Using this information along with the `start_of_record` field (which measures elapsed time since the start of recording), a new `timestamp` field can be created for each sample in the *eye-tracking data file*.

In order to incorporate fixation information from the *fixations file* into the *eye-tracking data file*, the `startSampleNumber` and `stopSampleNumber` fields of each fixation are used to locate their corresponding data in the *eye-tracking data file*, from which the relevant fixation-specific fields (such as the fixation's coordinates) are appended to.

5.2.1 Downsample

Prior to using the frame-level *MAOI file* (now at 30 frames per second after interpolation), the eye-tracking data (recorded at 180 samples per second) must be downsampled. This corresponds to sequences of roughly six data samples being mapped to a single sample for a single frame. We do this by using only the most recent eye-tracking sample since it was recorded closest to the start of the sequence. A depiction of this can be seen in Figure 5-3.

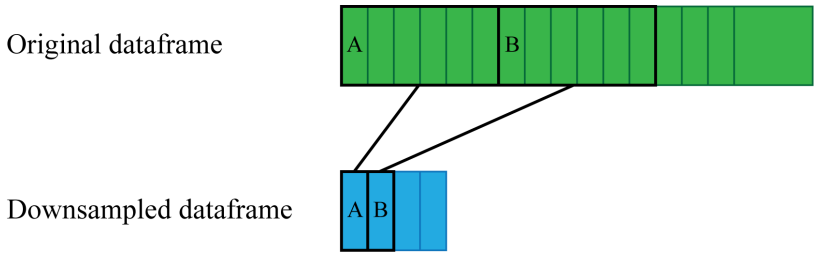

Figure 5-3: Downsampling point of gaze data from 180 samples per second to 30 samples per second, where only the earliest of the original samples is kept.

5.2.2 Reference Frame Transformation

The original fixation points provided by the ETVision system are in terms of x and y coordinates in units of pixels of the 1280x720px *scene video*. This reference frame follows the standard convention where (0,0) lies in the top-left corner. However, in order to categorize the participant's fixation points to dSDT cells they must lie in a reference frame relative to the iPad screen displaying the test document. This section will describe the method used for transforming fixation coordinates from a reference frame of the scene video ("image space") to a reference frame of the iPad screen ("document space"), depicted in Figure 5-4.

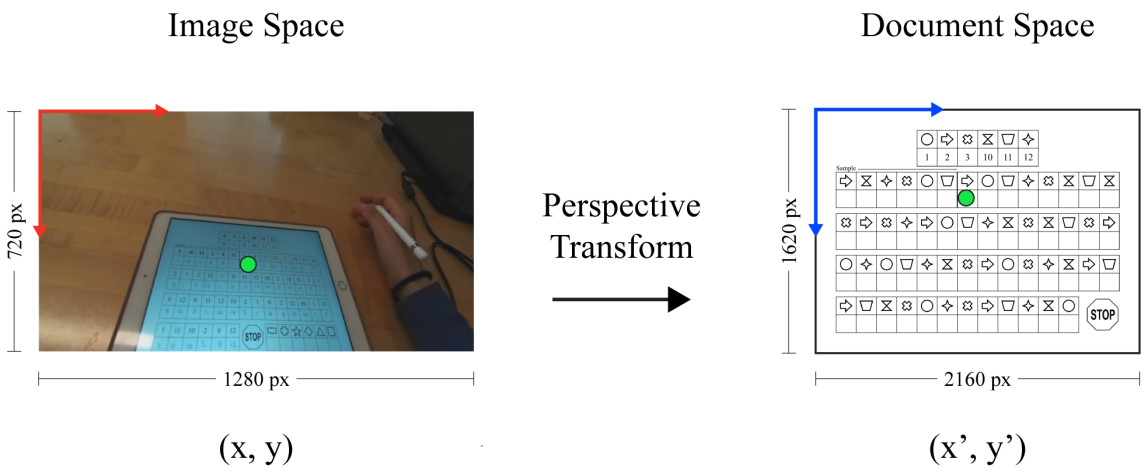

Figure 5-4: Desired reference frame transformation from image space (depicted in red) to document space (depicted in blue) for fixation points (depicted in green).

In previous work involving eye-tracking, this was achieved by utilizing physical fiducial markers attached to the four corners of the paper version of the dSDT [6]. This would allow for straightforward object tracking via the ETAnalysis App. Perspective geometry could then be used to create a transformation matrix that maps points from image space to document space.

However, this method will not work for the purposes of this data processing pipeline because we want to be able to handle participant self-administration of the dSDT with the Cognitive Health App, which was designed to be taken in natural and

flexible settings. Moreover, using fiducial markers requires all corners remain visible. This is almost never the case in a natural setting. Additionally, the fiducial markers are large and would likely be distracting to participants using the iPad.

Instead, this work explores using specific substructures in the dSDT document itself as natural fiducial markers. These substructures (e.g. the symbol-digit key, the first cell in the sample section, etc.) tend to remain visible during the relevant portions of the *scene video* and are tracked as MAOI with the ETAnalysis App as discussed in section 5.1.2.

For each frame with fixation coordinates from the *eye-tracking data file* and MAOI coordinates from the *MAOI file*, three transformation matrices are computed: M_1 (transforms image space to MAOI space), M_2 (transforms MAOI space to an intermediate space), M_3 (transforms the intermediate space to document space). When each matrix is multiplied in sequence, the result is a single transformation matrix that maps coordinates from image space to document space. This sequence can be seen in Figure 5-5.

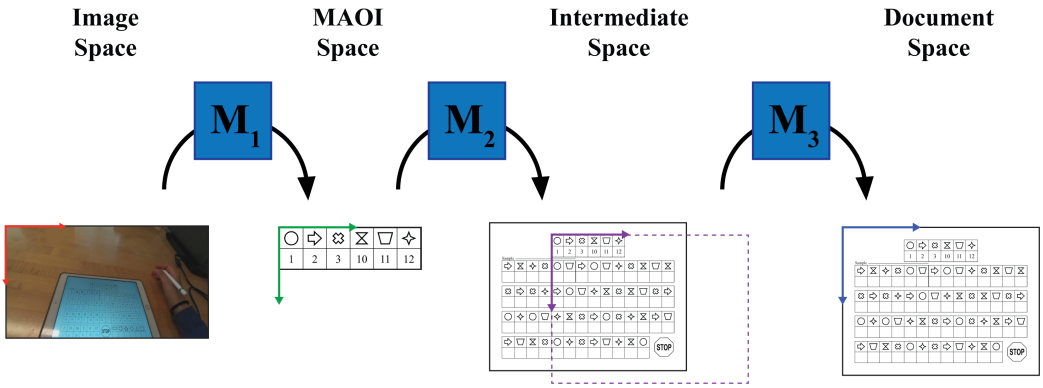

Figure 5-5: The sequence of transformation matrices to map from image space to document space, using the symbol-digit key as the MAOI.

