King's College London

Institute of Psychiatry, Psychology & Neuroscience (IoPPN)

MSc Neuroimaging – Module 5: Research Dissertation

Eyes Wide Open: A Behavioural and Computational Study of MNIST Digit Identification

Submitted by: 1887976

Date of submission: 30th of August, 2019

Word count: 10'896

Number of figures: 28

Acknowledgments

Throughout this research project, I received unreserved support and help from Dr. Rosalyn Moran. Despite being one of the busiest scientists I have ever had the privilege to meet and work with, she has never failed to make me feel like a top priority in her laboratory. She is without a doubt one of my greatest inspirations and a person I tremendously look forward to collaborating with in my future career.

None of my project would have been possible without the exemplary mentoring of Dr. Berk Mirza, whose extensive knowledge about eye-tracking and MDP models helped me grow evermore passionate about my project and the results obtained. I will also strive to meet the impressive expertise Mr. Maell Cullen has with regards to computational neuroscience. His guidance and valuable input helped shape my work throughout this project, and his knowledge never ceases to humble me.

Finally, I would like to extend my gratitude to my fellow researcher, and colleague, Mr. Michael Miller. I can confidently assert that my understanding of this project would not equal its current state without the countless hours of collaborative work. His constant energy and positivity were essential in this project succeeding.

Abstract

Context: Current deep convolution neural networks treat vision as a static sequential readout of pixel information. Active inference defines an agent's selection of policies (set of actions) as belief based. In sampling the environment, these beliefs are generatively update in order to minimize surprise.

Objectives: We hypothesize human vision to be based upon the active inference framework as we actively seek out salient visual features. Therefore, we infer human vision to be a top-down process based on our current beliefs about the environment.

Methods: A visual eye-tracking study was performed on healthy participants to analyse gaze scan paths in trying to identify a scene. The scenes contained partially obstructed handwritten digits from the MNIST database. To further test our hypothesis, we are developing a Markov Decision Process (MDP) computational model based on the active inference framework. The MNIST database was selected as this platform is a benchmark for computer vision algorithm testing. Behavioural data analysis was performed using a novel method we developed, taking inspiration from fMRI data analysis, whereby scan paths were converted to volumetric NIfTI format for SPM analysis.

Results: In all tasks, participants showed better performance in trial duration and less scene exploration when prior beliefs created an expected scene identity.

Conclusions: Our results support our hypothesis that human vision is a predictive top-down process, based on our prior beliefs, as per active inference. Results from our MDP model show how an agent learns to look for salient features for scene identification but is still in development.

Table of Content

Introduction

Free Energy & Active Inference

In the hypothetical scenario of receiving the unfortunate news of having tested positive for a neurodegenerative disorder that affects 0.5% of the population, with a testing accuracy of 99%, one would be correct to ask for a second test to be performed. Although the reasoning for this is most likely due to wishful thinking, statistically there is only a 33.2% of actually having the disease if the result relies solely on a single test (see Figure 1).

This probability calculation is known as Bayes' Theorem (Cox, 1946; Joyce, 2019). Previous research has found evidence through various functional neuroimaging modalities that connectivity in the brain may be modeled as a Bayesian inference engine, leading to the term "Bayesian Brain" (Friston, 2010; Knill & Pouget, 2004; Körding & Wolpert, 2004). The principle of Free Energy is founded on the premise of an agent's behaviour being modeled using Bayesian statistics, and attempts to explain all living behaviour as an underlying drive to

$P(o)$ = the probability of an outcome (eg. actually having the disease = 0.5%)

$P(e)$ = the probability of an event (eg. testing positive)

$P(e, o) = P(o, e)$ = the joint probability of the event and outcome

$P(e \mid o)$ = the probability of an observation, given an event
(eg. testing positive if you actually have the disease = 99%)

$P(e) = P(e, o) + P(e, \bar{o})$ = the sum of the joint probability of an event and an observation happening & the joint probability of an event and that same observation NOT happening

$P(e, o) = P(o, e) = P(o \mid e) \times P(e) = P(e \mid o) \times P(o)$

<u>Bayes' Theorem:</u>

$$P(o \mid e) = \frac{P(e \mid o) \times P(o)}{P(e)} = \frac{P(e \mid o) \times P(o)}{P(e, o) + P(e, \bar{o})} = \frac{P(e \mid o) \times P(o)}{P(e \mid o) \times P(o) + P(e \mid \bar{o}) \times P(\bar{o})}$$

<u>In this example:</u>

$$P(o \mid e) = \frac{(0.99) \times (0.005)}{(0.99) \times (0.005) + (0.01) \times (0.995)} = \underline{33.2\ \%}$$

<u>Figure 1.</u> Bayes' Theorem and an example of a probability outcome given an event.

reduce the surprise about an agent's environment. A Bayesian brain continuously updates its beliefs about its environment based on probabilistic perceptions created through sensory interactions with the environment (Friston, 2010, 2011).

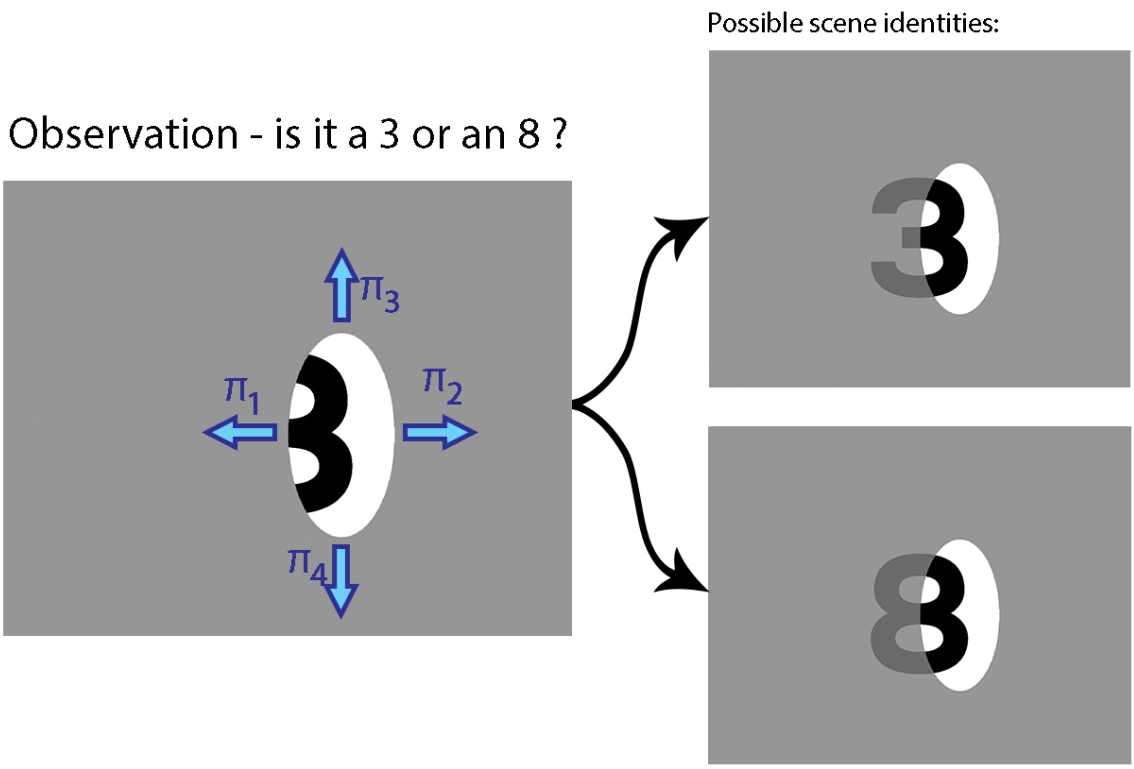
By sampling our environment through our senses, we are able to create a perception of our surroundings. This is only a tangible representation, as we may never fully access the reality of our environment – (Friston, 2011)we are effectively separated from its true state by what Karl Friston refers to as a Markov Blanket (Friston, 2013). By minimizing free energy, the agent acquires information about the true aspects of the world that give rise to the outcomes (perceptions). In other words, the more we update beliefs about our environment (by sampling it), the less we may be surprised while interacting with it. The free energy principle states this as being any living agent's greatest predilection (Friston, 2010, 2013; Mirza *et al*., 2018).

The active inference framework evaluates policies (i.e. a sequence of actions) in terms of the amount of information and utility expected under all policies (Friston *et al*., 2016; Mirza *et al.*, 2016; Pezzulo, *et al.*, 2015). The agent's beliefs about the hidden states is actively constructed through sequential observations, by engaging with its environment (Friston *et al*., 2016). Active inference, as an approach to the free energy principle, inferes that our behaviours are a direct measure of our prior beliefs about our environment. Given the context of this research project, Figure 2 gives a visual example of policy selection. In this visual example (see Figure 2), by considering different locations as hidden states, as well as the contents of these locations, an agent may choose to sample one location over another given the value of the information it believes to obtain. This observation should reduce uncertainty as much as possible (Behrens *et al.*, 2007; Mirza *et al*., 2018; Yu & Dayan, 2005).

In this example, our environment is a scene which contains a digit hidden by a mask containing a foveation through which we may perceive part of the underlying digit.

The current hidden state is the observation below.
By selecting a policy (π, inthis case one of four actions - look left; look right; look up; look down) we may change to a new hidden state through which an observation is made.

The preference for one policy over another is dependant on the expected value of the information it will allow us to retrieve (i.e. reduce uncertainty).



$\pi^* = \{\pi_1, \pi_2, \pi_3, \ldots, \pi_n\}$ = a set of single-action policies

In this example, the first policiy (π_1) is the one that would reveal the most valuable information about this scene's identity and reduces the most our uncertainty.

Figure 2. Policy decision under the active inference framework for free energy minimisation.

Here selecting the "move left" policy, an agent behaves according to its prior belief of the information this hidden states holds, and how much it will allow to resolve surprise.

Markov Decision Processes (MDP)

A Markov Decision Process (MDP) model treats states and outcomes as discrete units where hidden states generate outcomes at each timestep. In other terms, an MDP may be considered a process through which an agent samples its environment, and an observation it makes represents a reward obtained. The value of this reward, or the value of the information obtained in the case of active inference, is what causes a reinforcement in policy decision-making. This means that the reward will define the next action we take to sample our environment, based on a constructed belief we make about the state being sampled. Each policy is linked to its transition matrix (B), which is a probability distribution matrix, such that a current state is mapped to a state in the next timestep. The likelihood matrix (A), gives a probability distribution of an outcome being observed for specific hidden states (Bellman, 1957; Mirza *et al.*, 2016, 2018). These concepts are visually shown in Figure 3.

The likelihood matrix (A) in the example shown in Figure 3 maps one specific outcome to one hidden state with 100% probability. To sample its environment, our agent can look at any of the four locations, by selecting a single-action policy (π_1 = look at location 1, π_2 = look at location 2, π_3 = look at location 3, π_4 = look at location 4). The first factor in the transition matrices (B{1}), the location, is the only parameter that we may change by enacting a selected policy. This means that by selecting a policy, we change the current hidden state "location at time t" to a new hidden state (i.e. "location at time t+1"). Additionally, each policy (π_1, π_2, π_3, or π_4) maps to a single hidden state (at time t+1), regardless of the current hidden state (at time t). The hidden state "shape" (the second factor in our transition matrices, B{2}) remains the same, regardless of the policy selected; our actions cannot change the nature of the shape in a location.

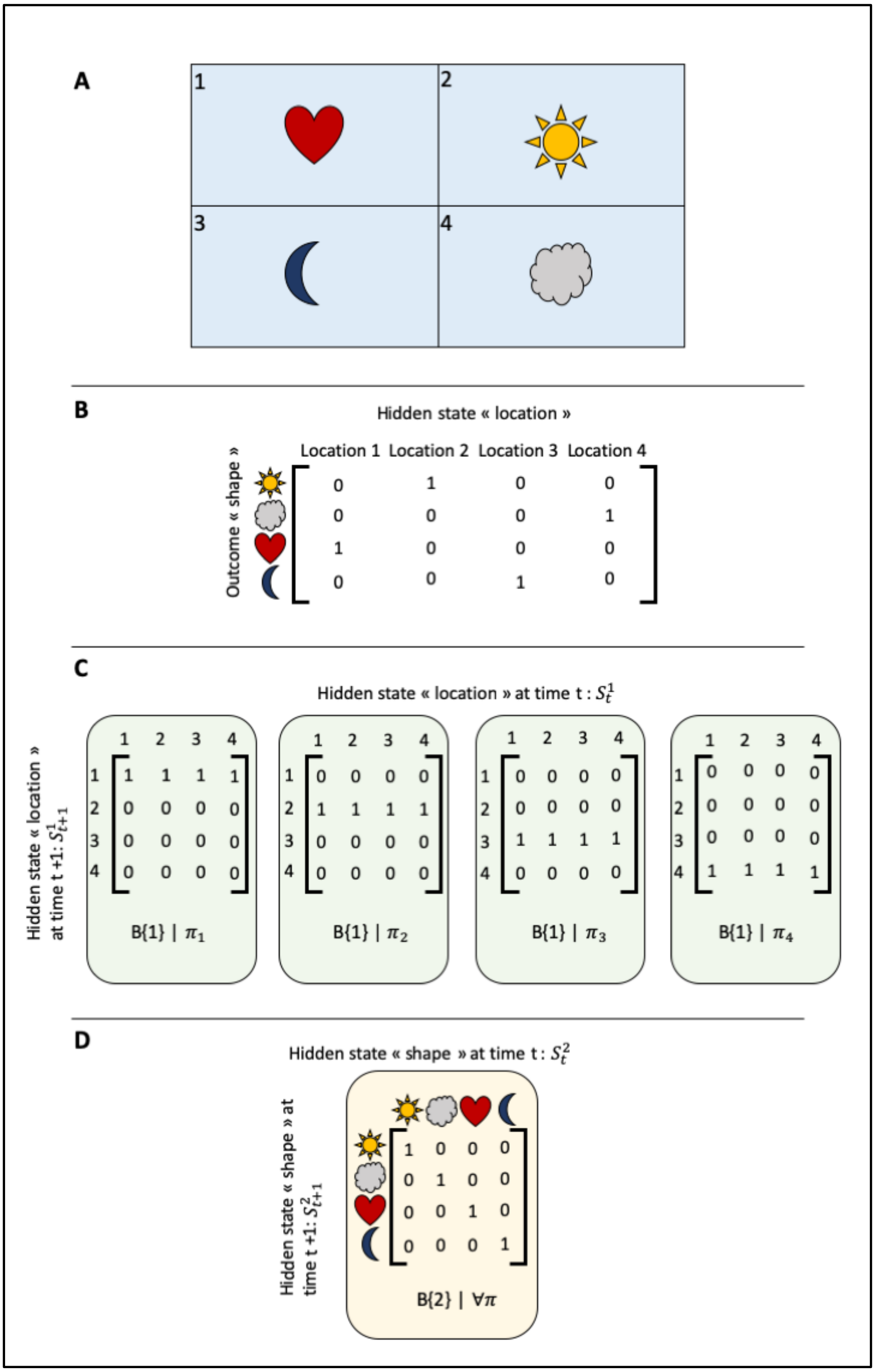


Figure 3. **A**: A simple environment with four locations, each containing a shape. **B**: The likelihood matrix of each outcome ("shape") given a hidden state ("location"). **C**: The transition matrices of the modifiable factor ("location"), given a selected policy (π_i). **D**: The transition identity matrix of the constant factor ("shape"), for any policy ($\forall\pi$).

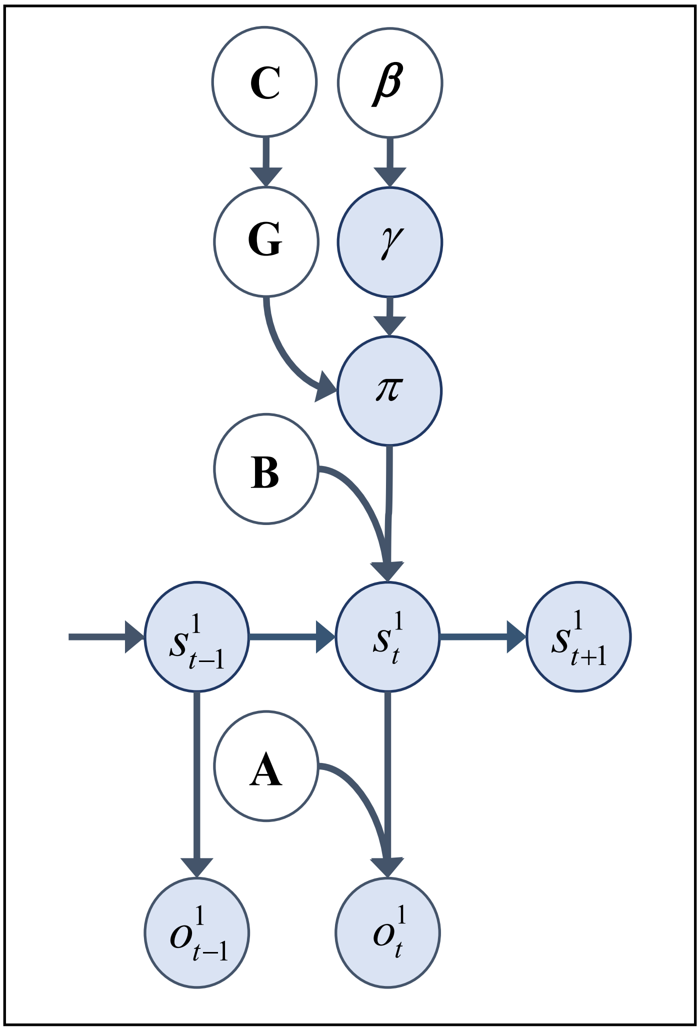


Figure 4. Illustration of transient components in the MDP considered under active inference framework.

The model that we considered is an active inference treatment of MDP for digit exploration and identification. The shape of the model is illustrated in Figure 4. This figure illustrates the dependencies in the MDP model (Mirza *et al.*, 2016). The hidden state in the current timestep (S_t^1) depends on the hidden state in the previous timestep (S_{t-1}^1) and the transition probabilities are encoded by the transition matrix (B). Certain hidden state transitions can be mediated by policies (denoted as π, e.g. where to look next). The outcomes at each timestep (O_t^1) depend only on the hidden states at each respective timestep (S_t^1). The probability of outcomes at each hidden state is encoded in the likelihood matrix (A). Precision about policies (denoted γ), which depends on the temperature parameter β, reflects a confidence in beliefs about policy selection. The C matrix reflects the preference over outcomes and G is the expected free energy. The MDP model we will construct for digit identification will follow this schema closely.

Visual System (ventral & dorsal streams)

In humans, the visual system is spatially segregated according to various stages of information processing. As visible light penetrates through the various layers of an eye, its lens focuses this light to form a sharp image (in a typical eye) on the retina, where the photoreceptive cells lie. Summation of photoreceptor cell activation is relayed through the optic nerve to a region in the thalamus known as the lateral geniculate nucleus (LGN). The visual fields of each eye are divided into central (nasal) and lateral (temporal). At the optic chiasm (before reaching the LGN), the nasal fiber bundles of the respective left and right optic nerves cross to their contralateral hemispheres before continuing their paths to the LGN (see Figure 5) (Posner & Petersen, 1990).