5.2.3 Categorization to dSDT Cell

To categorize each transformed fixation coordinate to a dSDT cell, we need the positions of bounding boxes for each cell in the dSDT. Each cell is assigned a unique name for identification. These cell assignments can be seen in Figure 5-6 for the translation

task.

The categorization is computed by a straightforward approach of checking all cell bounding boxes and selecting the one in which the fixation point lies, if any. The unique cell name for each gaze point and fixation point is saved as a new entries in the `doc_gaze_cell` and `doc_fix_cell` fields, respectively. The update eye-tracking data is then saved as an Excel spreadsheet known as a *trajectory file*.

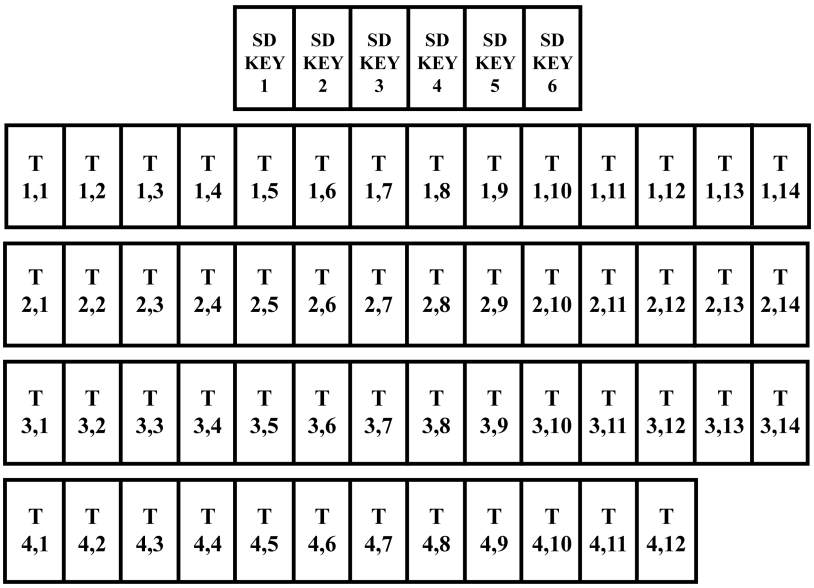

Figure 5-6: Assigned cell labels in test form for the translation task in the dSDT.

5.3 Handwriting Pipeline

This section describes the processing done to the handwriting data that will result in the categorization of each stylus position to a particular cell on the dSDT. An overview of this pipeline is depicted in Figure 5-7.

Upon a participant's completion of the dSDT via the Cognitive Health App, a timestamped record of their stylus positions is saved in the form of an *SSK file*. Each trial of the dSDT saves its associated handwriting data in a separate *SSK file*. The primary fields for each data sample in this file include the timestamp and stylus coordinates, along with additional measurements such as altitude, azimuth, and pressure.

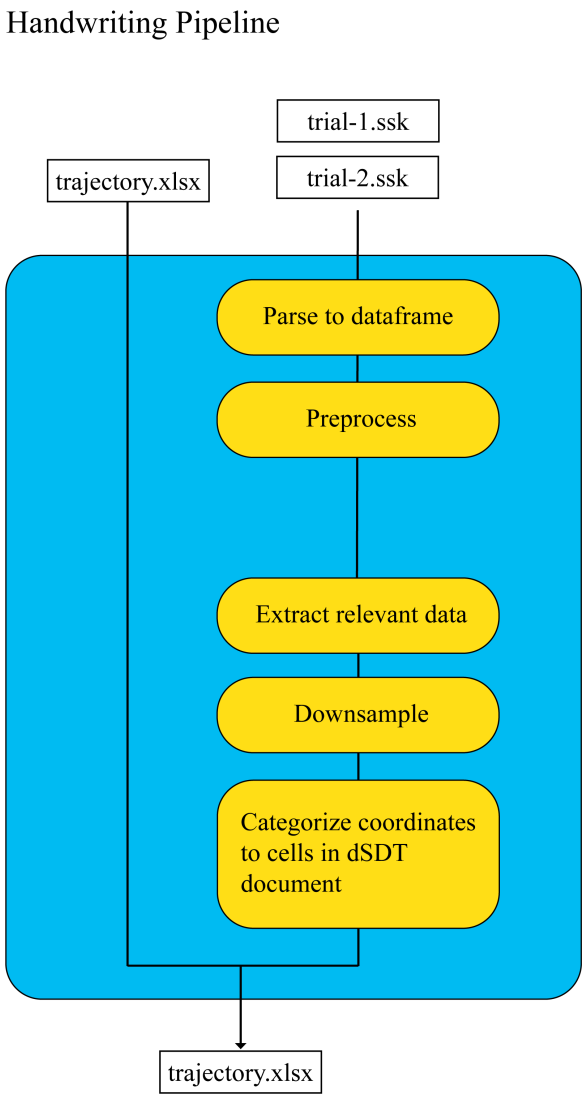

Figure 5-7: An overview of the processing done in the Handwriting Pipeline. The pipeline receives as input the *trajectory file* (trajectory.xlsx), the trial 1 *SSK file* (trial-1.ssk), the trial 2 *SSK file* (trial-2.ssk) and outputs the updated *trajectory file* (trajectory.xlsx).

5.3.1 Preprocessing

Initially, the x and y coordinates of the stylus positions are in units of points. To convert them to units of pixels within the 2160x1620px iPad screen, scaling factors from metadata in the *SSK file* are used to convert the coordinates to pixels. Moreover, since the Cognitive Health App presents the dSDT in landscape mode on the iPad (by

default x and y positions assume a portrait mode orientation) the x and y positions must be swapped for correctness. A sample of the handwriting data for a participant can be seen in Figure 5-8.

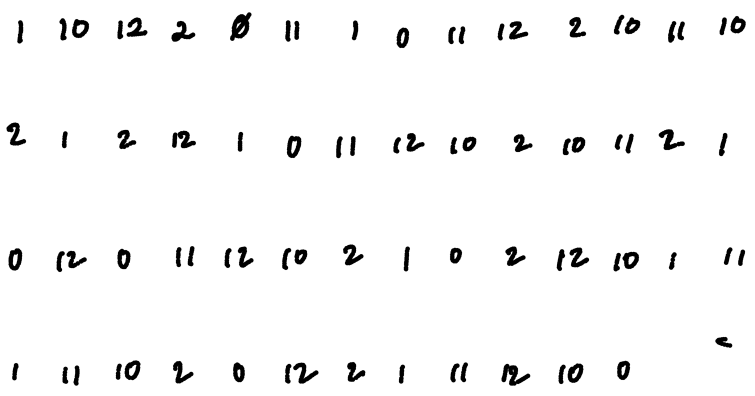

Figure 5-8: Example of a participant's stylus positions saved in an *SSK file*.

5.3.2 Segmentation

The data in the *SSK file* spans the duration of a single trial of the dSDT. There is no distinction between data samples which occur in the sample section or the main section of the translation task. Moreover, there is no distinction between data samples that occur in the copy or recall tasks. In order to distinguish between these various sections, the timestamped handwriting data must be segmented.

By plotting `pressure` vs `timestamp` data for multiple *SSK files* (see Figure 5-9), it was observed that there is an interval of time (approximately 15 seconds) with no handwriting activity from participants. This occurs between the sample and main sections and between the copy and recall sections. These are periods where the Cognitive Health App provides verbal and visual instructions on how to complete the test. During these periods, the Cognitive Health App prevents the participant from writing with the stylus. As a result, constants for these brief periods of inactivity

where empirically determined. By detecting the timestamps where this period of inactivity occurs, the data samples can be separated and categorized into their proper sections.

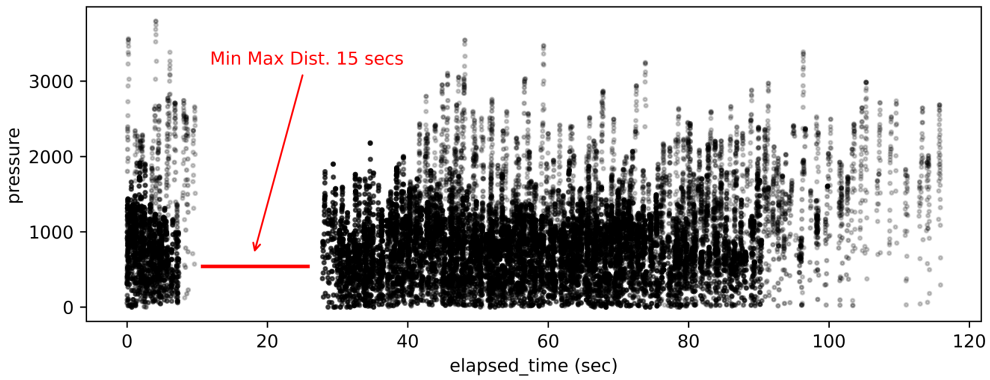

Figure 5-9: Empirical determination of threshold value for separation of sample and main sections in *SSK files*.

5.3.3 Downsample

Similar to the eye-tracking data, the handwriting data is also downsampled to the frame-level at 30 frames per second. This is done using the same method described in section 5.2.1, where only the earliest sample of the binned handwriting samples is selected for each frame.

5.3.4 Categorization to dSDT Cell

We categorize stylus positions to dSDT cells using the same technique described in section 5.2.3 for fixation points, where the cell is determined by checking all cell bounding boxes and selecting the one in which the stylus position lies.

Chapter 6

Behavior Detection

A central goal of the dSDT is to detect higher-level learning behaviors, as these behaviors are expected to be indicative of participants' learning and cognitive abilities. These behaviors may serve as components for training classifier models in future work.

The following sections describe how these behaviors are detected using the processed eye-tracking and handwriting data in the *trajectory file*.

6.1 Modality Synchronization

The processed eye-tracking data from the Vision Pipeline and the processed handwriting data from the Handwriting Pipeline both contain timestamped data. For synchronization, the data from both modalities must be downsampled to the level of frames (as described in section 5.2.1), after which they can be joined along their `frame_number` field. [1]

After synchronizing the eye-tracking and handwriting data, we are left with a single sequence of frame data containing fixation and stylus positions categorized to cells in the dSDT document. This resulting data is saved as the *trajectory file*, whose relevant fields are listed in Table 6.1.

[1] Due to a now resolved bug in the Cognitive Health App, the recorded timestamps in the SSK files for the handwriting data were incorrect by a constant factor. To account for this, an empirically defined offset was added to all SSK file timestamps. This will not be necessary for future sessions with the Cognitive Health App, however all examples shown in this work will have this offset.

FIELD	DESCRIPTION
timestamp	Timestamp (as a datetime string) of when current data was collected.
frame_number	Frame number since start of ETVision recording.
is_fix	True if part of a fixation sequence.
doc_horz_gaze	Horizontal coordinate of gaze within test document, in pixels.
doc_vert_gaze	Vertical coordinate of gaze within test document, in pixels.
doc_horz_fix	Horizontal coordinate of fixation within test document, in pixels.
doc_vert_fix	Vertical coordinate of fixation within test document, in pixels.
gaze_cell	Cell within test document that contains current gaze.
fix_cell	Cell within test document that contains current fixation.
cell_position	Integer representing the position of `fix_cell` in the sequence of cells in the test document.
pen_cell	Cell within test document that contains current stylus position.

Table 6.1: Relevant fields in the *trajectory file*.

Using the `fix_cell` and `pen_cell` fields of the *trajectory file*, we now have a determination of what cells in the dSDT a participant is looking at and writing in for the given (trial, section) configuration in the test's translation task. Using this information, learning behaviors can be detected by searching for the fixation and handwriting patterns described below.

6.2 Defining Search Space Intervals

The learning behaviors defined in section 3.1 can be divided into two main types of behaviors: those that depend on the current stimulus cell (stimulus-cell behaviors) and those that do not (non-stimulus-cell behaviors). The stimulus cell is defined to be the current cell that the participant is trying to find the symbol-digit mapping for. Visual Match, Concentrated Match, Forward and Backward Fixations, and Spatial Association behaviors all depend on knowing the current stimulus cell for detection. In contrast, Scans and Back-and-Forth behaviors depend solely on the processed eye-tracking data, i.e. the `fix_cell` field of the *trajectory file*. [2] As a consequence, the detection of each of these type of behaviors are handled differently.

6.2.1 Non-Stimulus-Cell Behaviors

Since non-stimulus-cell behaviors such as Scans and Back-and-Forth are dependent only on specific patterns of a participant's fixations, detection of these behaviors consists of searching the `fix_cell` field for every frame in the *trajectory file*.

6.2.2 Stimulus-Cell Behaviors

To detect stimulus-cell behaviors, the current stimulus cell must be known for every frame in the *trajectory file*. Due to the fact that the Cognitive Health App enforces users to write in each cell before advancing, each cell in the dSDT will be a stimulus

[2] The only exception to this is the *confirmation* subtype of the Back-and-Forth behavior, which does depend on the stimulus cell.

cell for some interval of time. This interval is defined as the *stimulus cell interval* for a given cell.

The `pen_cell` field of the trajectory file is used to determine the interval $[f_{i,start}, f_{i,stop}]$ for each cell, where $f_{i,start}$ is the frame number when the participant starts writing their response for $cell_i$ and $f_{i,stop}$ is the frame number when the participant finishes writing their response. This interval is known as the *cell response interval* for a given cell.

In order to determine the start and end frame numbers defining a cell's *stimulus cell interval*, we use the assumption that when the *cell response interval* starts, the *stimulus cell interval* ends. In other words, when a participant starts writing their response to a cell, that cell's *stimulus cell interval* ends. This is because they are no longer trying to determine that cell's symbol-digit mapping.