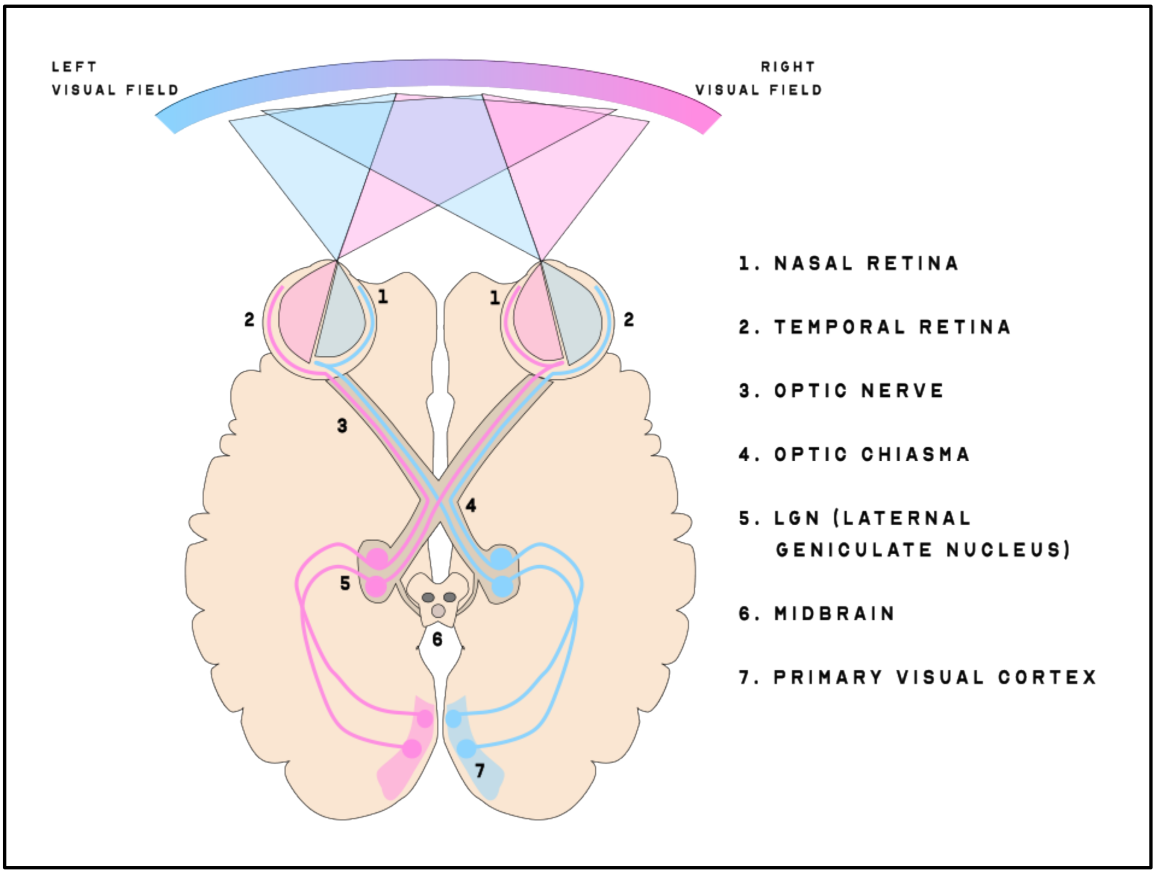


Figure 5. Axial view of the visual system[1].

[1]Inspired by: https://commons.wikimedia.org/wiki/File:Human_visual_pathway.svg

The function of this is two-fold: firstly, it allows for preliminary visual processing in the midbrain (through the LGN) to take place and encode for rapid reflexive depth-dependent (perspective-driven) behaviours. Secondly, this evolutionary feature reduces the risk of rendering an organ (i.e. an eye) to a state of complete indolence should a lesion occur along the optic nerve (Bullier, 2001; Gavornik & Bear, 2014; Posner & Petersen, 1990; Thorpe *et al.*, 1996).

From the LGN, visual stimulation is further relayed to the visual cortex (located in the occipital lobe). The primary visual cortex (V1 & V2) serves to encode basic visual constructs, such as contours, angles and contrasts. The secondary visual cortex (V3, V4 & V5) serves to reconstruct a visual perception of the observed scene by associating visual stimuli, recreated at a lower level, to familiar objects and whereabouts (see Figure 6).

In 1983, Ungerleider, Mishkin & Macko proposed a 1-brain/2-streams theory, which greatly gained in popularity since. This theory may be summarized by a dorsal cortical stream that serves to encode the location ("where") of an observed scene, and a ventral cortical stream that is devoted to determining the contents of a scene ("what") (see Figure 6) (Mishkin *et al.*, 1983). As these two streams display high functional and structural connectivity, the parsing of these two processes has garnered its fair share of criticism, although it remains a popular and useful synthetization of how visual processing allows us to navigate within our environment and to interact with objects in it (Mishkin *et al.*, 1983; Ungerleider & Haxby, 1994).

Gaze location may be simplified as being either fixating (with little to no eye movement) or saccading (rapid eye movements to a new location in the visual field for further information harvesting). The siege of these visual foraging behaviours is thought to be (in

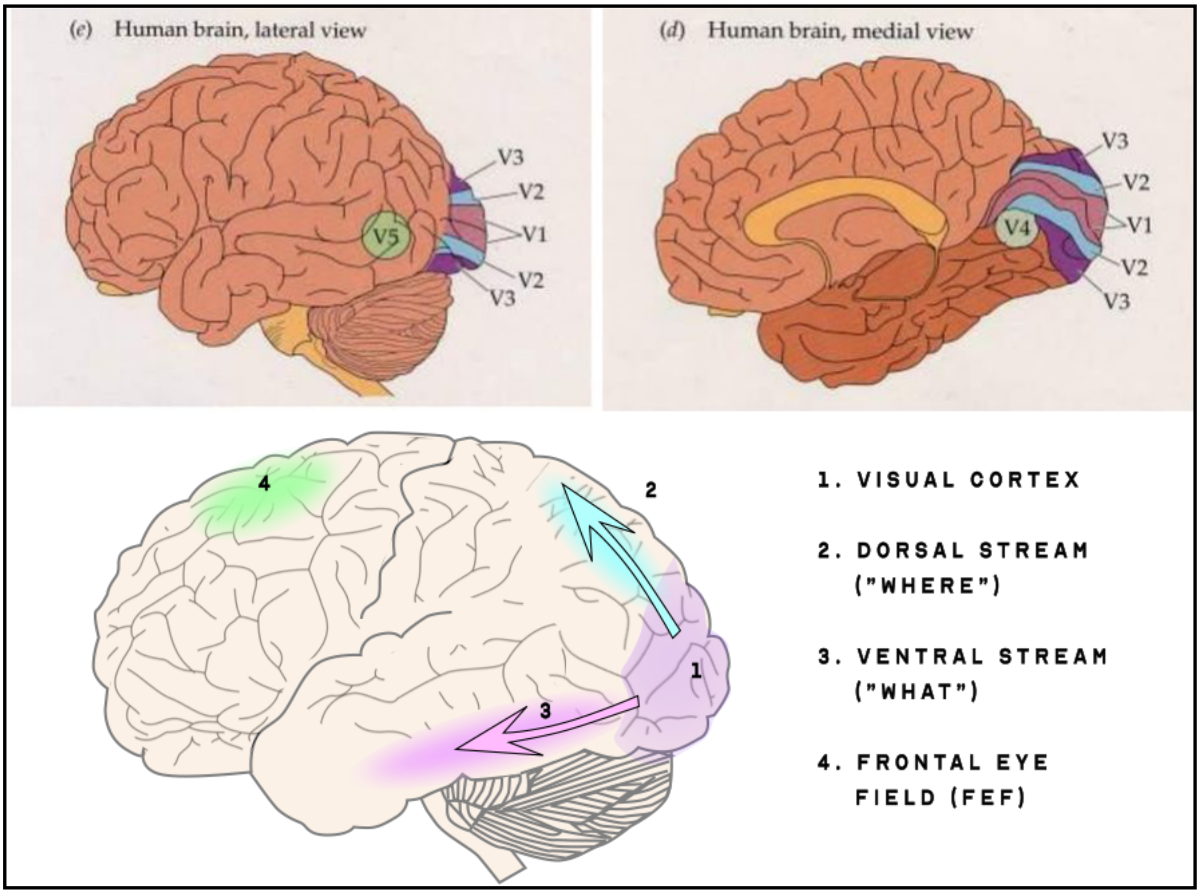


Figure 6. Top: a lateral and medial view of the visual cortex and it's hierarchical processing areas[2]. Bottom: The human visual cortex and the 2 streams (ventral & dorsal) and the frontal eye field (FEF)[3].

addition to the frontal eye field) located in the intraparietal sulcus (Bullier, 2001; Oliver, Rosario, & Pentland, 2000; Posner & Petersen, 1990). A 2003 fMRI study by McCoy *et al.* revealed saccadic reward, that is the value of the information gained under a selected policy (i.e. a specific saccade movement) may be linked to the posterior cingulate cortex, thereby linking a neuroanatomical location to the active inference framework for decision making based on previous beliefs of sampled hidden states (McCoy *et al.*, 2003). The results from this

[2] Courtesy of : https://thebrain.mcgill.ca/flash/i/i_02/i_02_cr/i_02_cr_vis/i_02_cr_vis.html
[3] Inspired by : https://en.wikipedia.org/wiki/Visual_cortex#/media/File:Ventral-dorsal_streams.svg

research effectively attribute salience-seeking behaviours to dopaminergic projections in the posterior cingulate cortex.

Saccadic movement has been found to be controlled by a region in the frontal lobe called the Frontal Eye Field (FEF). The FEF is responsible for our voluntary and reflexive eye-movements. A 2009 study found that responses to stimuli in the FEF were as rapid as 23ms after a new stimulus was presented (Kirchner *et al.*, 2009). This is suggestive of a "shortcut" pathway between the eyes and higher cortical regions, bypassing the visual cortex. The FEF is further believed to be important in visual attention and perception, as suggested in a TMS study by Grosbras & Paus (Grosbras & Paus, 2002).

Eye movement is widely used for observable behaviour measures given its rapid and direct link to perception inference (Oliver *et al*., 2000). For this reason, eye motion caption is an optimal modality for comparative analyses in testing the active inference framework. It is a sensory input an agent may rapidly and consciously redirect to whichever location they intend on sampling (i.e. acquire information on their environment's hidden states). While binocular vision is primordial for accurate depth perception, this study will focus primarily on the nature of a 2-dimensional scene's identity, eliminating the need for depth perception.

In a 2014 paper, Gavornik & Bear found that through sequence learning, activation in V1 is increased during pattern recognition (Gavornik & Bear, 2014). Of particular interest is the activation of cells for each element of a sequence that all activate even when a visual element is omitted from the sequence. This shows the brain is primed to recognize a sequence, even at a lower level. From this, it recreates the learned sequence, even in the absence of visual stimuli, as evidence of neural mapping of prior beliefs (Ernst & Banks, 2002).

Computational modelling & the MNIST Database

The MNIST database is a popular database of 60'000 handwritten digits, that is commonly used to train and test machine learning models (Huang *et al.* , 2015; Rawat & Wang, 2017). It is seen as a benchmark in computer vision for artificial intelligence and is ubiquitous in the field of machine learning. The human capacity to evaluate a wide range of handwritings pertaining to one same digit or distinguishing two similarly-looking digits from one-another is a feature that encapsulates the essence of visual processing (Ernst & Banks, 2002; Turk-Browne *et al.* , 2005). A comparative analysis between a behavioural experiment and a computer model using the MNIST database is therefore an ideal way to test how well the model correlates with the observed scene exploration behaviour in humans. The digits exist in 28 x 28 bitmap matrices (28 x 28 pixels).

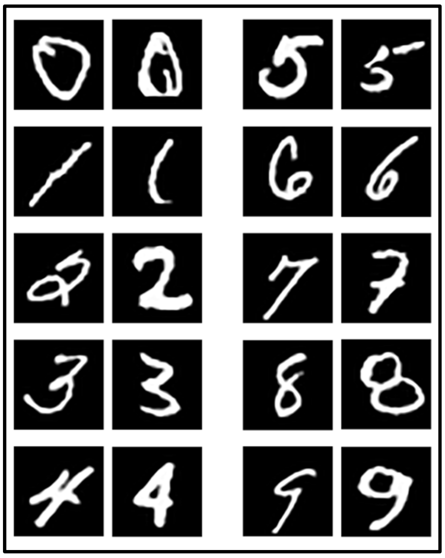


Figure 7. Sample examples for all 10 digits from the MNIST database

As the number of possible digits is limited to 10, this makes creating a testable model less taxing than using an alphanumeric character database, while the richness of the database ensures accurate model training and testing. Furthermore, by using an MDP model under the active inference framework, we may observe a learning curve the model adopts before gaining in confidence and noticing an increase in performance. Figure 7 shows several diverse examples of handwritten digits that can be found in the MNIST database.

Computational models are extensively used in neuroscience for their effectiveness in better defining the underlying mechanism of neural processes and networks (Ernst & Banks, 2002; Itti & Koch, 2001). Furthermore, these models may act as proxies for given cohorts. This means they may inherently act as correlated populations would when exposed to a set task. This notion is used for computational phenotyping, whereby models of various conditions may be trained and compared to one-another on countless tasks (Montague *et al.*, 2012). This in turn yields data that may be extrapolated for further analyses based on new hypotheses that emerge from the comparative computational tests.

A current explosion in deep convolutional neural networks has seen exponential interest in computer vision (Rawat & Wang, 2017). Countless models and algorithms allege to have human-like capabilities in terms of visual identification (Rawat & Wang, 2017; Schmidhuber, 2015). These neural networks, for the most part, perform on a pixel-by-pixel basis. This means each of the smallest elements is analysed sequentially to build a "whole picture". Other algorithms, including some using backpropagation attempt to solve character recognition using minimal preprocessing (Lecun *et al.*, 1998).Here we bring an alternative view, asserting that human vision is built upon beliefs of an agent's environment. Therefore, rather than expending useless amounts of energy in exploring the entirety of a scene, we suggest human vision seeks only the most salient features of a scene. These features are selected based on our prior belief of the value of the information our foraging will yield (Pezzulo *et al.*, 2015).

Hypotheses

In this study we attempt to explain visual exploration for scene identification under the active inference framework. To do this, we will compare empirical behavioural data, collected by eye-tracking using a visual paradigm, with simulations obtained from a task-specific MDP.

Based on the free energy principle, we hypothesize that scene identification is performed by scanning patterns that depend on prior beliefs about the scene. In essence, exploration of hidden states is done by selecting policies (i.e. saccading to display locations) based on the beliefs about the scene created after sampling it in the previous timestep. This means our actions for visual exploration depend on our beliefs. Saccading is driven by these perceptions, as a top-down process. Furthermore, this innate impulse to explore a scene comes from the natural desire to reduce surprise (or free energy) to a minimum (Friston, 2011; Friston *et al.*, 2016; Mirza *et al*., 2018).

To test this hypothesis, we created three visual tasks to be performed by participants, wherein the scenes to be identified are digits from the MNIST database. These visual stimuli are partially occluded in a gaze-contingent paradigm. This allows for visual scene exploration, while reducing the immediate use of peripheral vision for latent context on the scene's identity. These tasks will each comprise of a series of scenes to be identified in a time-sensitive manner and be given instantaneous feedback on scene identity reported. The first task is a sequence of digits based on the premise of built expectation (Gavornik & Bear, 2014). In addition to an expected learning effect (observed as a gradual decrease in trial duration) due to repetitve exposure, we expect scanning patterns to differ between early and late trials for a given digit in the sequence. To subvert expectations, an occasional oddball stimulus is inserted. On these occasions, we expect an increase in active sampling, explained by an increase in surprise (i.e. the agent's belief of the scene's nature fails to match the observed outcome of a hidden state, causing a policy selection based on higher epistemic value). The second task further builds on this reasoning. It investagtes how exploration of a given scene is performed when its likelihood of being displayed is high, compared to when this likelihood is significantly lower. Two digits are chosen to alternate in occurence probability, set in predefined blocks. We expect to notice longer trial durations and increase visual foraging in blocks of low expectancy for a digit, when compared to the blocks of high expectancy. This is directly linked to the hypothesis of surprise minimization based on prior beliefs. The third task presents a series of random digits, each with the same likelihood of appearance. This task attempts to analyse scan paths for scene-specific identities with no prior belief about the nature of the scene. By analyzing the scan paths of in this task, we hope to determine distinct active sampling patterns to accurately identify a scene.

In addition to testing these experimental hypotheses, we will attempt to develop an MDP model based on the parameters set out by all three tasks for a direct comparison between the model and empirical data. This will allow us to correlate the model in addition to the either supporting or rejecting the overarching hypothesis under the active inference framework as a model explanation for visual foraging behaviour.

Materials and Method

Experimental Paradigm

This study was sectioned into three tasks, which had gaze-contingent eye-tracking visual paradigms closely resembling each other. All three tasks required correctly identifying an MNIST digit that was partially obstructed in view by a foveated mask, through which only part of the displayed digit could be observed (see Figure 8). The onus is on participants to explore the display as the gaze-contingent feature of this design forces the foveation in the mask to follow participants' gaze (thereby progressively revealing the digit). A visual analogy would be shining a flashlight in a dark room to reveal what digit is painted on the wall. The display's resolution was 1280 x 1024 pixels. All MNIST digits were rescaled to 1024 X 1024 (up from their original 28 x 28 pixel dimensions) and converted to grayscale 8bpc bitmap files, using Adobe Photoshop.

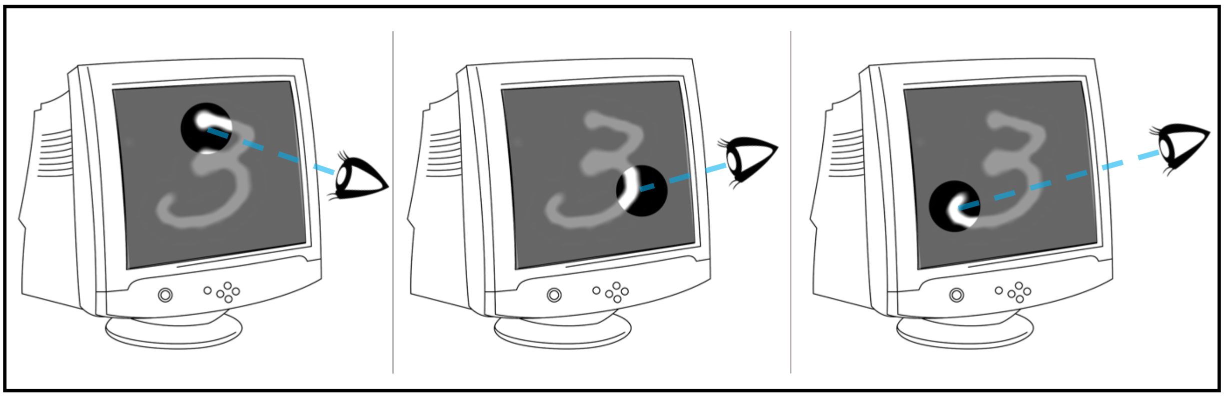


Figure 8. The gaze-contingent paradigm ensures the foveated window of the mask follows the participant's gaze as they explore the display to reveal the hidden digit being displayed (NB: the mask has transparency for illustration purposes only – participants could not see the entirety of the underlying digit)

Task 1 (1 – 3 – 5 sequence):

Task one is a sequence learning task. Three digits were sequentially shown (1 – 3 – 5), with a time limit of 3 seconds per digit. Participants were encouraged to press the space bar as soon as they had identified the digit. This would end the current displaying of the digit and show a fixation cross in preparation for the next digit in the sequence. At the end of a sequence (the third digit in the sequence), a digit dial (see Appendices) would be displayed. Participants were asked to focus their gaze on the digit in the dial they had last identified while pressing the space bar to validate their answer. Following this they would receive immediate feedback as to whether their identification was correct or incorrect (see Figure 9 for an example of a sequence). The third digit (the digit assessed in the sequence) was selected to be a 5 on 70% of occasions, or another random digit on the remaining 30% of instances. A total of 120 digits were shown, yielding 40 sequences, and 40 response validations to be entered by participants with feedback.