We also use the assumption that when one cell's stimulus cell interval ends, the *stimulus cell interval* of its consecutive neighbor begins. Therefore, the *stimulus cell interval* for $cell_i$ is $[f_{i-1,start}, f_{i,start}]$. In other words, detection of stimulus-cell behaviors consists of searching the `fix_cell` field for every frame in the *trajectory file* within that cell's *stimulus cell interval*.

An example of the the *stimulus cell interval* (labeled as S_1) for $cell_4$ is depicted in Figure 6-1.

One potential issue with using the *stimulus cell interval* of a cell as the sole search space to detect stimulus-cell behaviors is that it fails to account for instances where participants are able to learn multiple symbol-digit mappings at once. Take the example of detecting a Visual Match behavior. After a participant has written their response for $cell_1$ and is looking for the symbol-digit mapping for $cell_2$, they may look at the corresponding key cells for both $cell_2$ and $cell_3$. Then when they write their response for $cell_2$, they immediately write their response for $cell_3$ without looking at another cell. If we search only within the *stimulus cell interval*, then there would be no visual match detected for $cell_3$, even though there was one.

In order to detect these earlier behaviors, we define a second interval for $cell_i$ known as the *long-term stimulus interval* $= [f_0, f_{i-1,start} - 1]$ that starts from the

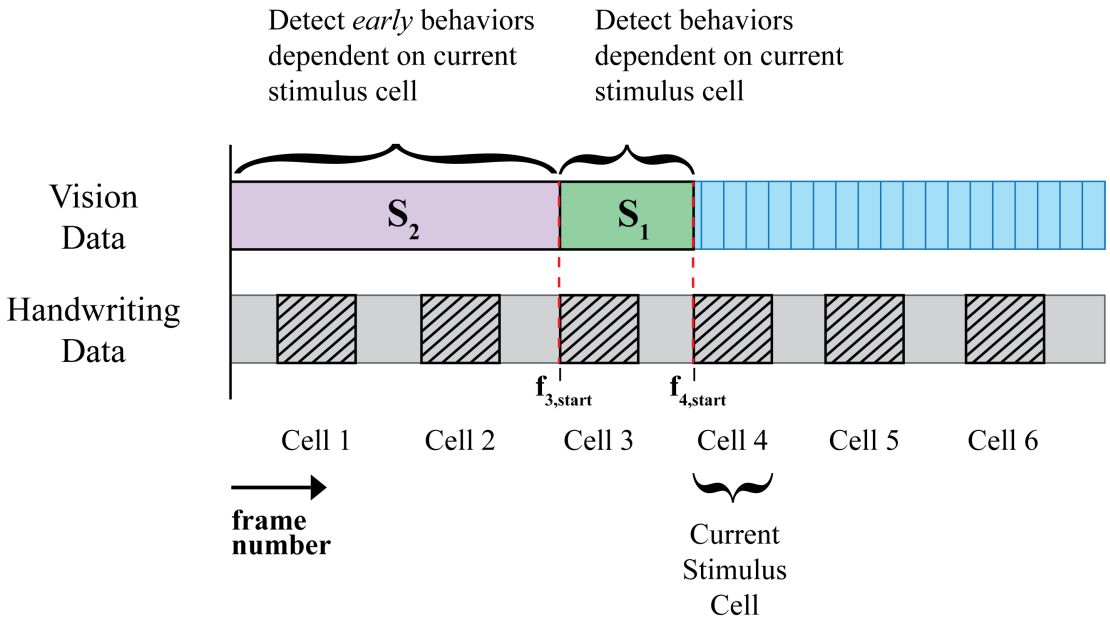

Figure 6-1: Definition of *stimulus cell interval* S_1 and *long-term stimulus cell interval* S_2 to search for behaviors with *Cell 4* as the current stimulus cell. Shaded intervals in the handwriting data represent frames with stylus positions in the labeled cell.

first frame number in the trajectory file and stops at the frame number before the cell's *stimulus cell interval* starts. Any behaviors detected in the *long-term stimulus interval* are prefixed by "*early-*". For example, *early-key-match* or *early-confirmation.*

An example of the the *long-term stimulus interval* (labeled as S_2) for $cell_4$ is depicted in Figure 6-1.

6.2.3 Trajectory Data Preprocessing

The *trajectory file* has dSDT cell locations for fixations and handwriting stylus positions for every frame in its particular (trial, section) configuration in the translation task. The detection of learning behaviors depends on the sequence of individual cells located in the `fix_cell` field. Whether a participant fixates on a particular cell for one frame or five frames is not important for behavior detection. As a result, we can simplify the detection process by transforming the *trajectory file* such that samples from adjacent frames with the same value in the `fix_cell` field are aggregated (Figure 6-2).

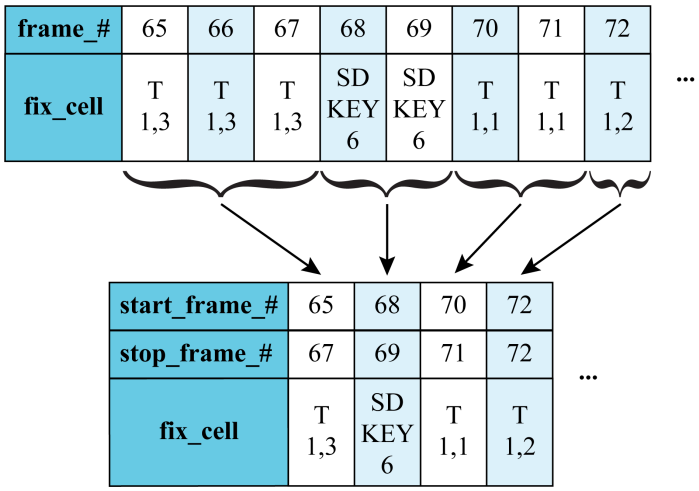

Figure 6-2: The process by which adjacent rows of a participant's trajectory data with fixations in the same cell are aggregated for a simplified view of the data.

6.3 Detecting Non-Stimulus-Cell Behaviors

The following subsections describe the detection of non-stimulus-cell behaviors by searching all frames in the *trajectory file*.

6.3.1 Scans

Scan behaviors occur when a participant has consecutive fixations across three or more adjacent cells in a single direction. Its various subtypes are dependent on whether the scan occurred in a rightward or leftward direction and whether it occurred within the key or not.

Each cell in the symbol-digit and digit-digit test forms are given an ordering from one to the total number of cells in the form, increasing from left to right and top to bottom. Each cell in the `fix_cell` field can be mapped to an integer value representing its place in the ordering, which is stored in the `cell_position` field.

Detecting rightward scans involves looking for groups of three or more adjacent *trajectory file* entries with consecutively increasing `cell_position` values. If all the cells in this group are key cells, it is a *right-key-scan*, otherwise it is a *right-same-row-scan*.

Similarly, detecting leftward scans involves looking for groups of three or more

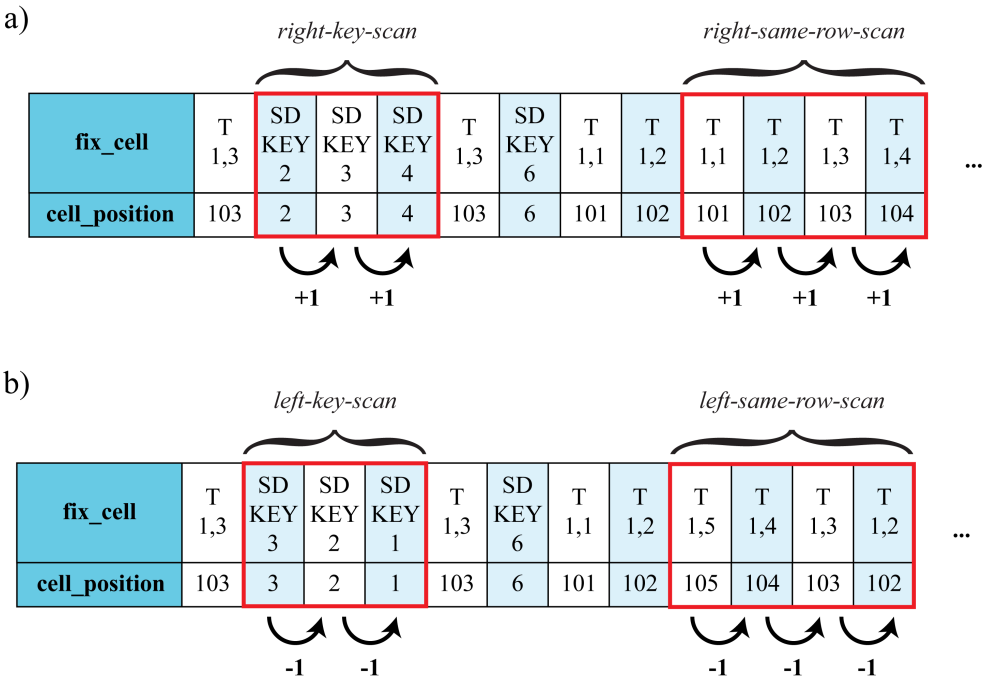

Figure 6-3: Detection of a) rightward scans and b) leftward scans located in a segment of the *trajectory file* with only the `fix_cell` and `cell_position` fields shown.

adjacent rows with consecutively decreasing `cell_position` values. If all the cells in this group are key cells, it is a *left-key-scan*, otherwise it is a *left-same-row-scan*.

The occurrences of the different variants of this behavior that can be found is illustrated in Figure 6-3.

6.3.2 Back-and-Forth

Back-and-Forth behaviors occur when a participant fixates on one cell, then another cell, and then back to the first cell. Its various subtypes are dependent on whether the cells involved are key cells or test cells.

Since Back-and-Forth behaviors involve sequences of three consecutive fixations, a sliding window of length three is used to inspect the `fix_cell` field of the *trajectory file*. For every triplet of fixated cells (x y, z) in the window, a Back-and-Forth behavior has occurred if x is the same cell as z.

When a Back-and-Forth behavior has been detected, its specific subtype is determined as follows: if x and y are both key cells the subtype is *key-key*, otherwise if x

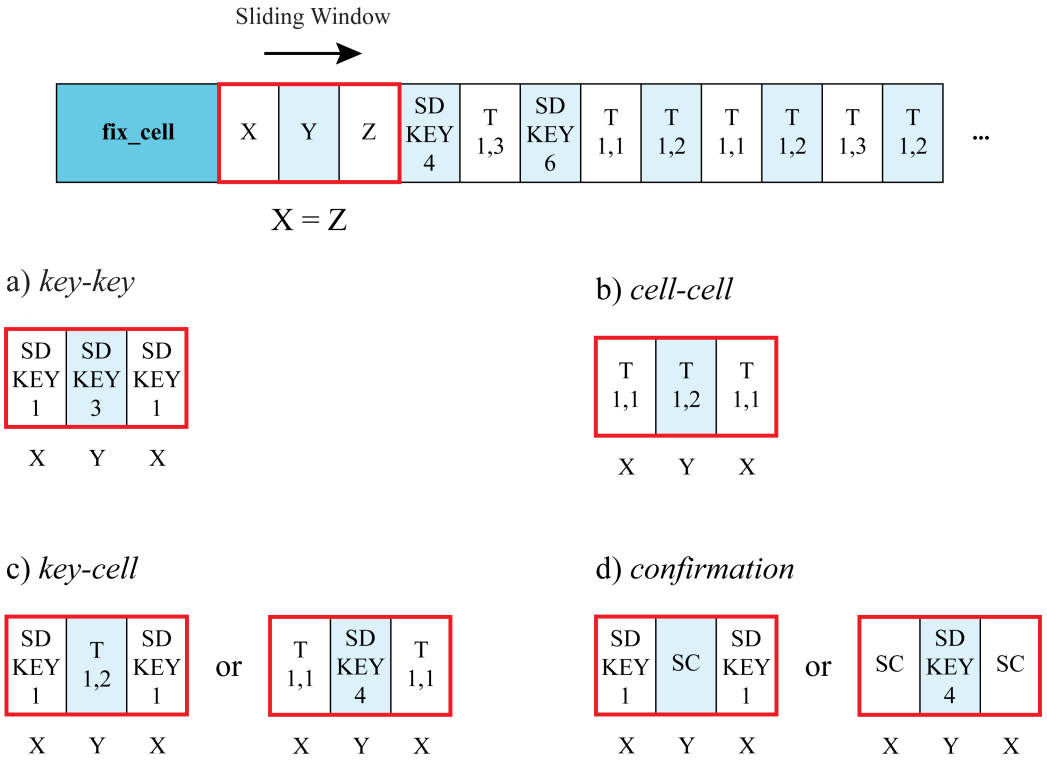

Figure 6-4: Detection of Back-and-Forth behavior subtypes: a) *key-key*, b) *cell-cell*, c) *key-cell*, and d) *confirmation* located in a segment of the *trajectory file* with only the `fix_cell` field shown. **NOTE:** SC refers to the current stimulus cell

and y are both test cells the subtype is *cell-cell*, otherwise if x and y consist of one key cell and one test cell the subtype is *key-cell*.

While the Back-And-Forth behavior is categorized as a non-stimulus-cell behavior, there is one final subtype that does depend on the current stimulus cell: the *confirmation* subtype. This subtype is equivalent to the *key-cell* subtype with the additional constraint that the test cell involved is the current stimulus cell. This subtype in particular is only detected in the *stimulus cell intervals*, as defined in section 6.2.2, of the trajectory data.

The occurrences of the different variants of this behavior that can be found is illustrated in Figure 6-4.