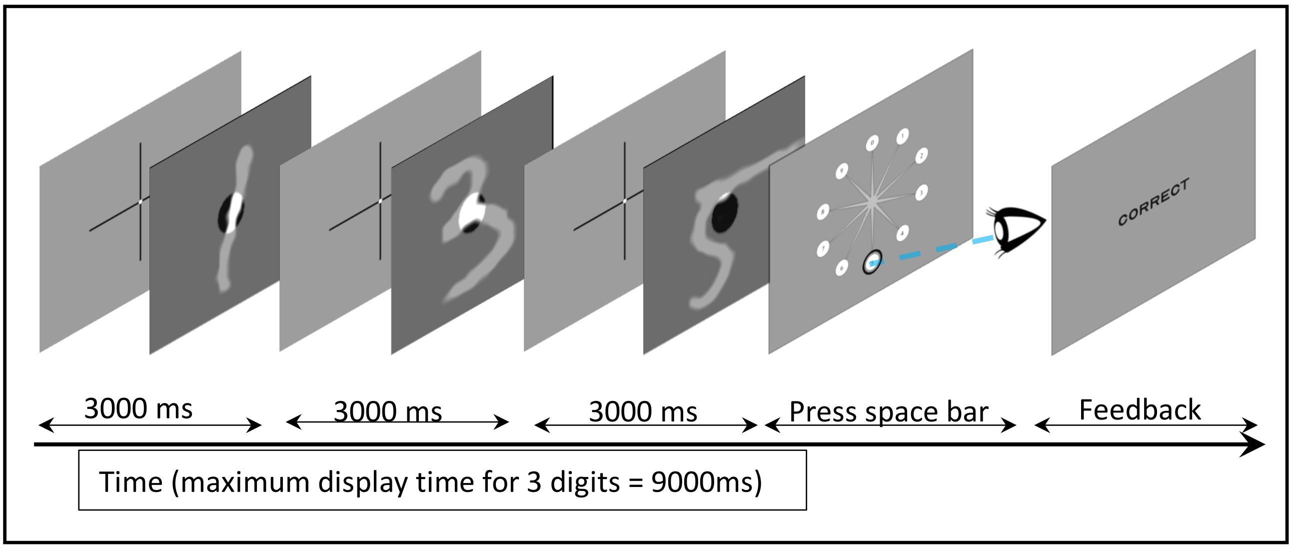


Figure 9. Timeline of displayed images during a typical (70% of cases) sequence. The foveated mask is semi-transparent here for illustration purposes, but opacity of the mask is full in the experimental paradigm.

Task 2 (volatility):

Task two consisted of a sequence of randomly selected digits based on a probabilistic distribution. The entire task displayed only the digits 2 and 8, in an attempt to create a task matching the principle of volatility, where the probability of a stimulus presenting itself shifts from one block to the next. Here the probability of seeing the digit 2 flips between 20% and 80%, and respectively the inverse for the digit 8. These flips in probability happened at a frequency of every 20 digits. Figure 10 summarizes this task's sequence timeline and the succession of probability distributions respective to both digits. In this task, a total of 100 digits were displayed, including five distribution flips in likelihood of digit occurrence. Participants are asked to validate an answer using the digit dial only once every five digits, for a total of 20 digits assessed and given feedback on for their answers. Here again a time limit of 3 seconds per digit is set, with participants being encouraged to press the space bar as soon as the digit has been identified, either bringing a fixation cross up in preparation for the next digit, or the digit dial if the digit is being assessed.

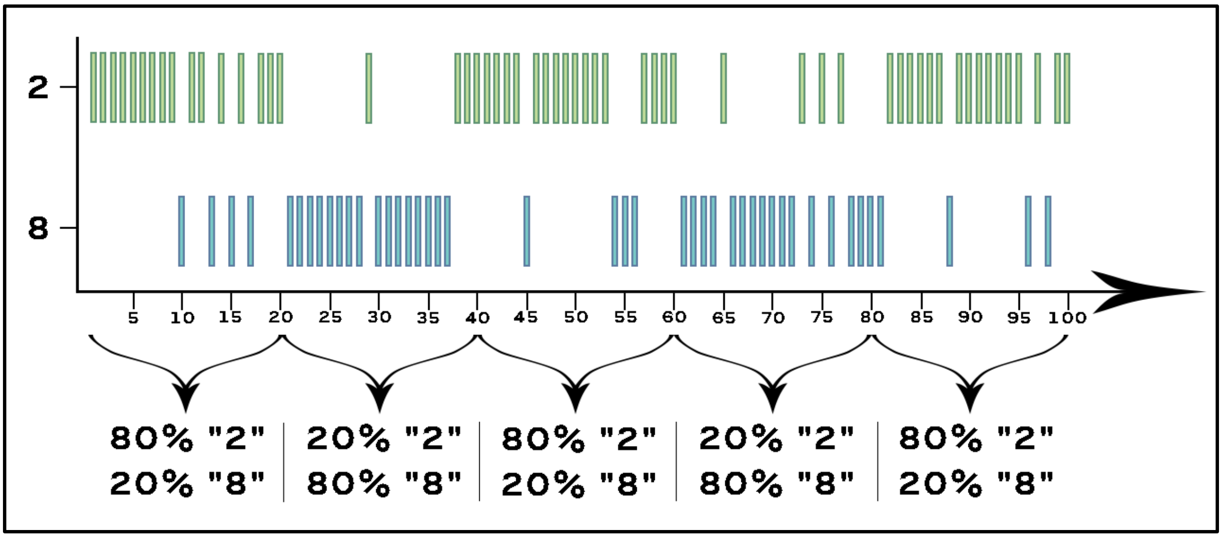


Figure 10. Sequence timeline of the digits displayed in task 2. The bottom of the figure shows the probability of each distribution appearing per block of 20 digits displayed (5 blocks in total). The digits shown on the abscissa represent the digit number and are shown for all digits that were assessed for feedback (20 in total).

Task 3 (random):

Task three displayed a sequence of 100 fully randomized digits. A list of 1000 digits taken from the MNIST database was created, and each digit is randomly selected from that list to be displayed. In this task, each trial also has a time limit of 3 seconds, with a reiterated goal of pressing the space bar once the digit has been identified. Upon pressing the space bar, or once the three-second time limit has been reached, the digit decision-dial is shown. Identification input occurs in each instance, for a total of 100 digits displayed and being assessed for feedback (see Figure 11).

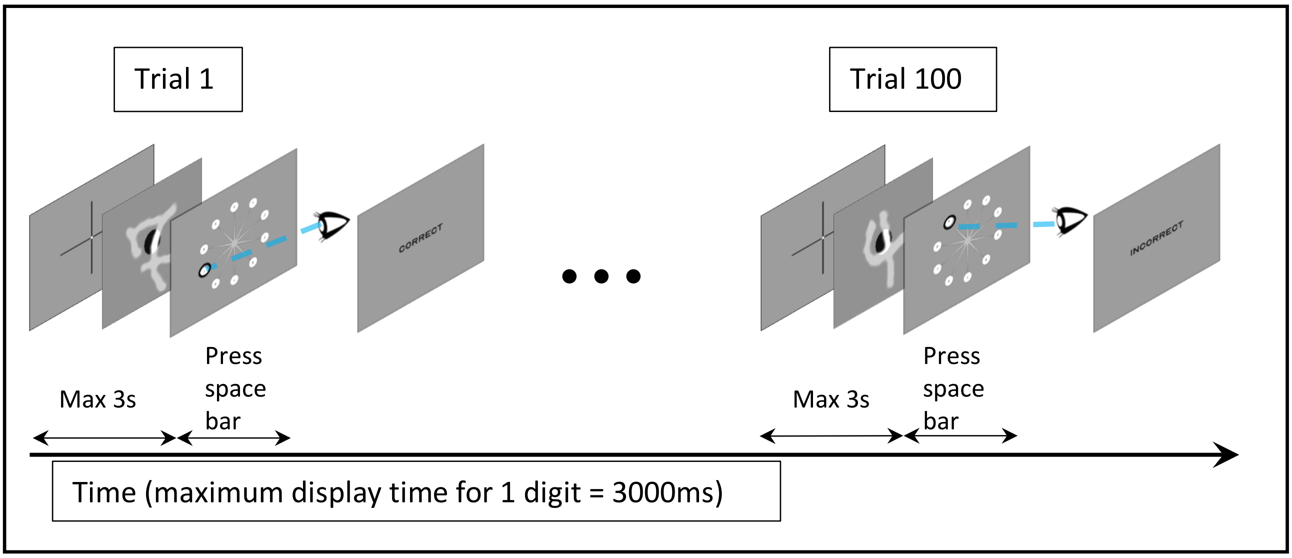


Figure 11. Example sequence of first and last digits in task 3. Each digit is assessed and given immediate feedback on, after being shown until the participant presses the space bar, or a maximum of 3 seconds has elapsed. Here as well the foveated mask is semi-transparent for illustration purposes, but opacity of the mask is full in the experimental paradigm.

Eligibility Criteria

The eligibility criteria used for this study were the following:

- Able to give written informed consent.
- 18-65 years of age.
- English-speaking.
- Healthy and with no history or presence of neurological conditions or neuropsychiatric conditions.
- No history of spinal cord, neck and back injuries.
- No history of drug and alcohol abuse.
- Not be taking psychoactive medication.
- Not wear eye make up on the day of the study.
- A visual acuity of at least 20/40 either with or without glasses (glasses are permisible for eye-tracking, not contact lenses).
- Non-pregnant.

Materials

Participants sat in a chair and placed their heads in a headrest, to minimise head motion. The height of the chair was adjusted such that the forehead bar would align right above the eyebrows, and the chin rest was then adjusted to fit comfortably accordingly to the head's resting position (see Figure 12).

The paradigm was run on a Windows 7 computer with a cathode ray-tube monitor displaying at a resolution of 1280 x 1024 (85.3 Hz). The distance between the top of the forehead bar and the bottom of the monitor was 60cm. The preset angle of sight for this set-up was 24.4° (visual degrees) vertically, and 32.5° (visual degrees) horizontally. The eye-tracking camera was set-up to track the left eye exclusively. To adjust for pupil coverage, the setting was adjusted *ad hoc* after applying the in-built auto-threshold, prior to calibration. A standard keyboard was used for button presses. The paradigm was built and displayed using the Matlab toolbox Cogent2000 (see References).

Figure 12. Picture of Berk Mirza positioned with his head on the chinrest in front of the eye-tracker.

Experimental Procedure

Participant data acquisition was completed during the last two weeks of the month of July 2019. Participants were asked to perform a visual acuity (VA) test, assessed using a pocket Snellen chart. Participants were allowed to do the VA test using glasses if this is how they would perform the experimental tasks. We asked participants to confirm they were not wearing any contact lenses, nor eye make-up. Should participants have a VA below 20/40 (driving standard), we would flag their data for potential exclusion based on performance.

Following this, a brief explanation of the experimental tasks, as well as of the eye-tracking set-up was given, with the opportunity of asking any questions. Participants were informed that they would have a break and could move their heads between all three tasks. An approximate task duration of 10-15 minutes was disclosed, with variation between participants due to the self-pacing nature of the tasks.

Participants were asked to perform these tasks bearing two goals in mind: the first one being to correctly identify the digit displayed, the second to press the space bar as soon as they believed to have correctly identified the digit. Participants were reminded that this allowed them a certain self-pacing liberty, with the incentive of being remunerated a set amount (10£/hour, rounded up to the closest hour) while spending as little time performing the task as possible.

Participants were shown how to adjust the chin rest to comfortably support their heads, as well as adjust seat height, in order to maximise comfort while performing the tasks. Eye-tracking calibration was performed using a 12-points grid location mapping system, as an in-built function of the EyeLink 1000 OS. Participants were asked to keep any head motion to a strict minimum for accurate gaze tracking. As these experimental tasks are gaze-contingent, the importance of accurate reading was paramount to online task performance, in addition to offline data analysis. Calibration was performed before starting each task, as breaks between tasks allowed for free movement in order to minimise periods of discomfort.

A short training session of ten digits, using the same paradigm set-up as task 3 was initially completed to allow participants to get accustomed to the feeling of scene exploration using exclusively eye-motion, in addition to familiarizing them to the digits, the mask's aperture, and the answer validation method. After this, participants were asked if they wished to retry the training session or if they felt confident enough to start with the first task.

No information was given regarding the pre-selected sequences of tasks 1 and 2. Participants were simply told task 1 contained 120 digits, but they would only be asked to validate the last digit observed using the digit dial "every so often". Task 2 was explained to be similar, with 100 digits, and also asking participants to give feedback "every so often". Task 3 was revealed to be 100 digits, completely selected at random from the database, with assessed feedback to be given after each digit had been presented.

Computational model

In this section we will describe the MDP model used for the digit exploration and identification task. The results from the simulations obtained will serve as a comparison basis for our behavioural data.

The model that we considered is a hierarchical model with two separate levels where we indicated each unit's level (denoted by "(1)" & "(2)") (see Figure 13). On the first level (illustrated at the bottom of the figure), the model explores the features of a digit on a seven-segment display. There are two hidden states on this level, namely "digit identity" and "where (location)". The sampled "digit identity" and "where (location)" generate one of three orientation outcomes (i.e. digit features): null, _ (horizontal), or | (vertical). The "where (location)" hidden state also generates a "where" outcome (identity mapping) (see Figure 14).

Whilst the first level explores the digit and makes lower-level visual observations to form beliefs about the observed digit's identity, the second-level addresses how those beliefs are reported. On the second level there are two hidden states, namely "digit identity" and "identity selection". The "digit identity" hidden state determines the identity of the digit that will be explored in the lower level. This hidden state generates the "digit identity" outcome (identity mapping), which is passed to the lower level as the initial hidden state (see dashed arrows in Figure 13). The agent reports its beliefs about the digits by sampling one of the eleven selections encoded in the "identity selection" hidden state. Based on the sampled choice location, the agent receives one of three feedback outcomes: correct, incorrect, or undecided. This works such that if the sampled choice location is the same as the digit explored, the agent is given a "correct" feedback. If the agent selects any other identity (except for identity 11), it will receive an "incorrect" feedback. Sampling location 11 generates

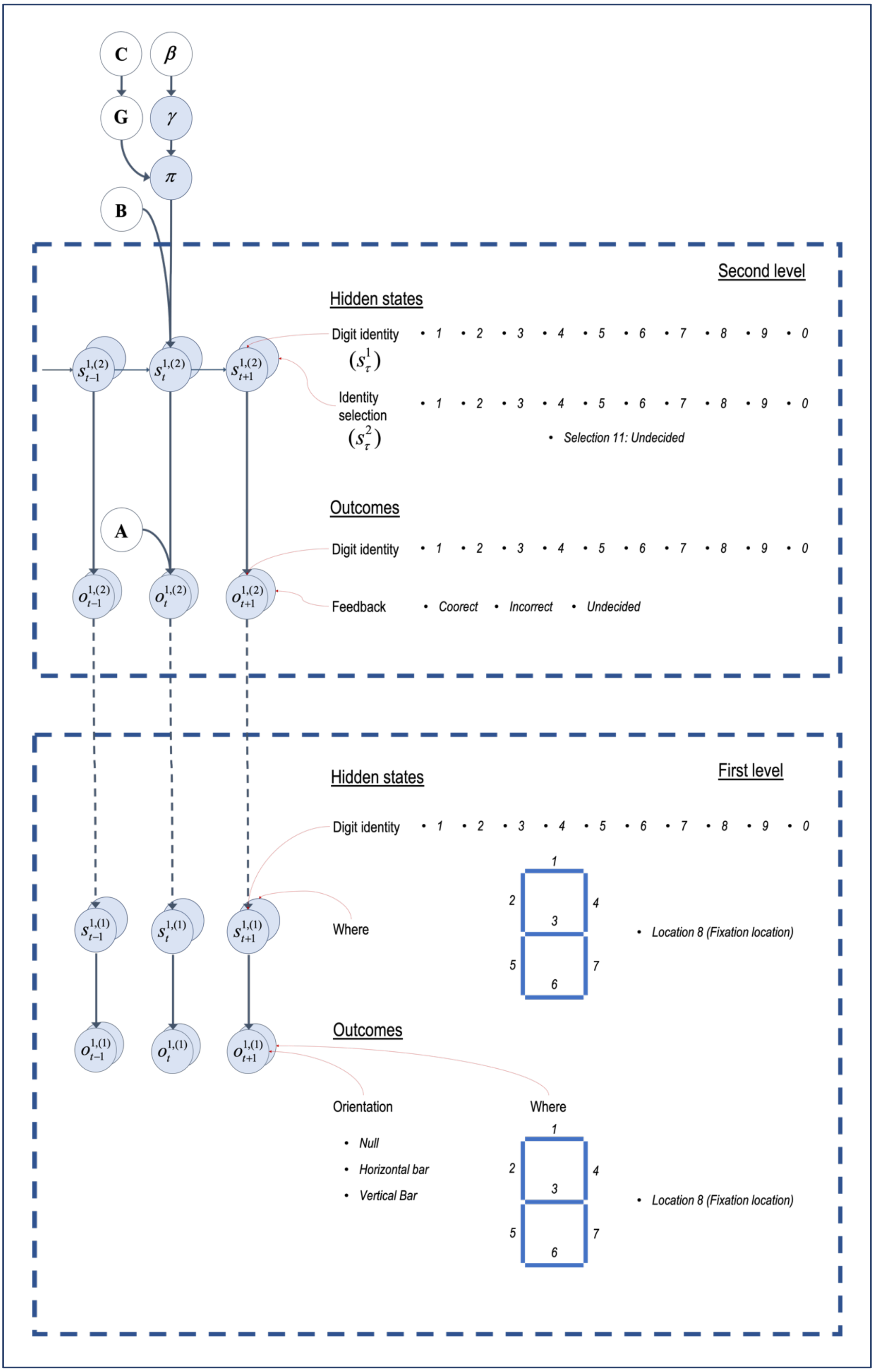


Figure 13. Schematic design of the MDP model built for digit identification. Each digit is comprised of features (horizontal, vertical, or null) in each of the seven locations that build the scene. The eighth location serves as a starting location and holds no digit-specific features.

the “undecided” outcome, regardless of the digit’s true identity. The agent will remain in this state until a decision is made about the identity of the digit. The agent then uses the observed features to update its beliefs about the hidden state “digit identity”.