6.4 Detecting Stimulus-Cell Behaviors

Similar to when detecting non-stimulus-cell behaviors, the data in the *trajectory file* is first transformed such that adjacent rows with the same value in their `fix_cell` field are aggregated. This is done to simplify behavior detection. Detection of stimulus-cell behaviors involves searching only the frames within the *stimulus cell interval* for every stimulus cell in the *trajectory file*. To detect potential early occurrences of these behaviors, additional searching is done in the *long-term stimulus cell intervals* for each cell.

6.4.1 Visual & Concentrated Matches

The process for detecting Visual Matches and Concentrated Matches are very similar and so will be discussed together (in fact, Concentrated Matches can be seen as a special-case of Visual Matches).

A Visual Match occurs when a participant fixates on a cell that has the same symbol as the current stimulus cell. Likewise, a Concentrated Match occurs when the *first* fixated cell has the same symbol as the stimulus cell, i.e. is a visual match.

For each stimulus cell, its associated symbol in the dSDT is used to filter out all cells in the `fix_cell` field of the *trajectory file* that do not share the same symbol. Any fixations in the stimulus cell itself are also filtered out. The remaining cells are therefore considered Visual Matches, i.e. cells that have the same symbol as the stimulus cell.

For each of these matches, their specific subtype is determined according to the following rules: if the matching cell is a key cell then it is a *key-match*, otherwise if the matching cell is a test cell it is a *cell-match*.

A Concentrated Match occurs when the very first fixated cell in a stimulus cell's *stimulus cell interval* is a Visual Match. It's subtype is either *concentrated-key-match* or *concentrated-cell-match*, depending on the subtype of the Visual Match in question.

The occurrences of the different variants of this behavior that can be found is illustrated in Figure 6-5.

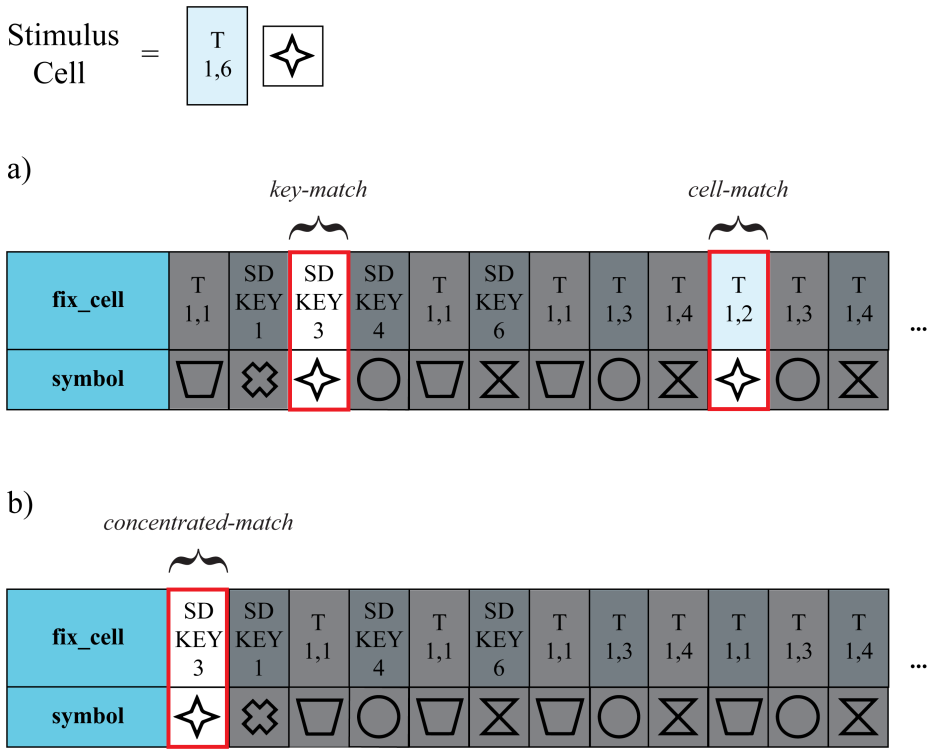

Figure 6-5: Detection of a) Visual Matches and b) Concentrated Matches located in a segment of cell T_1_6's *stimulus cell interval* in the *trajectory file* with only the `fix_cell` and `symbol` fields shown. **NOTE:** Grayed out cells represent cells that have been filtered out based on their symbol.

6.4.2 Forward and Backward Fixations

A Forward Fixation occurs when the participant fixates on a cell in the same row and to the right of the current stimulus cell, while a Backward Fixation occurs when the participant fixates in a cell in the same row and to the left of the current stimulus cell.

For each stimulus cell, we use its row and column number to create a list of all the cells in the same row as the stimulus cell. This list of cells can be separated into two sets: the set of cells that occur to the left of the current stimulus cell, and those that occur to the right of it. These are referred to as Backward Cells and Forward Cells, respectively.

Upon iterating over each fixated cell in the `fix_cell` field of the *trajectory file*, if a cell is in the set of Forward Cells, it is a Forward Fixation. Likewise, if a cell is in the set of Backward Cells, it is a Backward Fixation.

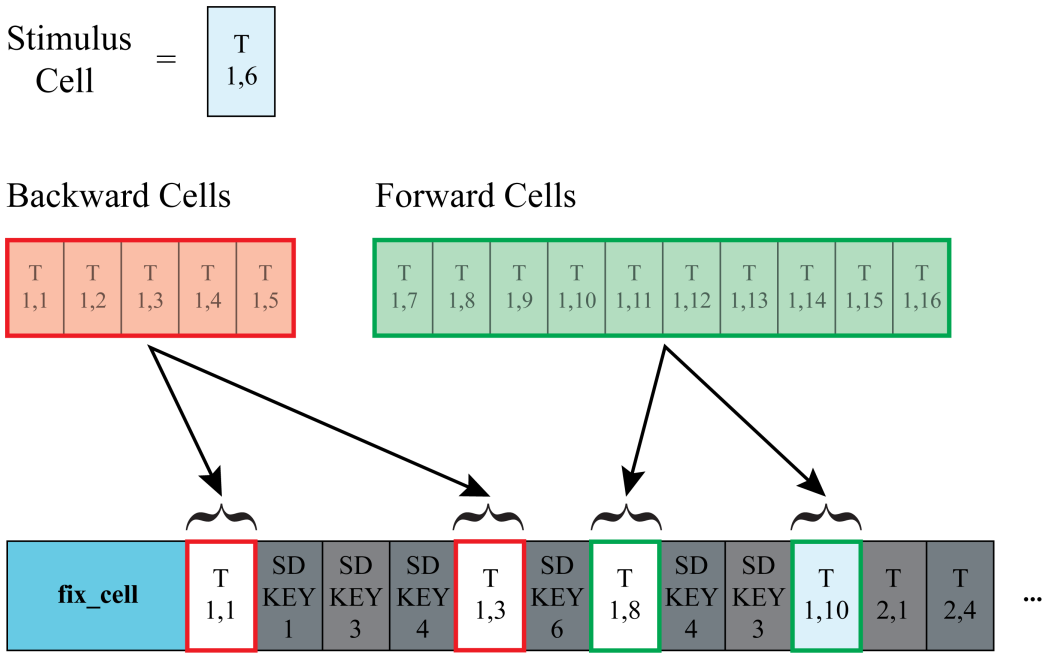

Figure 6-6: Detection of Forward and Backward Fixations located in a segment of cell T_1_6's *stimulus cell interval* in the *trajectory file* with only the `fix_cell` field shown. **NOTE:** Grayed out cells represent cells that do not lie on the same row as the stimulus cell.