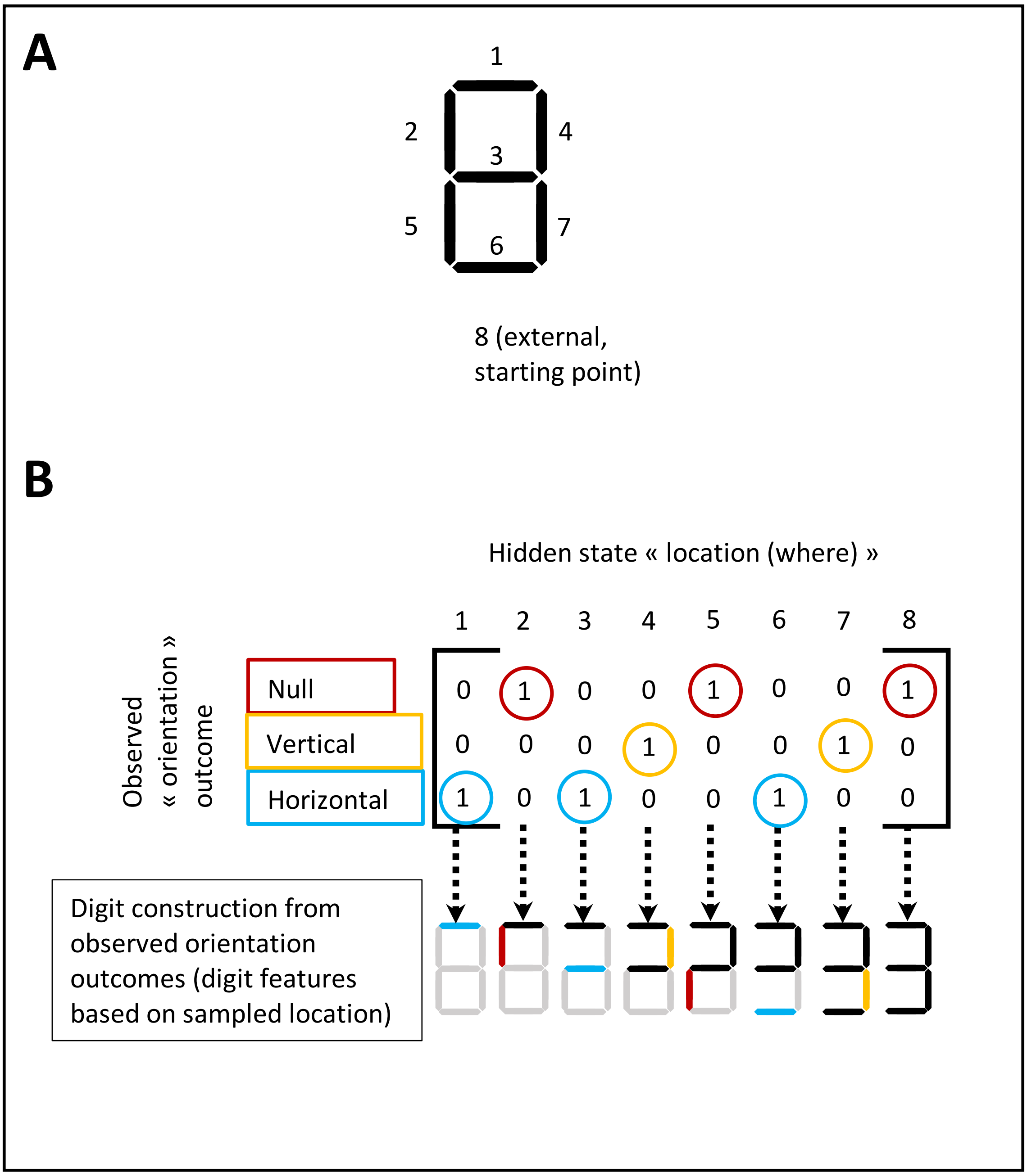


Figure 14. **A**: This shows the mapping “location (where)” hidden states (as numbered), with location 8 being a starting location, containing “null” as an outcome orientation for all digits. **B**: Here we show the likelihood matrix for digit 3, with the orientation outcomes associated with each “location (where)” hidden state.

Qualitative Work (analytical procedures & data cleaning)

All data processing was performed using self-coded scripts in Matlab2019Ra (see Appendices). Gaze location for scan path information was taken from the output ".mat" files encoded by matlab at a sampling rate of 85.3Hz during task performance. The EyeLink 100 system also outputs its own ".edf" file at a higher sampling rate of 1kHz. The ".edf" files were converted to ASCII format for processing in matlab. EyeLink's innate system labels blinks, saccades and fixations using in-built algorithms, allowing for simplified processing of data.

To apply statistical parametric modelling to the scan paths observed, we developed a novel method to visualize and process eye-tracking data. Using fMRI methods for volume-based statistical analysis, we created volumes containing the scan paths for all trials for further analysis using SPM12. In all three tasks, scan paths were extrapolated into NIfTI format. The dimension of the ".img" subfile were X = horizontal gaze location on the display, Y = vertical gaze location on the display, Z = the sampling timestep for each discrete gaze location. Location was rounded to the nearest ten, and display size (X, Y coordinates) were then down-sampled to 10% (128 x 102). This accounted for foveation size (the diameter of the aperture in the mask) in addition to reducing noise in the data caused by micro-saccades and random effects. A gaussian full-width half-maximum (FWHM) smoothing kernel of [5 5 10] was applied to all files in all tasks. The decision to increase the smoothing step in time was made to account for variation in fixation durations between participants. This allows increased flexibility for fixation durations, while correlating fixation locations.

Conversion of the extrapolated 3D Matlab arrays to NIfTI format was done using the Matlab toolkit called "Tools for NIfTI and ANALYZE image", by Jimmy Shen. Viewing planespecific 3D arrays (used for videos of canonical scan paths – see QR codes) was done using the Matlab toolkit called "volumeViewer" by Ran Klein.

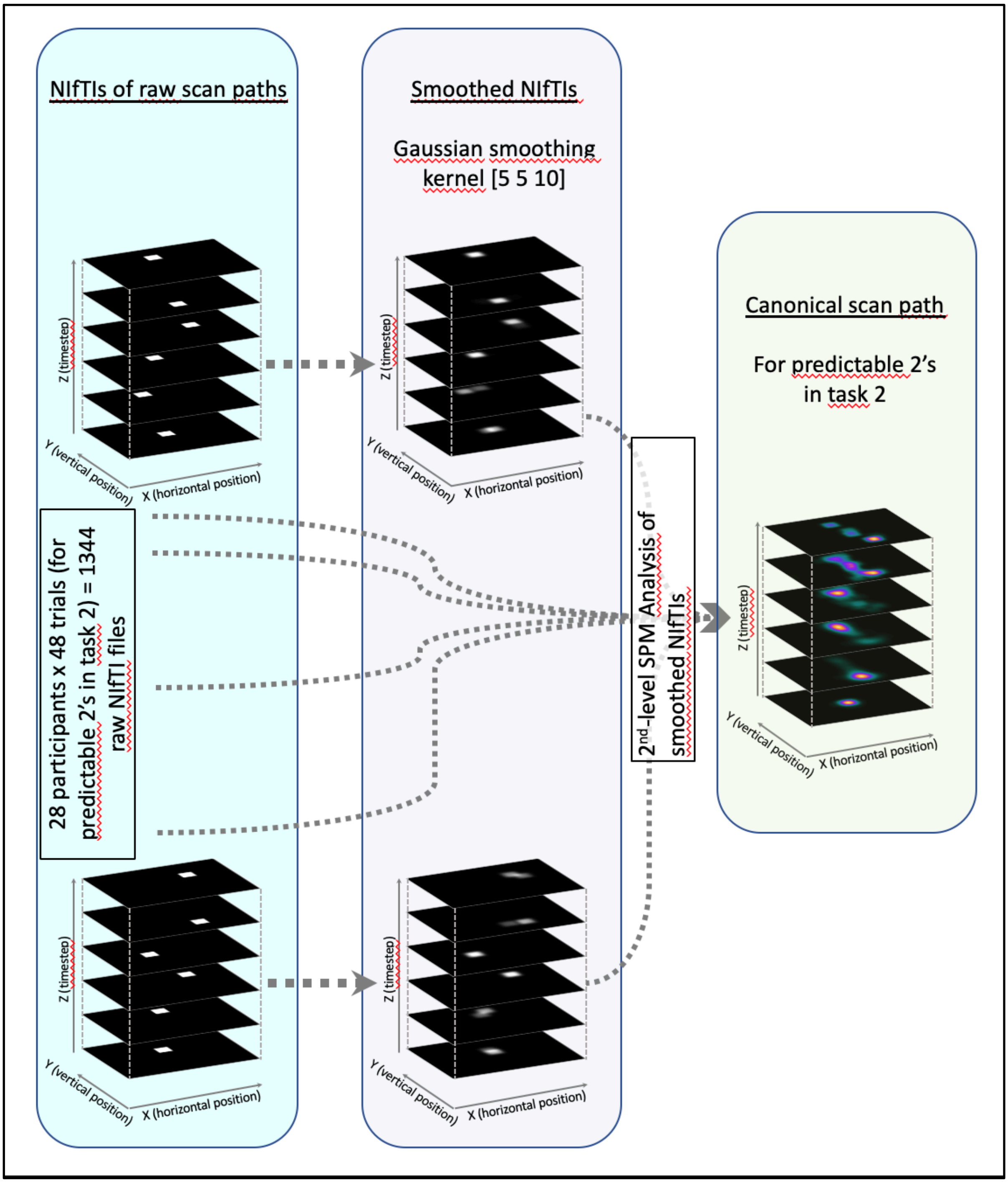


Figure 15. Scan path processing pipeline: 1) 3D matrix (Z dimension = time) is converted into NIfTI format at 10% the resolution of the display; 2) A smoothing kernel is applied to the raw data; 3) Paradigmatic scan paths are generated by performing a 2nd-level SPM analysis on the combined smoothed NIfTIs.

After pre-processing the data in Matlab, 3D arrays were stored in a variable that was converted into NIfTI format, which was then saved to be fed into the processing pipeline outline in Figure 15. A total of 280 files (28 x 10) files were created for each of the early/late 1's/3's in task 1 (grand total of 1120 files for task 1). Task 2 had 16 instances of predictable (80% probability) 2's and 4 unpredictable (20% probability) 8's in blocks 1,3,5, yielding 560 files (20 x 28) each, and 16 instances of predictable 8's and 4 unpredictable 2's in blocks 2,4, yielding 560 files (20 x 28) each (grand total of 2800 files for task 2). Due to the random presentation of digits in task 3, not all of the ten digits were displayed the same number of times.

To obtain a paradigmatic (significantly correlated across all participants) scan path, all NIfTIs for a same category were tested for fixed effects using a one-sample t-test in a 2nd-level SPM analysis. After model estimation, the contrast images were evaluated using a t-contrast matrix of [1]. We further developed the visualization method for these canonical scan paths to be interpreted in a 2D image. To do this the values from the output paradigmatic scan paths were compressed by summing all values in the Z coordinate (time) for a given X, Y location (gaze location on the display). This output matrix is a correlational matrix of the paradigmatic scan paths. This means the central fixation which initiates each trial had the highest t-score values, thereby biasing the overall weighted representation of the paradigmatic scan path. For this reason, it was necessary to remove the initial central fixation to un-bias the compressed 2D image (see Appendices for an example render with and without central fixation filtering). The results for these compressed paradigmatic scan paths in task 1 (early vs. late 1's and 3's) can be seen in Figure 18. The results for task 2 (predictable vs. unpredictable 2's and 8's) are shown in Figure 21. Canonical (paradigmatic) scan paths for all ten digits based on the observations in task 3 may be found Figure 25 and Figure 26.

Results

Demographic Data

A total of 30 participants were recruited: 19 females and 11 males. The ages of participants ranged between 20 – 34 (mean 26.5 years old, SD = 3.5 years). 28 participants had a visual acuity of 20/20, read from a Pocket Snellen Chart (at 2m distance). This measurement was taken with participants in experimental condition (i.e. with or without glasses, depending on their needs). Two participants had a visual acuity of 20/25. All participants had over 15 years of total schooling. Recruitment was done through an internal circular for the IoPPN (King's College London). Participants all gave written consent to take part in the study and have their data analysed. Ethical approval was granted by the Psychiatry, Nursing and Midwifery Research Ethics Subcommittee (King's College London Research Ethics Committee), with ethical approval reference: MRA-18/19-11544.

Excluded Data

From the 30 participants recruited, data from 2 of the participants could not be acquired. Despite meeting all eligibility criteria, the eye-tracking device was unable to accurately continuously located their gaze, causing flickering, no control of the foveated mask and impossibility to validate their answers. Both participants had strongly corrected glasses with significant thickness, believed to be the cause for this. Another participant with very weakly corrected glasses also had similar issues, however they were able to perform the task without their glasses. Upon inspecting their glasses, we noticed the thickness was below that of others with whom no problems had arisen, however it was noted they were blue-light-filtering glasses, which may have caused scattering of the infrared light emitted by the EyeLink 1000 for the eye-tracker's camera to read. Thus, the analysed data is taken from a sample population of n = 28.

Summary of Main Data

Overall Results:

Table 1. Mean trial duration, standard deviation (SD), standard error (SE) and ANOVA results for all three tasks.

Trial Duration	Mean (seconds)	SD (seconds)	SE (seconds)
Task 1 (sequence)	2.11	1.03	0.02
Task 2 (volatility)	1.95	0.98	0.02
Task 3 (random)	2.06	0.68	0.01
Statistical test	One-way ANOVA: F-score = 24.13, p-value = 3.55×10^{-11}, df = 8957		

Table 2. Mean response accuracy, SD, SE and ANOVA results for all three tasks.

Response Accuracy	Mean	SD	SE
Task 1 (sequence)	83.84%	36.69%	1.10%
Task 2 (volatility)	89.11%	31.18%	1.32%
Task 3 (random)	88.18%	32.29%	0.61%
Statistical test	One-way ANOVA: F-score = 7.03, p-value = 7.00×10^{-4}, df = 4477		

Task 1:

Table 3. Mean trial duration, SD, SE and T-test results for 5 and non-5 digits in task 1.

Trial Duration	Mean (seconds)	SD (seconds)	SE (seconds)
5 digits	2.66	1.00	0.04
Non-5 digits	2.36	0.98	0.05
Statistical test	Two-sample unpaired T-test: t-score = 4.70, p-value = 2.96×10^{-6}, df = 1116		

Table 4. Mean response accuracy, SD, SE and T-test results for 5 and non-5 digits in task 1.

Response Accuracy	Mean	SD	SE
5 digits	79.60%	40.33%	1.44%
Non-5 digits	94.31%	23.20%	1.27%
Statistical test	Two-sample unpaired T-test: t-score = -6,24, p-value = 6.06×10^{-6}, df = 1116		

Task 3:

Table 5. Mean gaze displacement, SD, SE and T-test results for horizontal and vertical movements in task 3.

Gaze Displacement	Mean (pixels)	SD (pixels)	SE (pixels)
Horizontal	1.97×10^{5}	4.18×10^{4}	1.32×10^{4}
Vertical	2.51×10^{5}	5.01×10^{4}	1.58×10^{4}
Statistical test	One-sample paired T-test: t-score = -2.93, p-value = 1.67×10^{-2}, df = 9		

Analyses

Overall results:

We sought to determine if there was a significant difference in performance (trial duration and response accuracy) between all three tasks. The average trial duration was a direct measure of time difference between the end of a digit being displayed and when it first appeared on the display. Statistical testing was performed using a one-way ANOVA. All three tasks displayed significantly different average trial durations, with the shortest being task 2, and the longest being task 3 (F-score = 24.13, $p = 3.55\text{x}10^{-11}$, df = 8957) (see Table 1).

Percent accuracy overall was also measured and yielded a result of 87.04% (SD = 2.82%, SE = 1.62%). Task 2 showed the highest accuracy, with an average accuracy of 89.11% (SD = 31.38%). A one-way ANOVA also allowed to compare performance across all three tasks, producing a significant result (F-score = 7.03, $p = 3.55\text{x}10^{-4}$, df = 4477) (see Table 2).

Task 1:

In task 1, responses were given on the third digit of a 3-digit sequence (1 – 3 – 5/other). On 70% of occasions, the third digit was a 5, and another digit on the remaining instances. We performed a two-sample unpaired t-test on the accuracy of the reported digit for 5 against non-5 digits. There was a significant difference between 5’s (79.6 % accuracy, SD = 40.3%, SE = 1.44%) and non-5’s (94.3 % accuracy, SD = 23.2%, SE = 1.27) (t-score = -249.05, p = 0.00, df = 999) (see Table 3). A two-sample unpaired t-test was also performed to compare trial duration distributions between both groups. The average trial duration for the digit 5 was 2.66(s) (SD = 1.00(s), SE = 0.04(s)), and 2.36(s) (SD = 0.98, SE = 0.05) for non-5 digits. The difference between these results was also statistically significant (t-score = 151.86, p-value = 0.00, df = 999) (see table 4).

To test for a learning pattern in the sequence, the scan paths of the first and last ten 1's and 3's were extrapolated into NIfTI format for 2nd-level SPM analysis. Heatmaps showing highly correlated points of visual salience for scene identification in early vs. late 1's and 3's are shown in Figure 18. A subtraction mask was then created to reveal areas of increased gaze location in early/late trials, referred to as signal. A decrease in signal for later trials for both 1's and 3's is observed in the subtraction masks in Figure 18. The QR code in Figure 18 links to a video showing a direct comparison between these two canonical scan paths.

Task 2:

To test the volatility hypothesis, where we would expect an increase in trial duration and visual exploration of the scene in blocks of low likelihood versus blocks of high likelihood, scan paths were extrapolated for "predictable" and "unpredictable" 2's and 8's into NIfTI format for 2nd-level SPM analysis. Heatmaps of significantly correlated scan paths (paradigmatic scan paths) were produced for each of the four categories (see Figure 21). A subtraction mask was then created to reveal areas of increased gaze location in predictable/unpredictable trials. An increase in signal for unpredictable trials for both 2's and 8's is observed in the subtraction masks in Figure 21. The QR code in Figure 21 links to a video showing a direct comparison between these two canonical scan paths. Figure 20 shows all gaze fixations during the trials of all 2's and all 8's. This visualization of fixation location allows to appreciate the variation in scene exploration between both digits. This figure shows all gaze locations on every trial for each participant.

Task 3:

In task 3, we show how vertical and horizontal movements may differ, and therefore allows us to infer value of information stored in locations for further comparison in policy choice in the MDP model. As task 3 shows all digits with equal probability, it answers the question of how an agent resolves uncertainty about a digit's identity with minimal prior belief about which digit will be displayed. Figure 18 shows the average values of horizontal and vertical gaze displacement per digit. This is also an indirect measure of total scan path length. Trial duration may not be directly inferred, as this does not take into account any notion of fixation duration. The two waterfall graphs in Figure 24 are digit-independent but show instead how horizontal versus vertical movements occur on average during trials, and to which scale. The QR code in Figure 26 links to a video displaying the paradigmatic scan paths for all ten digits, using the same method of scan path extrapolation into NIfTI format for 2nd-level SPM analysis. An error matrix is shown in Figure 22, displaying the most commonly made errors in digit identification by participants, with digits 2, 6, and 8 being the most often mistaken for other digits. Table 5 shows the mean value of horizontal displacement for all digits (1.97x10^5 pixels, SD = 4.18x10^4 pixels, SE = 1.32x10^4 pixels), and vertical displacement (2.51x10^5 pixels, SD = 5.01x10^4 pixels; SE =1.58x10^4 pixels). We performed a one-sample paired t-test to compare both distributions on all ten digits (t-score = -2.93, p-value = 0.017, df = 9) (see Table 5).

MDP model:

In Figure 27, we show the learning of digit identifying features (contained in the likelihood matrices A) for each digit by our model. Figure 28 is evidence of better performance of the model as it becomes more confident about the features observed and how they relate to a specific identiy. By learning the features, the model gains in confidences and the number of undecided outcomes is reduced. Its prior beliefs build a better certainty of accurate identification.

Figures

Overall Results:

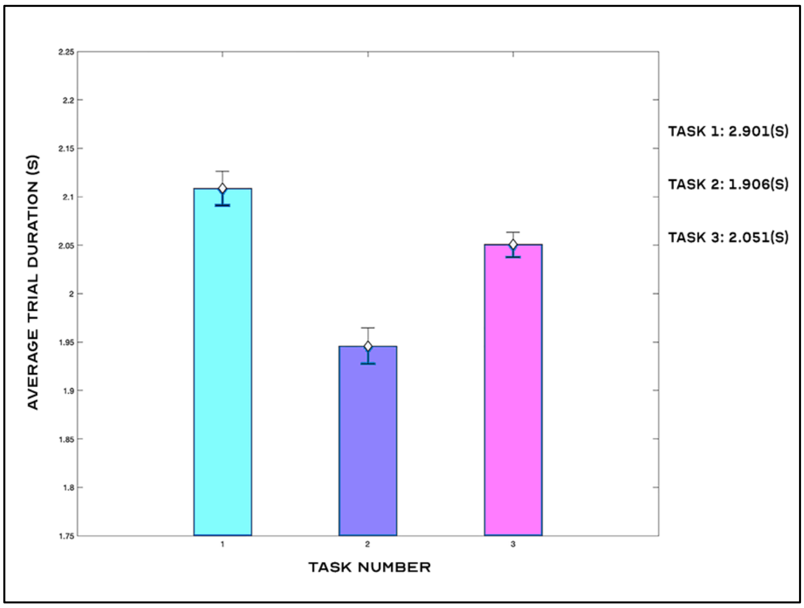


Figure 16. Average trial durations for tasks 1, 2, and 3 (error bars represent respective SE).

Task 1:

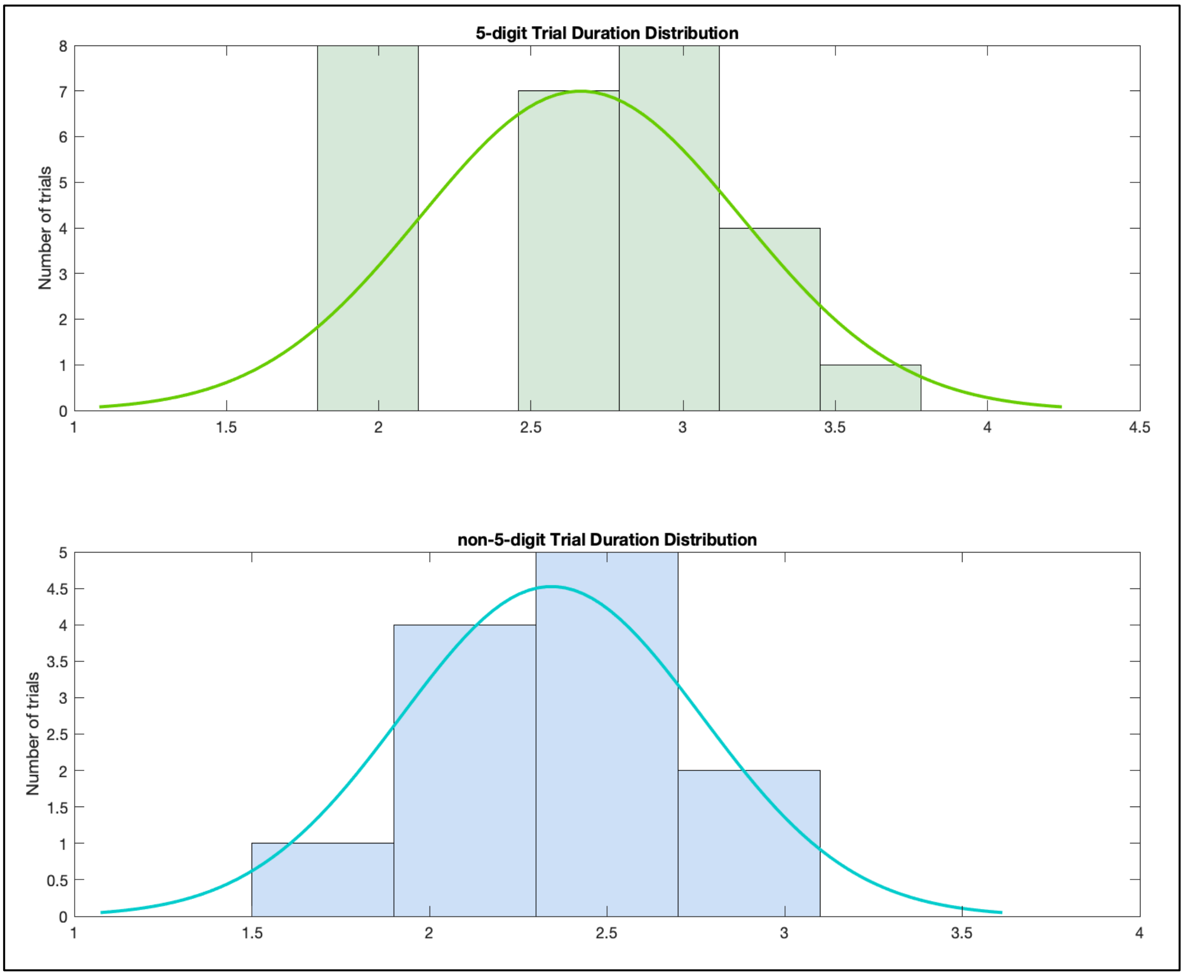


Figure 17. Top: distribution of average trial durations for the third digits in each sequence that were 5's in task 1 with normal distribution curve. Bottom: distribution of average trial durations for the third digits in each sequence that were non-5's in task 1 with normal distribution curve.

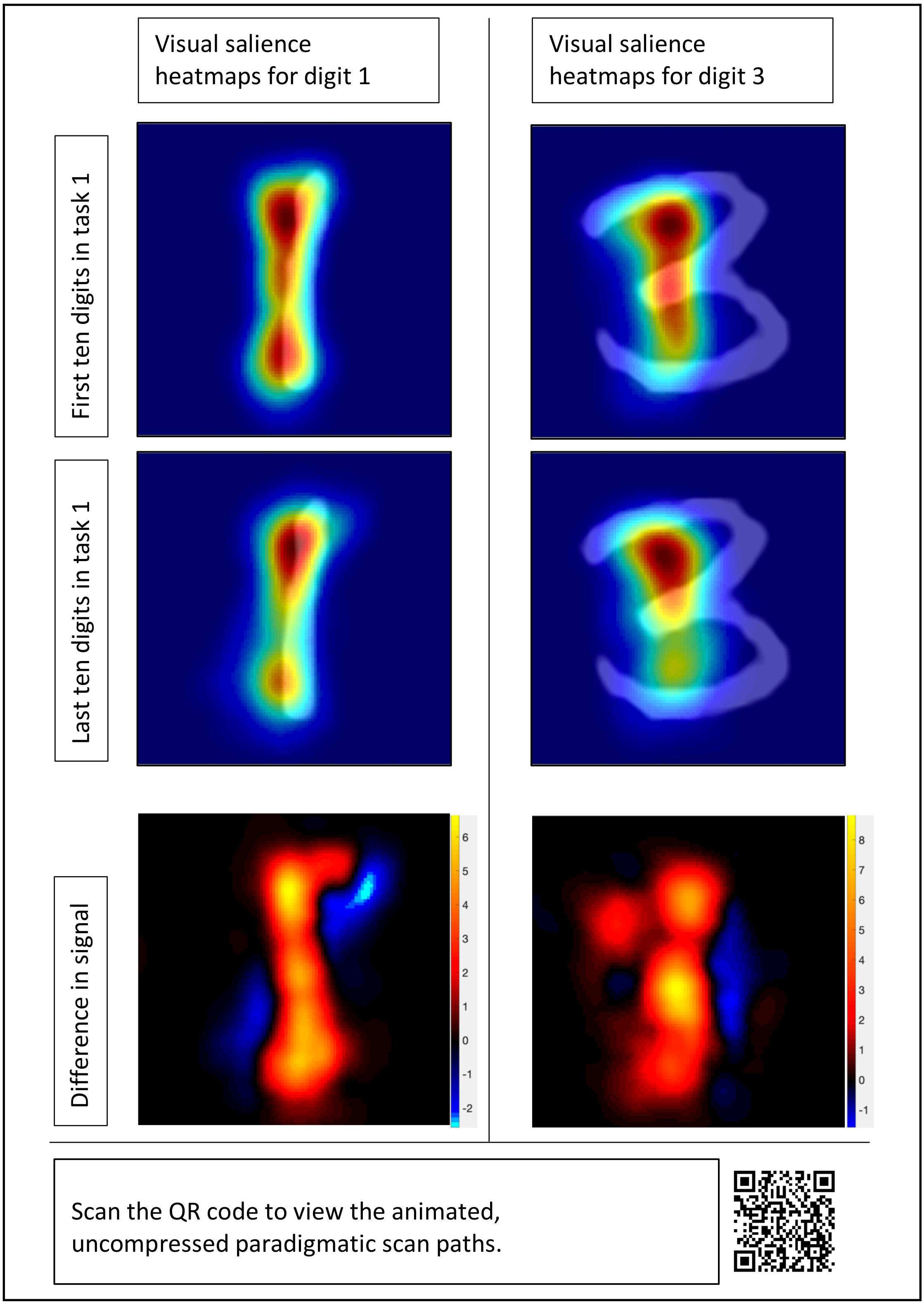


Figure 18. Compressed paradigmatic scan paths for early and late 1's and 3's in task 1. The difference in signal images represent increased probability of gaze location in early digits (hot colours) and late digits (cold colours)[4].

[4] QR code URL: https://drive.google.com/open?id=1OZLMuTjRT6eZav95vfQumJbU1PeVXSVT

Task 2:

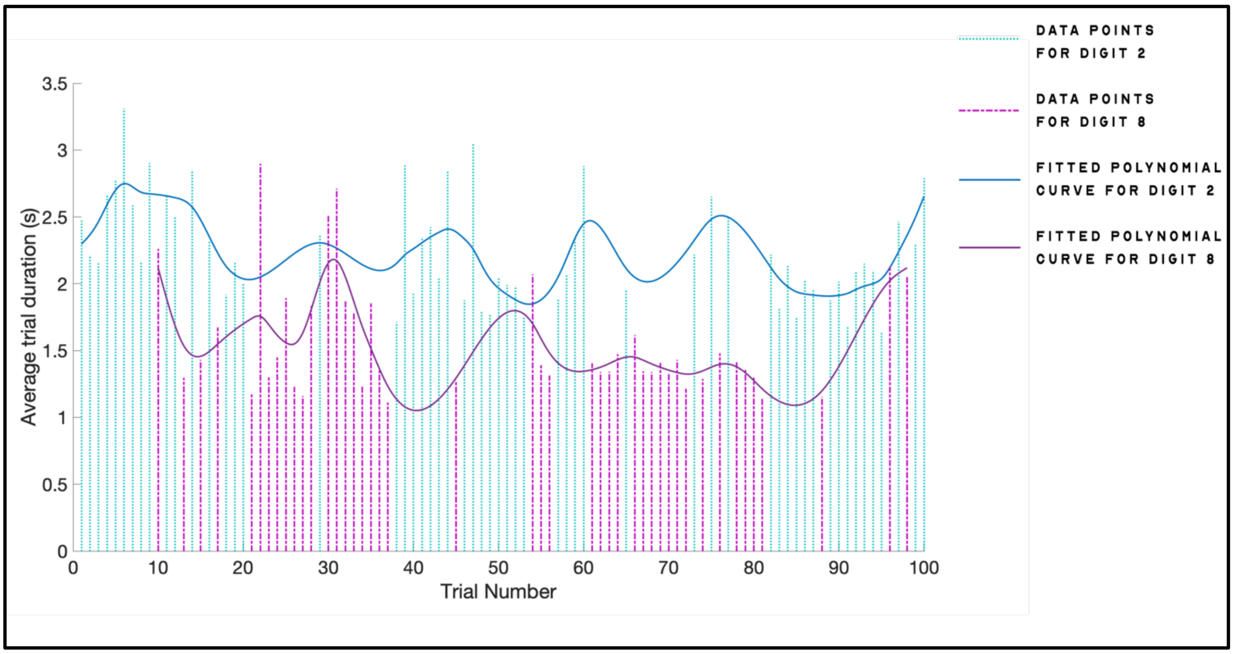


Figure 19. Average trial duration for all 28 participants in task 2. The trials displaying the digit 2 are shown in blue, and those displaying the digit 8 are in magenta, with their respective fitted polynomial curves.

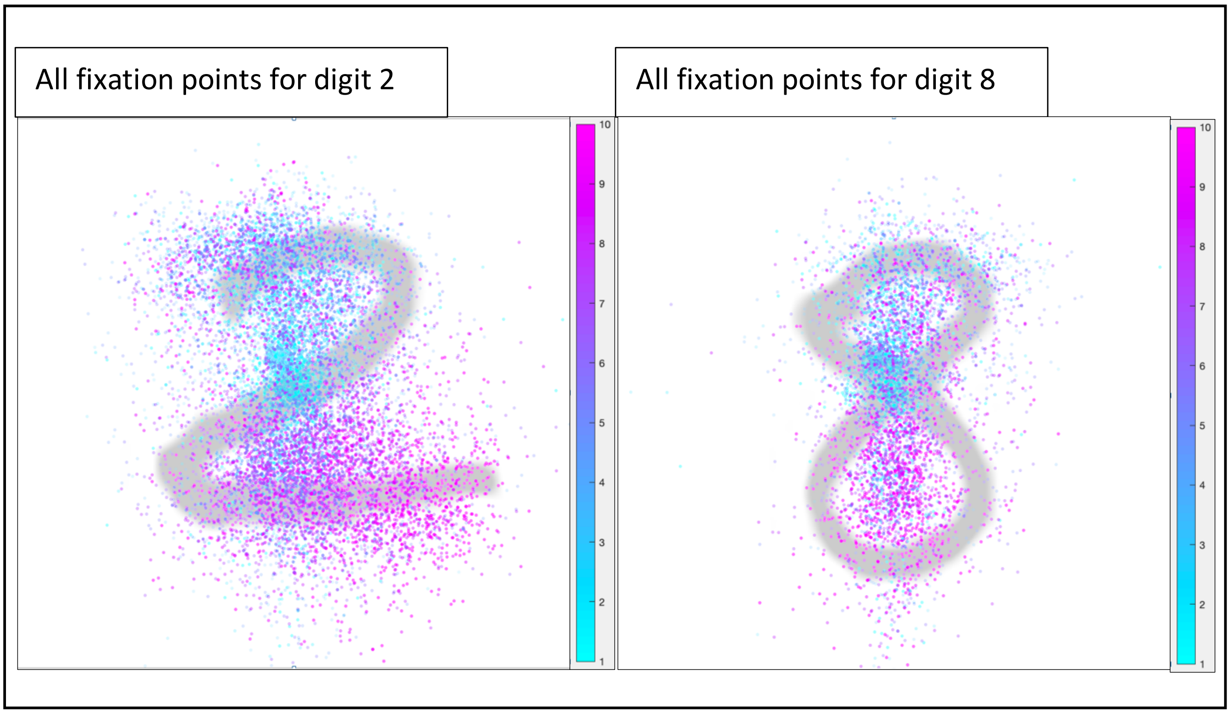


Figure 20. All fixation locations for digits 2 and 8 in task two for all 28 participants. The colour bar maps from cyan (1st fixation) to magenta (last fixation) of each trial.