The different variants of this behavior is illustrated in Figure 6-6.

6.4.3 Spatial Association

Each row of cells in the dSDT, along with the key, can be divided into three spatial sections: START, MIDDLE, and END. Each section corresponds to roughly one third of the cells that make up the row. These spatial sections are identified in Figure 6-7 for a single cell row and the key.

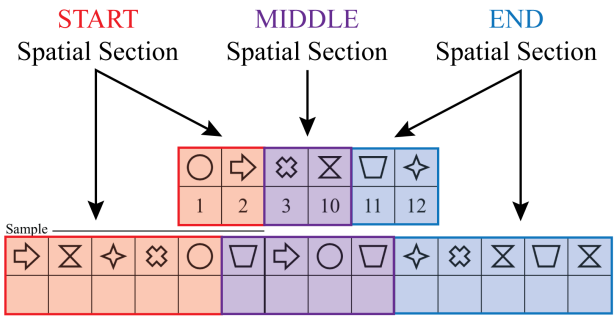

Figure 6-7: START, MIDDLE, END spatial sections for key and test cell rows.

A Spatial Association behavior occurs when the participant's first fixation in the *stimulus cell interval* lies in a cell that belongs to the same spatial section as the stimulus cell.

A list of all the cells belonging to each spatial section across the dSDT are computed and defined as START Cells, MIDDLE Cells, and END Cells. To map any particular cell to its spatial section, it's simply a matter of determining what list the cell is contained in.

Each stimulus cell's spatial section is determined, along with the spatial section of the first fixated cell in their corresponding *stimulus cell interval*.

The occurrence of one of the variants of this behavior is illustrated in Figure 6-8.

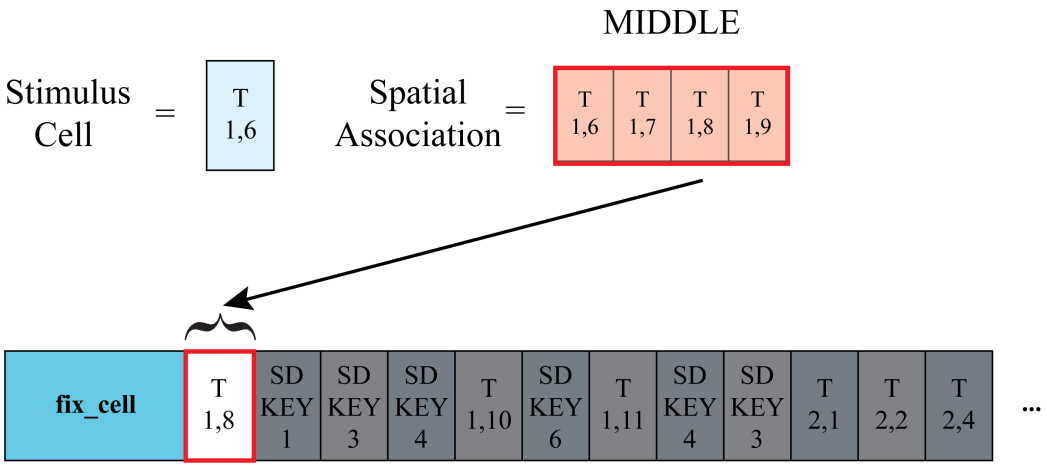

Figure 6-8: Detection of a MIDDLE Spatial Section located in a segment of cell T_1_6's *stimulus cell interval* in the *trajectory file* with only the `fix_cell` field shown. **NOTE:** Grayed out cells represent cells that do not lie in the same spatial section as the stimulus cell.

6.5 Visualization

Results of the behavior detection are saved to the *behaviors file* in a CSV file format. The data collected for each behavior is listed in Table 6.2.

In order to better study and analyze participants' learning behavior, we developed a visualization tool.

Using the data from the *trajectory file* and *behaviors file*, this tool allows one

to explore a simulation of a participant's test session frame by frame. The detected learning behaviors can be overlaid on the dSDT document along with the current gaze or fixation point, and stimulus cell. Figure 6-9 shows a screenshot of the visualization tool.

FIELD	DESCRIPTION
behavior	Name of the detected behavior.
subtype	Subtype of the detected behavior.
stimulus_cell	Current stimulus cell of the detected behavior.
start_frame_number	`frame_number` field value of the first sample of the detected behavior.
stop_frame_number	`frame_number` field value of the last sample of the detected behavior.
cells_involved	Sequence of cells involved in detected behavior.

Table 6.2: Fields in the *behaviors file*.