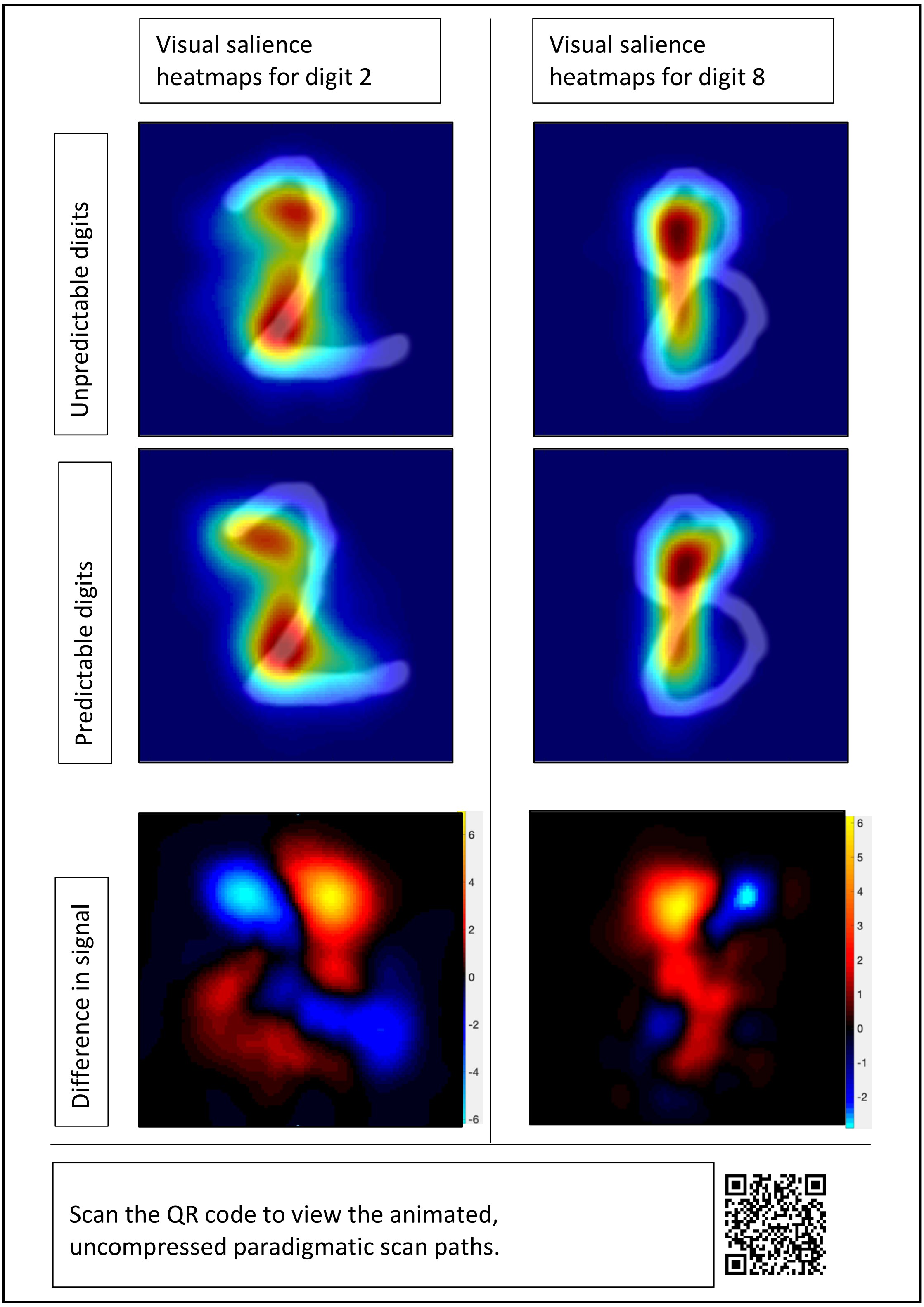
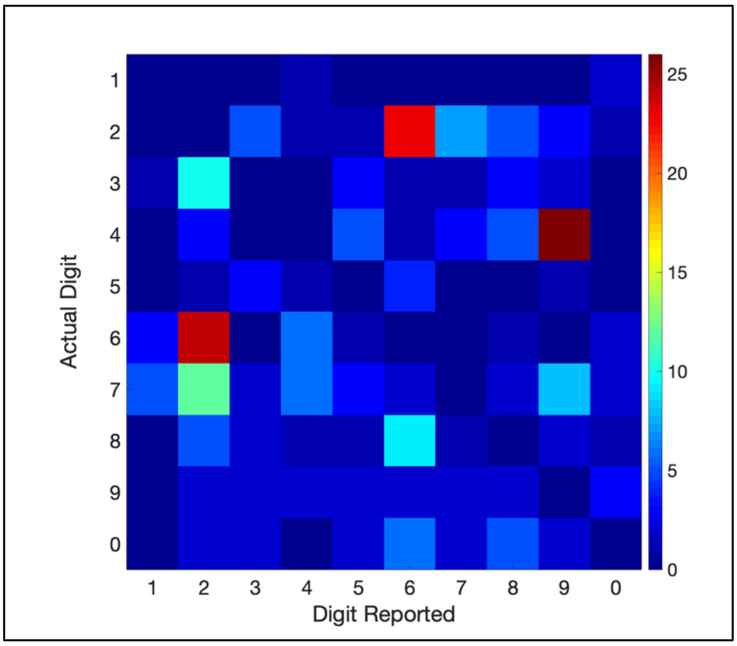


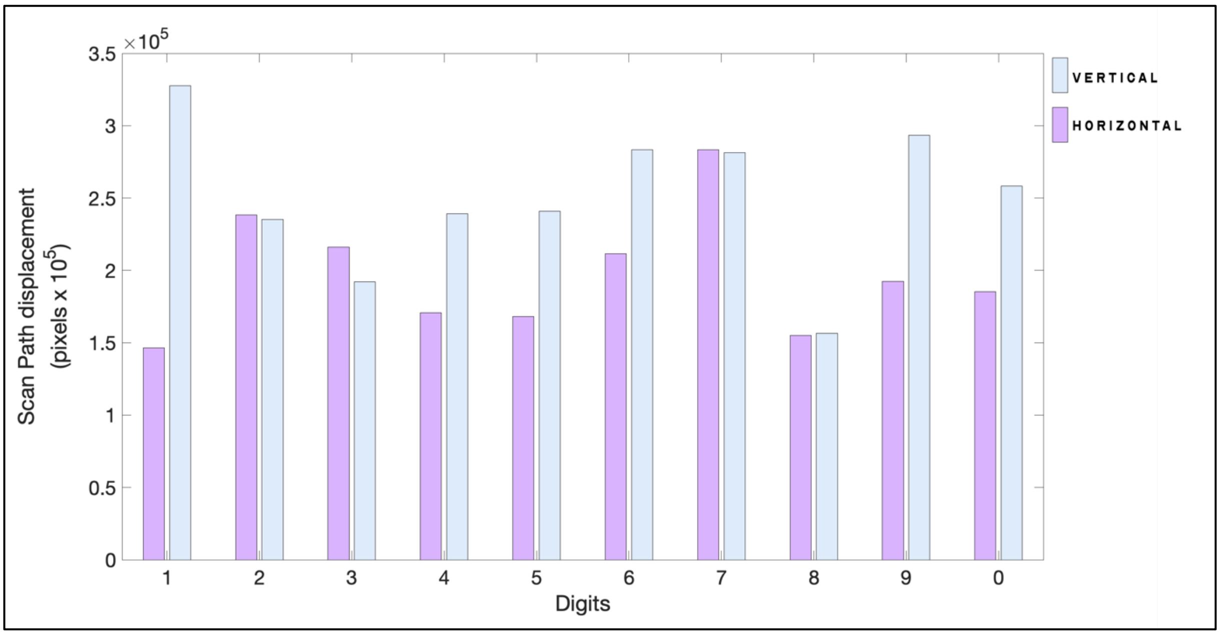
Figure 21. Compressed paradigmatic scan paths for predictable and unpredictable 2's and 8's in task 2. The difference in signal images represent increased probability of gaze location in unpredictable digits (hot colours) and predictable digits (cold colours)[5].

[5] QR code URL: https://drive.google.com/open?id=1Gp6lyt7faLj8mlb_RcHL-mHMfnkkCU8t

Task 3:



<u>Figure 22.</u> Error matrix based on responses given by all 28 participants during task 3. Digits 2, 4, 6 and 7 were most often mistaken for other digits. The colour bar represents the number of errors occurring for a reported digit/ actual digit combination.



<u>Figure 23.</u> Average for all 28 participants of horizontal and vertical gaze location displacement for each digit in task 3. Digits 2, 3, and 7 were the only ones to have equal/greater horizontal than vertical displacement.

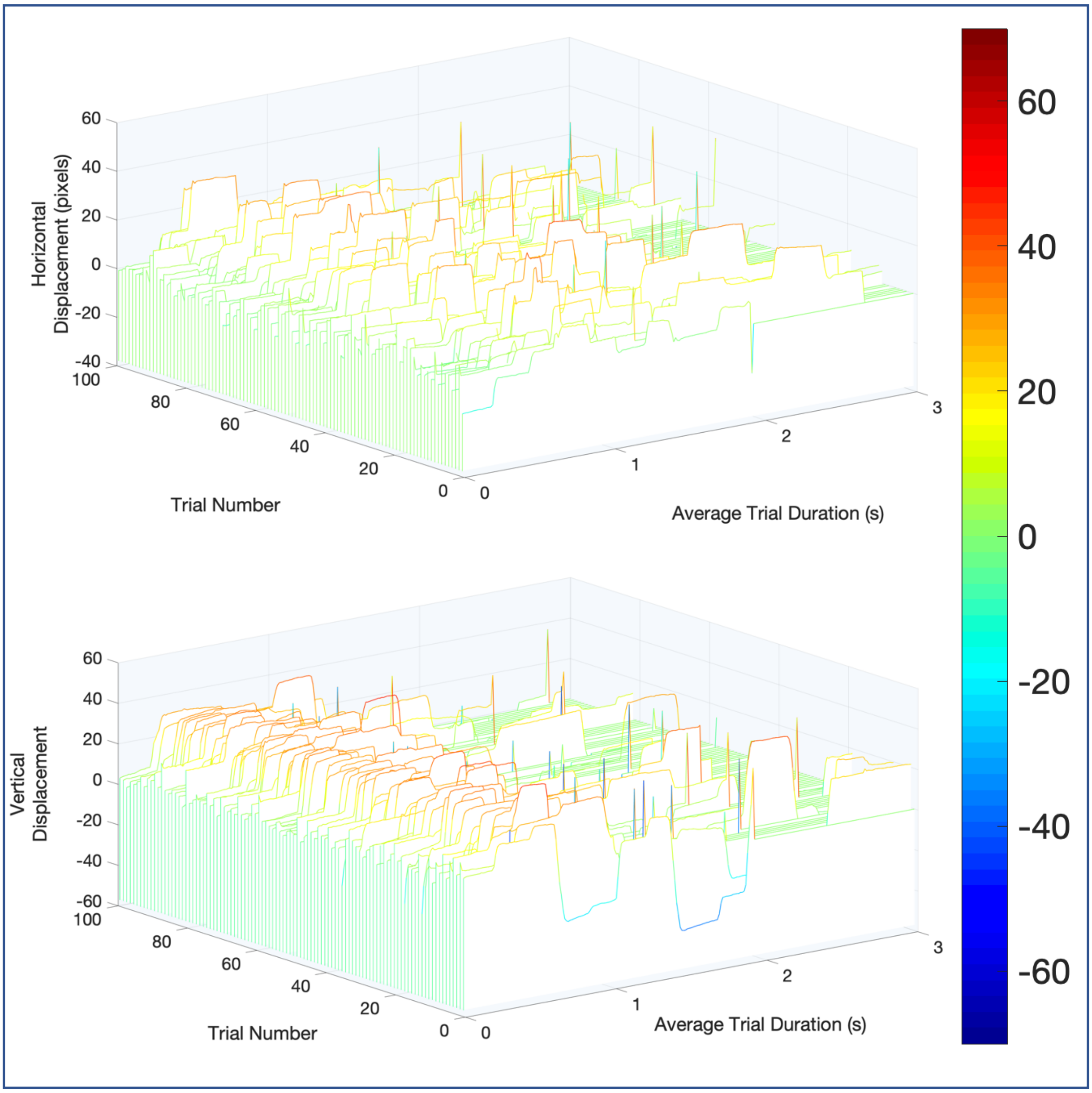


Figure 24. Top: Average for all 28 participants of horizontal displacement for each trial (maximum trial duration: 3(s)). Displacement is spread out and heterogenous between trials. Bottom: Average for all 28 participants of vertical displacement for each trial (maximum trial duration: 3(s)). Most displacement happens in the first second of the trial (an increase in vertical location, i.e. towards the top of the display) and is homogenous across all trials.

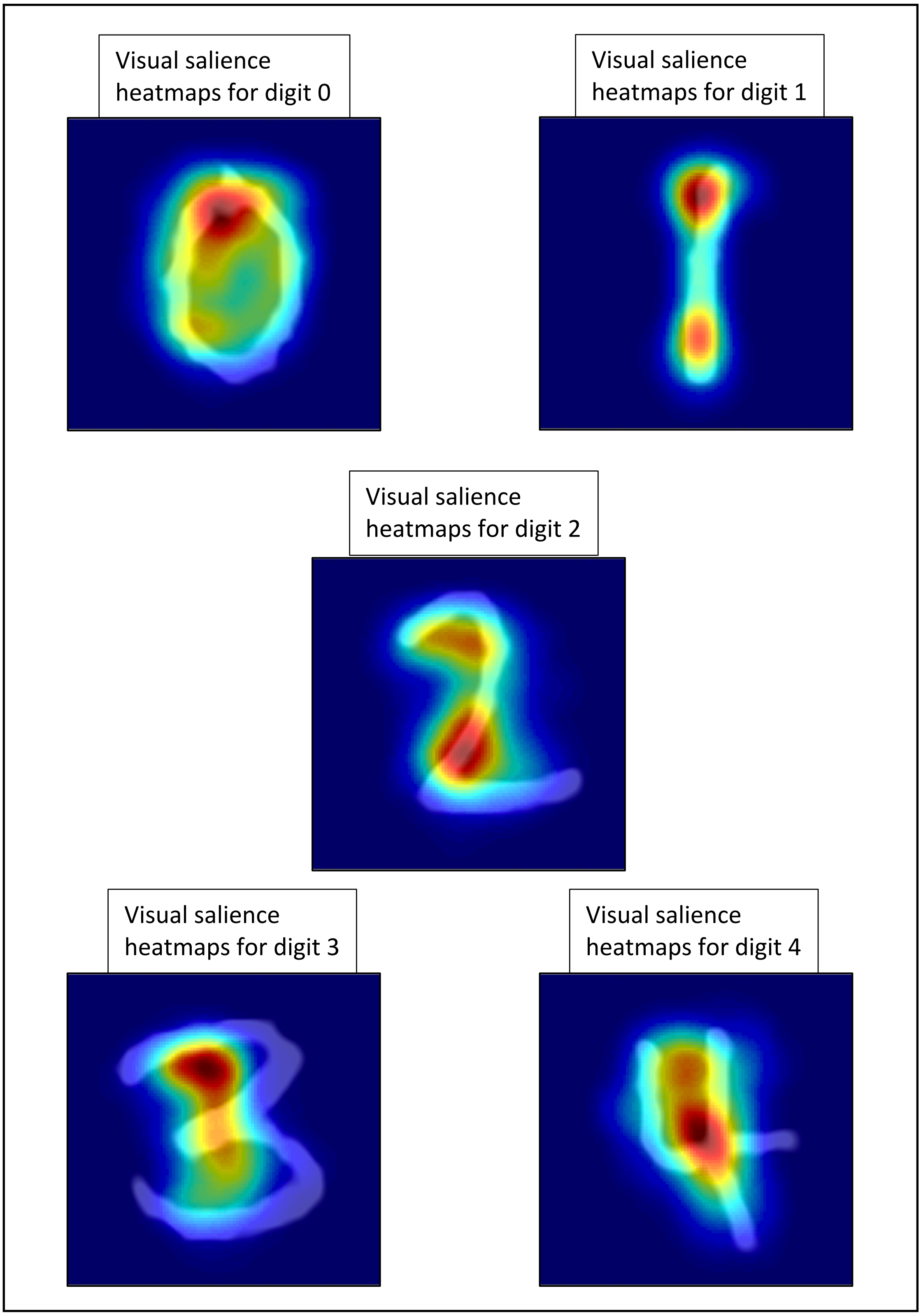


Figure 25. Compressed paradigmatic scan paths for digits 0, 1, 2, 3 and 4.

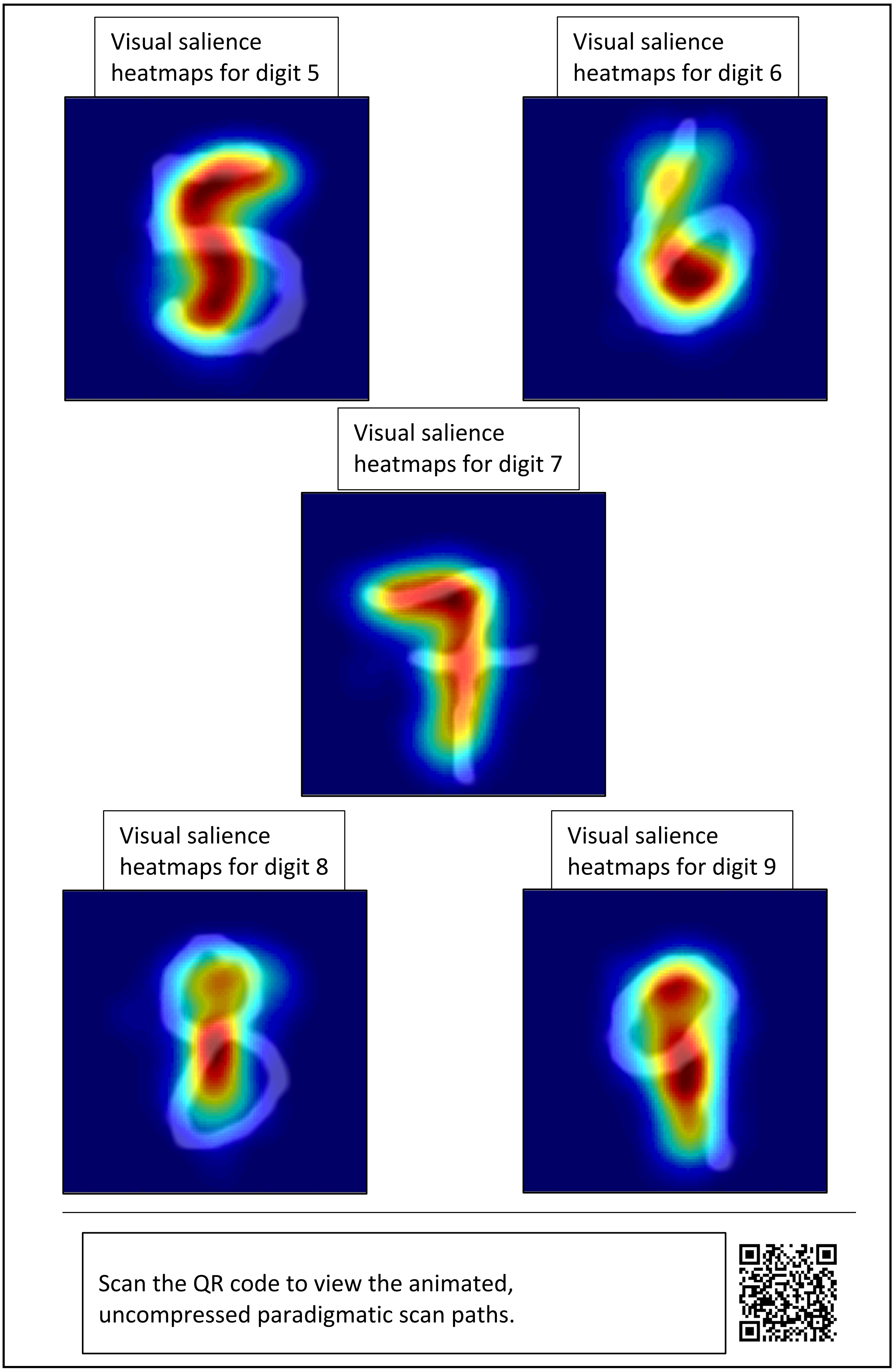


Figure 26. Compressed paradigmatic scan paths for digits 5, 6, 7, 8 and 9.[6].

[6] QR code URL: https://drive.google.com/open?id=1khLccomfFuFZ6GekegoJ7NhBzkjvyzV6

MDP model:

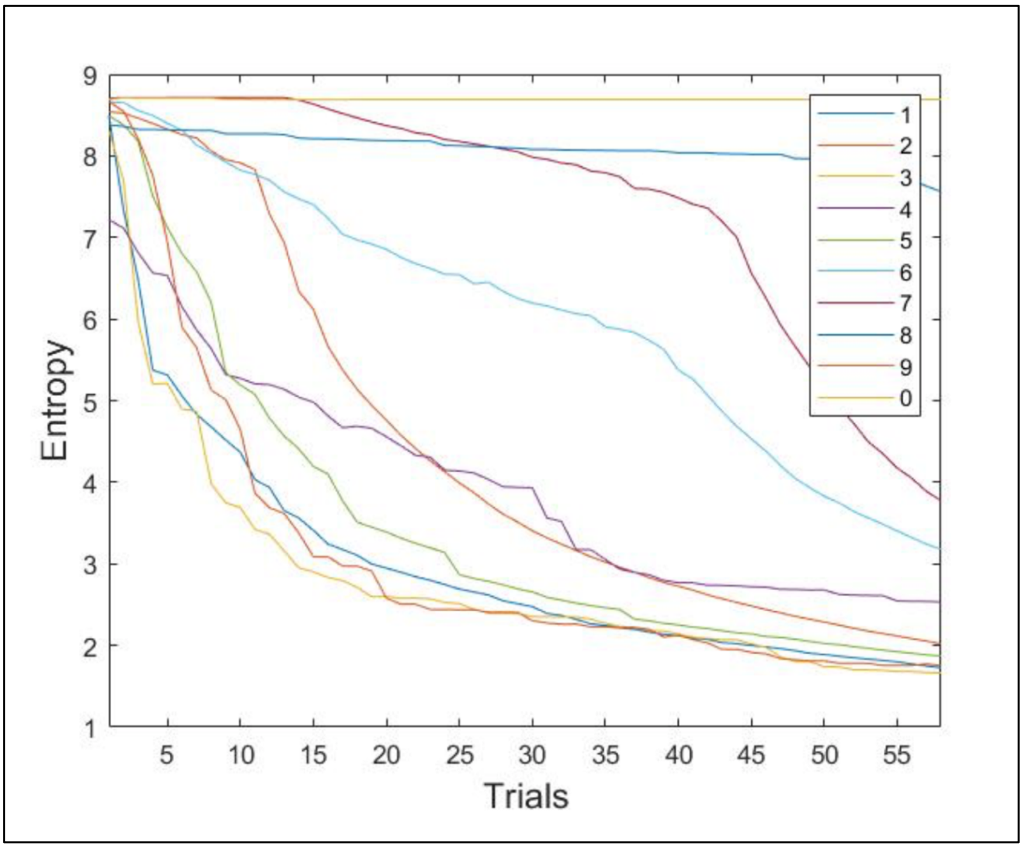


Figure 27. This figure illustrates the summed entropy (uncertainty) over the probability distributions encoded in the columns of learned likelihood matrices for each digit over 58 exposures. As the model learns the likelihood matrices, the entropy declines for most digits.

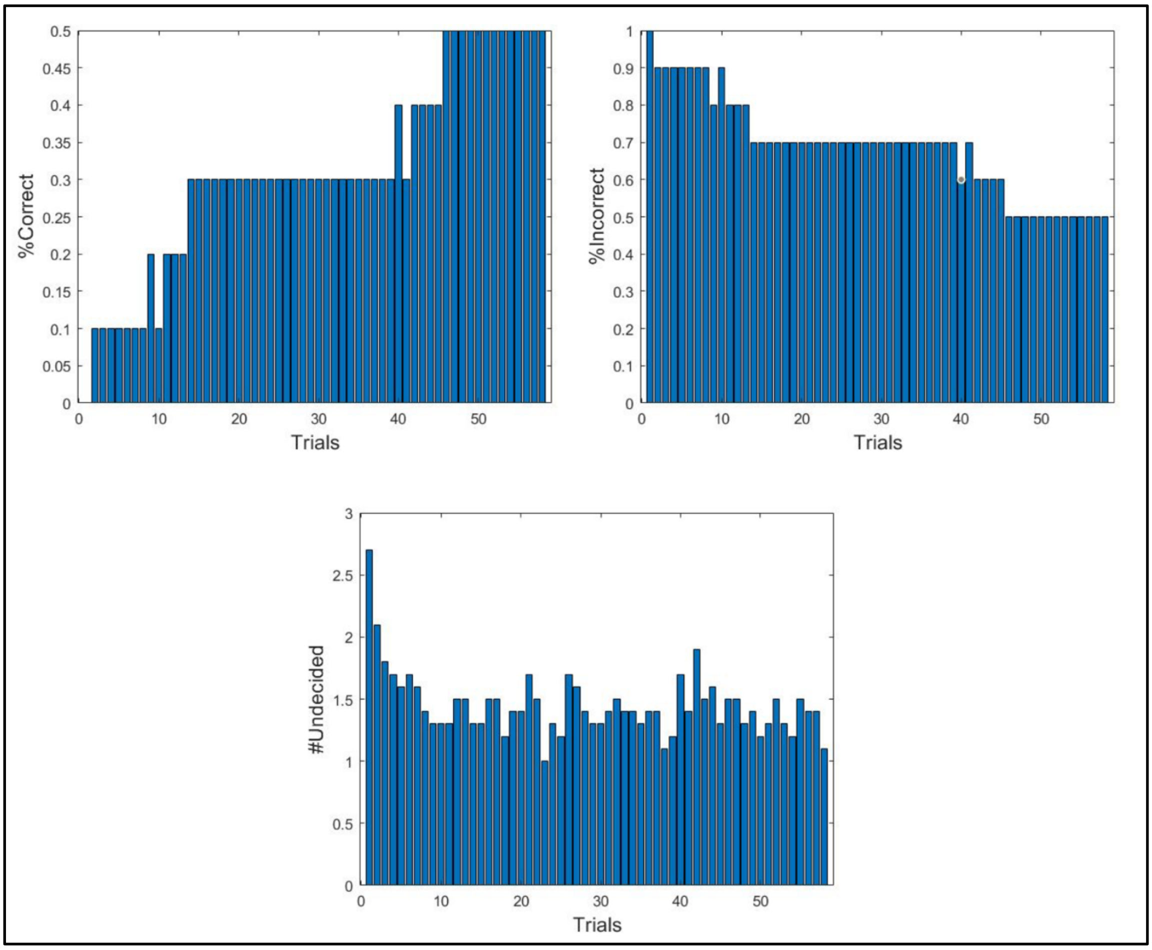


Figure 28. The panels on the top left and right show the percentage correct and incorrect digit identifications over 58 exposures to digits. The panel at the bottom shows the number of times the model was undecided about reporting its beliefs about digits.

Discussion

Results Summary & discussion

Overall results:

Our overall findings show a significant difference between all three tasks both in trial duration and response accuracy. Task 2 (volatility) was the shortest with an average trial duration of 1.95s (SD = 0.98 s) and the one with the highest accuracy score with an average of 89.11% (SD = 31.18%) (see Table 1). These results imply this task was easiest to perform and yielded the best scores as a result of this ease in performance. A probable cause for this lies in the nature of the task, which presented only two digits (2's and 8's). While participants were not made aware of this, they showed rapid understanding of the limited pattern they were being exposed to. Task 1 (1 – 3 – 5 sequence) was the task participants performed least well on and showed the longest average trial duration. This is probably due to an overall learning effect that can be observed. As the task progressed, trial durations shortened (see Task 1 section below). An increase in average trial duration and a decrease in performance between Task 2 and Task 3 is indicative of a harder to accomplish task, for Task 3. This result was expected, as no learned sequence or pattern could be inferred by participants, and each digit had an equal probability of being displayed.

We would expect our MDP model to perform similarly in terms of accuracy and trial duration. As expectancy of a digit's identity grows, trial duration decreases (expectancy is shown to be highest in task 2 from our behavioural data). However, the better performance, with regards to trial duration, on task 3 compared to task 1 does not comply with this hypothesis. We may suggest trial duration decreased in task 3 as a global better understanding of the general paradigm. We may infer a better performance would be observed if participants were to redo task 1 at the end of the session.

Task 1 (1 – 3 – 5 sequence):

In task 1 (1 – 3 – 5 sequence), we attempted to test two main hypotheses. Firstly, trial durations and active sampling decrease as a learning effect is observed. Secondly, performance (accuracy of identification) increases on digits that have a higher likelihood of being presented. In Figure 18, we compare the canonical scan paths of the first ten 1's and 3's with their corresponding last 10 digits. This effectively tests the former hypothesis, by showing a decrease in scene exploration and trial duration (decrease in the number of fixations) between both groups. This supports the hypothesis that a learning effect due to relying on prior beliefs has caused participants to speed up in their identification of each digit. This is true for both digits 1 and 3.

While the first hypothesis is supported by our empirical findings, the second hypothesis, which states that we would expect an increase in performance on digits that have a higher likelihood of appearing (in the case of this task, digit 5 was presented 70% of the time in the third position of each sequence, and another random digit was presented the remaining 30% of time) was not supported by our findings. These unexpected results show a decrease in performance and an increase in trial duration when the digit 5 was shown, which goes against our expected outcome, whereby the more probable digit would elicit better results. One reason for this unexpected observation is that we may suggest it is due to the digit 5 being a harder-than-average digit to identify. We base this suggestion on the results seen in the video of task 3. The digit five appears to have a longer lingering signal (average trial duration). To resolve this unexpected result, we propose performing this task with an easier-to identify digit within the sequence (such as 8) might yield results that better test our hypothesis.

Task 2 (volatility):

The main hypothesis tested in task 2 is that trial duration and scene exploration will decrease when the probability of a digit being shown increases. We may attempt to test this hypothesis by comparing the digits shown in the high probability blocks with those shown in the low-probability blocks. The probability of 2's appearing is 80% in three of the five blocks, and 20% in the other two (these probabilistic distributions alternate, see Figure 10). Figure 21 compares the salience heatmaps (canonical gaze locations) for predictable and unpredictable digits. Here, we expect greater signal in the unpredictable digits, as surprise (i.e. uncertainty) would increase, leading to more hesitation in decision-making. This directly translates to longer trial durations and active sampling, to reduce uncertainty before making a decision. This, again, is due to the agent's fundamental expectancy to receive "correct" feedback. This is programmed in the MDP model and is one of two set-out goals for participants (the other being to press the space bar as soon as they have identified the digit). The video linked in Figure 21 and the difference in signal images testify that trial durations do not seem to vary between predictable and unpredictable digits. On the other hand, the foraging behaviour (i.e. active sampling) appears to increase, implying more exploration is needed to confirm a digit's identity. This supports our hypothesis that when the likelihood of seeing is digit is lower, information-gathering becomes more important to correctly assess our belief about the nature of our environment. One interesting feature of the digit 2 that appears in the predictable patterns, is the recurring "edge seeking" behaviour. In these trials, participants displayed limited overall exploration, but distinct saccades to both ends of the digit (high consistency in scan paths). This is further evidence of a learning effect, as participants attempt to distinguish the digit 2 by looking for edges, whereas the digit 8 has none.

The fitted curves to the average trial durations shown in Figure 19 are evidence that the digit 8 was, both in low likelihood and high likelihood blocks, easier to identify than the digit 2. This is not indicative of the accuracy of responses, but simply points towards less exploration being required for an agent to create a belief about the identity of the digit being assessed. Although a direct comparison between 2's and 8's is difficult as they were not shown the same number of times, trial durations involving the digit 2 were consistently longer than for the digit 8. This observation is further outlined in Figure 20, which shows all fixation locations for all participants in digits 2 and 8. Here we see that digit exhibit more spread in fixation locations throughout a trial, as evidence that an increase foraging behaviour is necessary for decision-making about the nature of the scene's identity. Moreover, the digit 8, with a much more condensed pattern of fixation locations (i.e. low exploration required) has few dots in the magenta colours (as per the colour bar), indicating more trials ended before reaching 10 fixations than for the digit 2. This means less active sampling is required for an agent to come to the belief about the digit 8's identity than for the digit 2. Figure 20 therefore informs us not only on the foraging behaviour elicited by the digit, but also the lesser number of saccades required to identify the 8's than the 2's.

Task 3 (random):

By randomly presenting digits with equal probability in task 3, we are able to reveal intrinsically distinct identification features for each digit. In Figure 25 and Figure 26, we notice that certain canonical gaze locations (or compressed scan paths) bear great resemblance to their corresponding digit. This is notably the case for digits 0, 5 and 7. This becomes even more interesting when looking at the error matrix in Figure 22 as these digits, on average, have low error rates compared to other digits. One exception is the high number of reported 2’s when the actual digit shown was 7. This may, however, be explained by the resemblance in the upper portion of the digits 7 and 2, which may have led to confusion. Despite this, the increased amount of visual exploration, matching the shape of the digit, appears to have had the effect of increasing certainty, which resulted in lower errors. One notable error is 2’s and 6’s, which were often mistaken for one-another. Participants reported this was due to many 2’s containing a lower “loop”, which looked like the loop of a 6. Referring back to scan paths, we can also notice a slight resemblance in the scan paths of the digits 6 and 2. Perhaps by comparing the resemblance of these scan paths, we may further predict how likely it is for two digits being mistaken for one-another. This gives rise to a new research question: could matching canonical scan paths (i.e. active sampling patterns) and trial durations lead to equal confusions for distinct scene identities? Another notable feature of these figures is the salience found in the ends of shapes. All shapes that have ends display clear concentrations of gaze fixations at these ends, indicating the extrinsic value these locations may bring an agent (Mirza *et al*., 2018). Digit 0 shows how a lack of edge leads to an equal probability of the gaze appearing anywhere on the shape, while digit 8 shows clear evidence of salience in the crossing of the upper and lower loops.

To further analyse how gaze location salience may lead to gaze displacement (saccading), Figure 23 shows that most salient visual information, containing high epistemic value, is in essence attained vertically. One fascinating finding is that two of the most similar digits, 1 and 7, are also on either end of the spectrum for average horizontal displacement. While digit 1 is the digit that on average expressed the least horizontal exploration, digit 7 is the one that yielded the most. One suggestion for this observation would be an initial belief that the digit is 1, as would be supported by our MDP model when witnessing initial resembling features. Upon sampling the top portion of the digit, the salient "edge" is not detected, causing a sudden avid search for this salient feature. This leads to a horizontal displacement to find the end of the digit 7. This theory should be supported by a noticeable increase in trial durations between 7's and 1's. While this may be indirectly inferred by looking Figure 23, this would be an incomplete conclusion as this Figure does not take into account fixation duration, which may increase trial duration, in addition to scan path length. Further analysis is required to determine average trial durations per digit.

Figure 24 shows how each trial systematically starts with a strong vertical movement at around 200ms (bottom graph). This is akin to an event-related potential (ERP) such as the decision-making P300 seen in electroencephalography (EEG) data. The homogeneity in this response appears identity-unspecific, meaning this behaviour appears to favour uncertainty-resolving *de facto*, before and agent has even had time to process a belief about what the digit may be. The upper graph in Figure 24 suggests a decision to explore horizontally is more digit-specific (heterogenous response), once a prior belief about the identity is made. These policy selections are then used to further support evidence of the agent's prior belief before resolving enough uncertainty to make a decision about the identity observed. These results support the active inference hypothesis outline above, while bringing further insight into the initial salient features an agent seeks out for belief updating about the hidden states it is actively sampling.

MDP model:

The results from figure 27 show a learning effect in the model for how the features of digits should be used to categorise them. These results may help us determine how difficulty in identifying digits may cause misidentification. Further testing and analysis will alow, once the model is trained, to compare its performance directly to the results obtained from our behavioural data.

Limitations

Despite all stops being pulled to ensure the highest degree of accuracy in gaze-tracking, occasional head motion and eye shape variability caused certain participants displaying difficulty in accurately reporting responses while on the digit dial screen (for digit identity decision reporting). The gaze-contingent nature of this paradigm underlines how crucial calibration was to minimize the frustration and discomfort of participants. While participants were shown an “incorrect” message if the accurate number was not selected by their gaze, they were assured that unless the error lied in misidentifying a digit, the incorrect feedback would not be tallied as such. The feedback would be flagged as an error code which would then be matched to the closest digit in the dial. This was in an attempt to not lose their focus and keep them interested in trying to answer accurately and as soon as they had identified the digit. Certain responses were therefore not included in accuracy calculations.

One demographic limitation in this study was a clear bias towards females, as finding male participants proved to be challenging. Although performance did not appear to vary between males and females, our sample population misrepresents the general population at this level.

One bias in the paradigm’s design lies in the fact that all tasks started with a fixation cross located in the center of the screen, as is customary, but with the center of the scene of interest aligned behind it. Although displacing the digits in one task was considered (translational variance), this option was discarded as it would have introduced incomparable results between tasks in terms of trial duration. Time would have been spent looking for the digit before the agent starts creating beliefs about identify it gathered through active sampling. One alternative idea for a future study would be to keep the digits placed in the center of the screen, but to move the fixation cross around. This would not only allow to explore the scene differently but may also bring further evidence or counter our findings from task 3 that state that the most salient *de facto* information in identifying a digit may be found by saccading about 60 pixels upwards. Perhaps this may just be a reflexive reaction to a trial starting, although this suggestion seems unlikely given the significant occurrence in displaying this behaviour across participants. Furthermore, as saccades are controlled by the FEF, which exhibits reaction times (RT) of less than 30ms (Kirchner *et al*., 2009), these initial movements are unlikely linked to such potentiation.

Future Research

While our MDP model is showing promising results for scene identification based on prior beliefs and active sampling, it remains in its early stages. We plan to further develop it to accurately perform all tasks performed by participants and to produce comparable results. As it is built on the active inference framework, should the results closely match those obtained behaviourally, this would bring further support to human vision being driven by prior beliefs and perception. This would indicate a top-down functional network for visual active sampling. Moreover, it would indicate evidence of a generative model being updated constantly with regards to its beliefs about its environment.

We hope to follow up on this study with patients suffering from schizophrenia, as to further test the theory of aberrant salience. Aberrant salience is a visual search for irrelevant cues to solve a problem (reward is illogically obtained) (Roiser *et al.*, 2009). In our case, we would expect uncorrelated canonical scan paths to those obtained with healthy participants, and perhaps less correlation between patients than between healthy participants (increased within group variability in patients) (Howes & Nour, 2016). Further research involving autism may be of valuable interest, as a 2017 study found that adults with autism expressed over-expectancy in volatility, thereby lowering their prediction mapping of the likelihood of an event occurring (Lawson *et al.*, 2017). We would therefore expect task 2, where participants were assessed in five blocks on either 2's or 8's, to show lesser difference between predictable and unpredictable blocks of digits, in contrast to the result obtained in this study.

Conclusion

In conclusion, our behavioural findings showed consistency and supported the active inference framework for visual active sampling with regards to scene identification. While certain unexpected results were obtained, they may possibly be explained by an adjustment in the visual paradigm being required. Active inference as a framework was further tested by creating an MDP model under its assertions. Despite this MDP model still requiring further development, preliminary results display learning patterns expected under the assumption of scene exploration based on prior beliefs (top-down active scene exploration). We will continue to develop the MDP model and reevaluate our findings in the coming weeks. With our current findings, we can infer that human vision is sparse, deliberative and predictive. Finally, our findings support a top-down connectivity in human vision and exhibit active exploration behaviours for decision-making based on updated prior beliefs.