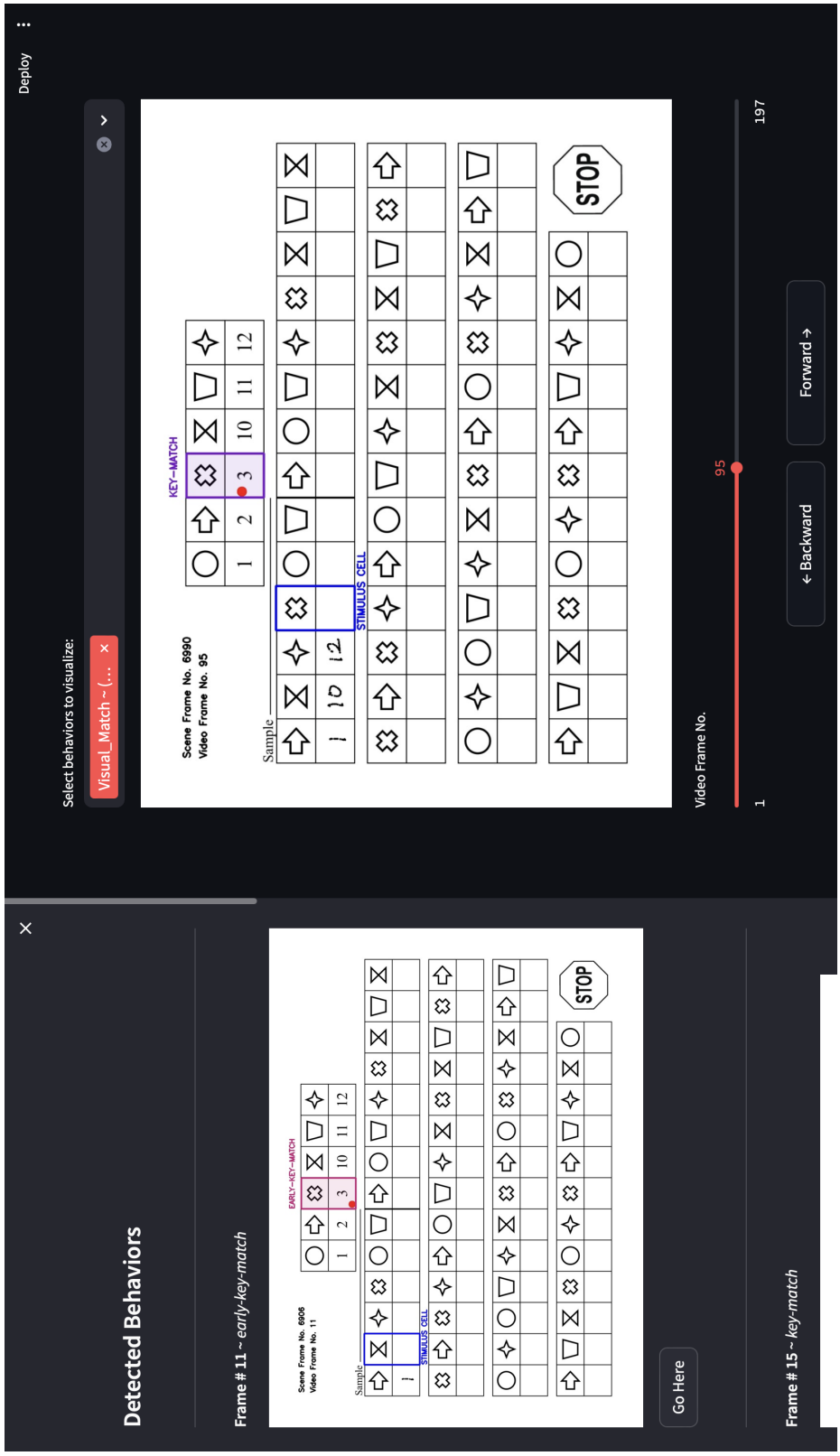

Figure 6-9: Screenshot of UI to visualize *trajectory file* and any detected behaviors.

Chapter 7

Conclusion

This work has made useful contributions in facilitating the collection and analysis of mutimodal data suggestive of human cognitive behavior.

7.1 Contributions

We developed a data processing pipeline that ingests eye-tracking data captured by Argus Science's ETVision system in order to categorize point of gaze and fixation coordinates into cells on the dSDT (ETAnalysis Pipeline, Vision Pipeline and Handwriting Pipeline). Moreover, this work develops a behavior detector (Behavior Detector Pipeline) that acts on synchronized eye-tracking and handwriting dSDT cell sequences to detect and visualize specific learning behaviors that were defined in previous work. Finally, this work provides data on detected behaviors from a subset of the healthy participants tested.

7.2 Limitations

Managing the trade-off between eye-tracking quality and the participant test-taking experience played a central role in the challenges encountered in this work. The ideal use case of eye-tracking systems such as the ETVision system rely on users looking straight ahead with a horizontal line-of-sight and the avoidance of harsh

angles. Examples may include actions such as driving a car or staring at a computer monitor.

When participants are instructed to complete the dSDT on the Cognitive Health App on an iPad, these ideal conditions are rarely met. Typically, participants let the iPad rest on the table and hunch over to look down as they complete the test. While it would be possible to instruct participants to restrict their motion in order to improve eye-tracking quality, since the dSDT involves writing with the stylus, it would likely make the test-taking process awkward and more difficult.

Another pain point where this trade-off becomes apparent lies in the need for MAOI tracking. Assuming a scenario where the eye-tracking quality is perfect, fiducial markers must still be used in order to map the point of gaze coordinates from image space to document space. While physical fiducial marker attachments to the iPad would improve MAOI tracking quality (a method previous work has used), one of the goals of this work was to explore a method that could circumvent this on the grounds of physical fiducial markers being too distracting, requiring additional setup, and forcing participants to restrict their movements in order to ensure the markers stay in the field of view of the ETVision scene camera. A major concern is that these additional distractions and constraints on participants may affect their concentration (particularly more for those with neurodegenerative disease). To prevent this, visual structures in the dSDT were used as fiducial markers, however this resulted in the increased reliance of object tracking and the introduction of some manual data processing.

7.3 Future Work

Future work should involve gathering additional data from healthy participants in order to increase the robustness of eye-tracking with the ETVision system and detect additional learning behaviors. This work proposes some recommendations. Keep ETVision eye-tracking recordings short seems to be a good idea. In the eye-tracking recordings used in this work, the dSDT was the second cognitive test completed by

participants. The first cognitive test, which involved the rotation of the iPad, may have played a part in the deterioration of the eye-tracking quality. Future work should experiment with only recording the dSDT or having separate recordings for each trial of the dSDT, recalibrating before each. Additionally, since requiring participants to wear an eye-tracking headset (which is already potentially distracting), for the long-term, it may be useful to eventually explore other eye-tracking systems. Particularly, those that use the iPad itself, as this could eliminate the need for fiducial markers.

Ultimately, being able to collect multimodal data from both classes of participants (those with neurodegenerative disease and healthy controls) would be necessary to examine the differences in the counts and distributions of learning behaviors. Metrics such as behavior counts, handwriting speed, duration of gaze in key vs non-key cells, etc. may serve as useful components for the larger goal of defining features to be used in the development of a classification model.

This work provides software specific to the ETVision system for processing eye-tracking data, as well as more general software for detecting and visualizing learning behaviors from multimodal trajectory data. It is our hope that these tools and recommendations will play a role in deepening our understanding of participants' learning abilities in the dSDT and support the development of models for the early-detection of neurodegenerative disease.

Bibliography

[1] Maria Ascanio Aliño. Testing for subtle cognitive impairments in a clinically informed ipad platform. Master's thesis, MIT, 2023.

[2] Jack Cook. An effective platform for assessing cognitive health. Master's thesis, MIT, 2023.

[3] Lauren Huang. The digital symbol digit test: Screening for alzheimer's and parkinson's. Master's thesis, MIT, 2017.

[4] World Health Organization. *Global status report on the public health response to dementia*. World Health Organization, 2021.

[5] Oyama A, Takeda S, Ito Y, Nakajima T, Takami Y, Takeya Y, Yamamoto K, Sugimoto K, Shimizu H, Shimamura M, Katayama T, Rakugi H, Morishita R. Novel method for rapid assessment of cognitive impairment using high-performance eye-tracking technology. *Sci Rep.*, 9(1), September 2019.

[6] Sarbari Sarkar. Gaze-tracking analysis for cognitive screening and assessment. Master's thesis, MIT, 2020.

[7] Tokushige Si, Matsumoto H, Matsuda Si, Inomata-Terada S, Kotsuki N, Hamada M, Tsuji S, Ugawa Y and Terao Y. Early detection of cognitive decline in alzheimer's disease using eye tracking. *Front. Aging Neurosci.*, 15, March 2023.