References

Behrens, T. E. J., Woolrich, M. W., Walton, M. E., & Rushworth, M. F. S. (2007). Learning the value of information in an uncertain world. *Nature Neuroscience*, *10*, 1214. Retrieved from https://doi.org/10.1038/nn1954

Bellman, R. (1957). A Markovian Decision Process. *Journal of Mathematics and Mechanics*, *6*(5), 679–684. Retrieved from http://www.jstor.org/stable/24900506

Bullier, J. (2001). Integrated model of visual processing. *Brain Research Reviews*, *36*(2), 96–107. https://doi.org/https://doi.org/10.1016/S0165-0173(01)00085-6

Cox, R. T. (1946). Probability, Frequency and Reasonable Expectation. *American Journal of Physics*, *14*(1), 1–13. https://doi.org/10.1119/1.1990764

Ernst, M. O., & Banks, M. S. (2002). Humans integrate visual and haptic information in a statistically optimal fashion. *Nature*, *415*(6870), 429–433. https://doi.org/10.1038/415429a

Friston, K. (2010). The free-energy principle: a unified brain theory? *Nature Reviews Neuroscience*, *11*, 127. Retrieved from https://doi.org/10.1038/nrn2787

Friston, K. (2011). What is optimal about motor control? *Neuron*, *72*(3), 488–498. https://doi.org/10.1016/j.neuron.2011.10.018

Friston, K. (2013). Life as we know it. *Journal of The Royal Society Interface*, *10*(86), 20130475. https://doi.org/10.1098/rsif.2013.0475

Friston, K., FitzGerald, T., Rigoli, F., Schwartenbeck, P., O'Doherty, J., & Pezzulo, G. (2016). Active inference and learning. *Neuroscience and Biobehavioral Reviews*, *68*, 862–879. https://doi.org/10.1016/j.neubiorev.2016.06.022

Gavornik, J. P., & Bear, M. F. (2014). Learned spatiotemporal sequence recognition and prediction in primary visual cortex. *Nature Neuroscience*, *17*, 732. Retrieved from https://doi.org/10.1038/nn.3683

Grosbras, M.-H., & Paus, T. (2002). Transcranial Magnetic Stimulation of the Human Frontal Eye Field: Effects on Visual Perception and Attention. *Journal of Cognitive Neuroscience*, *14*(7), 1109–1120. https://doi.org/10.1162/089892902320474553

Howes, O. D., & Nour, M. M. (2016). Dopamine and the aberrant salience hypothesis of schizophrenia. *World Psychiatry : Official Journal of the World Psychiatric Association (WPA)*, *15*(1), 3–4. https://doi.org/10.1002/wps.20276

Huang, G., Huang, G.-B., Song, S., & You, K. (2015). Trends in extreme learning machines: A review. *Neural Networks*, *61*, 32–48. https://doi.org/https://doi.org/10.1016/j.neunet.2014.10.001

Itti, L., & Koch, C. (2001). Computational modelling of visual attention. *Nature Reviews Neuroscience*, *2*(3), 194–203. https://doi.org/10.1038/35058500

Joyce, J. (2019). Baye's Theorem. In *The Stanford Encyclopedia of Philosophy* (Spring 201). Metaphysics Research Lab, Stanford University. Retrieved from https://plato.stanford.edu/archives/spr2019/entries/bayes-theorem/

Kirchner, H., Barbeau, E. J., Thorpe, S. J., Régis, J., & Liégeois-Chauvel, C. (2009). Ultra-Rapid Sensory Responses in the Human Frontal Eye Field Region. *The Journal of Neuroscience*, *29*(23), 7599 LP – 7606. https://doi.org/10.1523/JNEUROSCI.1233-09.2009

Knill, D. C., & Pouget, A. (2004). The Bayesian brain: the role of uncertainty in neural coding and computation. *Trends in Neurosciences*, *27*(12), 712–719. https://doi.org/https://doi.org/10.1016/j.tins.2004.10.007

Körding, K. P., & Wolpert, D. M. (2004). Bayesian integration in sensorimotor learning. *Nature*, *427*(6971), 244–247. https://doi.org/10.1038/nature02169

Lawson, R. P., Mathys, C., & Rees, G. (2017). Adults with autism overestimate the volatility of the sensory environment. *Nature Neuroscience*, *20*, 1293. Retrieved from https://doi.org/10.1038/nn.4615

Lecun, Y., Bottou, L., Bengio, Y., & Ha, P. (1998). GradientBased Learning Applied to Document Recognition. *Proceedings of the IEEE*, (November), 1–46. https://doi.org/10.1109/5.726791

McCoy, A. N., Crowley, J. C., Haghighian, G., Dean, H. L., & Platt, M. L. (2003). Saccade Reward Signals in Posterior Cingulate Cortex. *Neuron*, *40*(5), 1031–1040. https://doi.org/10.1016/S0896-6273(03)00719-0

Mirza, M. B., Adams, R. A., Mathys, C. D., & Friston, K. J. (2016). Scene Construction, Visual Foraging, and Active Inference. *Frontiers in Computational Neuroscience*, *10*, 56. https://doi.org/10.3389/fncom.2016.00056

Mirza, M. B., Adams, R. A., Mathys, C., & Friston, K. J. (2018). Human visual exploration reduces uncertainty about the sensed world. *PLOS ONE*, *13*(1), e0190429. Retrieved from https://doi.org/10.1371/journal.pone.0190429

Mishkin, M., Ungerleider, L. G., & Macko, K. A. (1983). Object vision and spatial vision: two cortical pathways. *Trends in Neurosciences*, *6*, 414–417. https://doi.org/https://doi.org/10.1016/0166-2236(83)90190-X

Montague, P. R., Dolan, R. J., Friston, K. J., & Dayan, P. (2012). Computational psychiatry. *Trends in Cognitive Sciences*, *16*(1), 72–80. https://doi.org/https://doi.org/10.1016/j.tics.2011.11.018

Oliver, N. M., Rosario, B., & Pentland, A. P. (2000). A Bayesian computer vision system for modeling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(8), 831–843. https://doi.org/10.1109/34.868684

Pezzulo, G., Rigoli, F., & Friston, K. (2015). Active Inference, homeostatic regulation and adaptive behavioural control. *Progress in Neurobiology*, *134*, 17–35. https://doi.org/10.1016/j.pneurobio.2015.09.001

Posner, M. I., & Petersen, S. E. (1990). The Attention System of the Human Brain. *Annual Review of Neuroscience*, *13*(1), 25–42. https://doi.org/10.1146/annurev.ne.13.030190.000325

Rawat, W., & Wang, Z. (2017). Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review. *Neural Computation*, *29*(9), 2352–2449. https://doi.org/10.1162/neco_a_00990

Roiser, J. P., Stephan, K. E., den Ouden, H. E. M., Barnes, T. R. E., Friston, K. J., & Joyce, E. M. (2009). Do patients with schizophrenia exhibit aberrant salience? *Psychological Medicine*, *39*(2), 199–209. https://doi.org/DOI: 10.1017/S0033291708003863

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, *61*, 85–117. https://doi.org/https://doi.org/10.1016/j.neunet.2014.09.003

Thorpe, S., Fize, D., & Marlot, C. (1996). Speed of processing in the human visual system. *Nature*, *381*(6582), 520–522. https://doi.org/10.1038/381520a0

Turk-Browne, N. B., Jungé, J. A., & Scholl, B. J. (2005). The Automaticity of Visual Statistical Learning. *Journal of Experimental Psychology: General*. Turk-Browne, Nicholas B.: Department of Psychology, Yale University, Box 208205, New Haven, CT, US, 06520-8205, nicholas.turk-browne@yale.edu: American Psychological Association. https://doi.org/10.1037/0096-3445.134.4.552

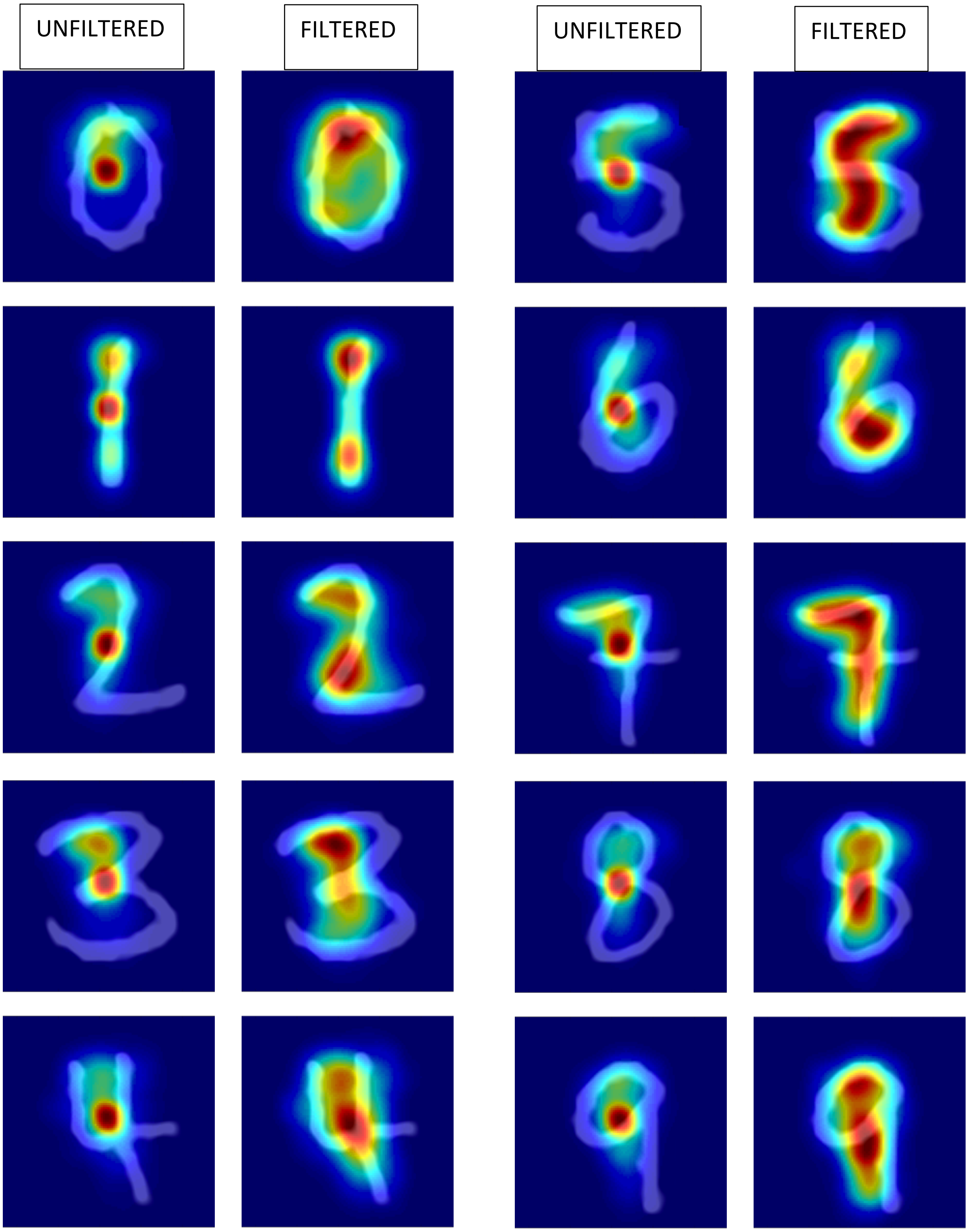
Ungerleider, L. G., & Haxby, J. V. (1994). ‘What’ and ‘where’ in the human brain. *Current Opinion in Neurobiology*, *4*(2), 157–165. https://doi.org/https://doi.org/10.1016/0959-4388(94)90066-3

Yu, A. J., & Dayan, P. (2005). Uncertainty, Neuromodulation, and Attention. *Neuron*, *46*(4), 681–692. https://doi.org/10.1016/j.neuron.2005.04.026

This experiment was realised using Cogent 2000 developed by the Cogent 2000 team at the FIL and the ICN and Cogent Graphics developed by John Romaya at the LON at the Wellcome Department of Imaging Neuroscience.

Appendices

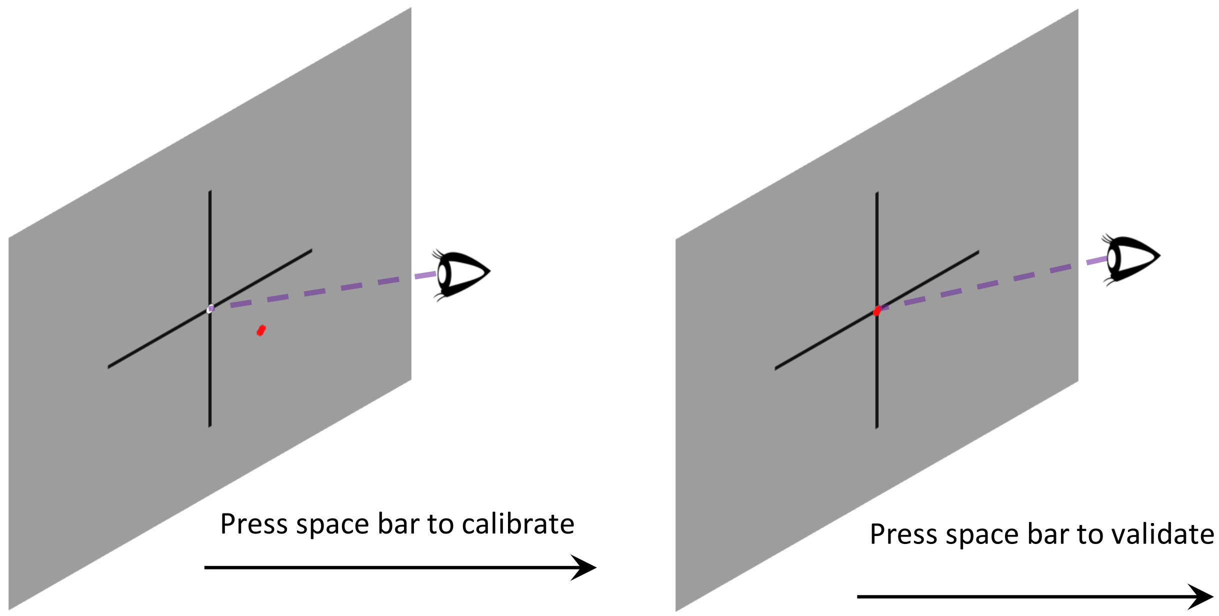
I would like to start these appendices with examples from the results of the compressed canonical scan paths where the initial fixation point in the center of the display is not removed, compared to the filtered render (as seen in the results section):



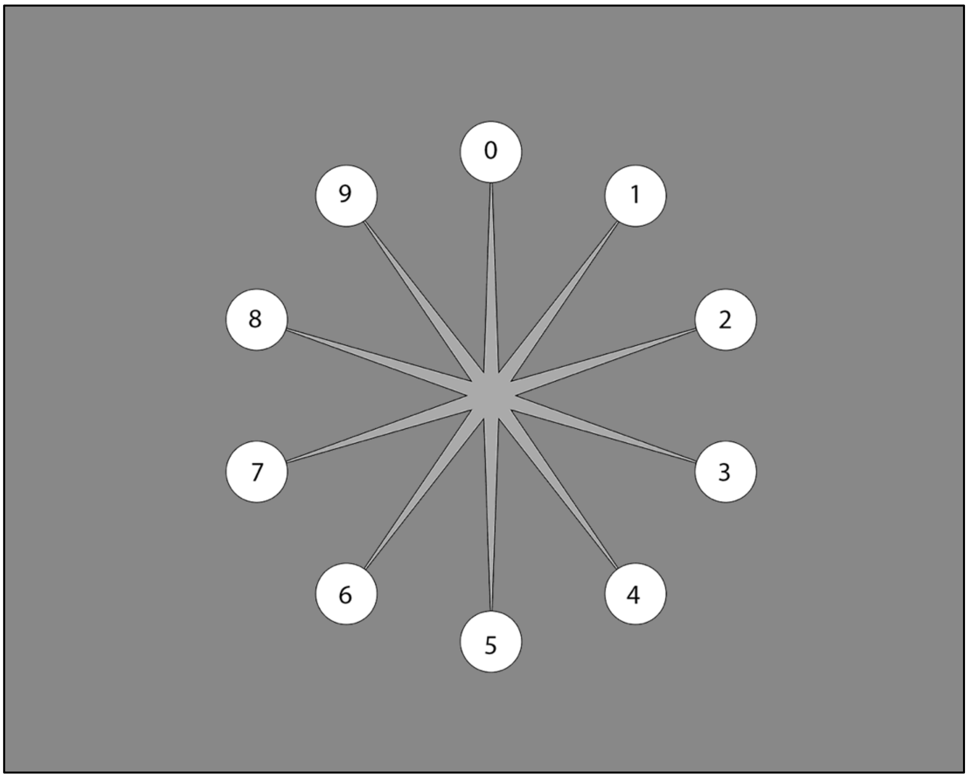
In some instances, the built-in calibration system provided by EyeLink 1000 proved inefficient in accurately locating a participant's gaze, although movements were consistent. To overcome this, I scripted a short matlab function using Cogent to add an X and Y correction value to displayed gaze location, to accommodate for participant's comfort in completing tasks. Although this was prepared prior to data collection, it was only used on two occasions. The following is a short description of this online calibration function.

A simple fixation cross will be displayed on the screen. A red dot represents the location the eye-tracker is interpreting the gaze being located at. Whilst participants are staring at the centre of the fixation cross, this dot may appear to be slightly off-centered.

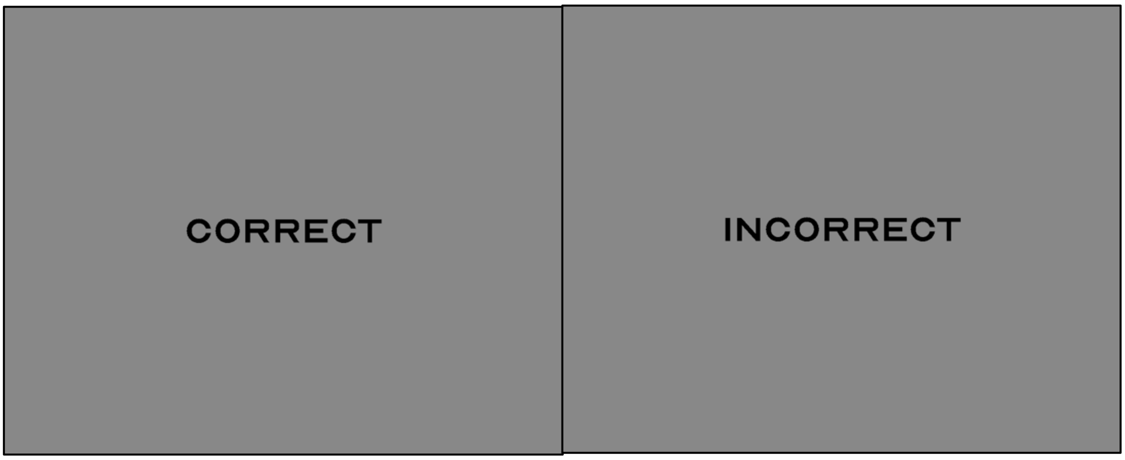
By pressing the space bar once while staring at the center of the fixation cross, this should re-center the red dot in the middle of the display.
To validate this calibration, simply press the space bar again.



The following image is the digit dial participants were shown to validate their selection:

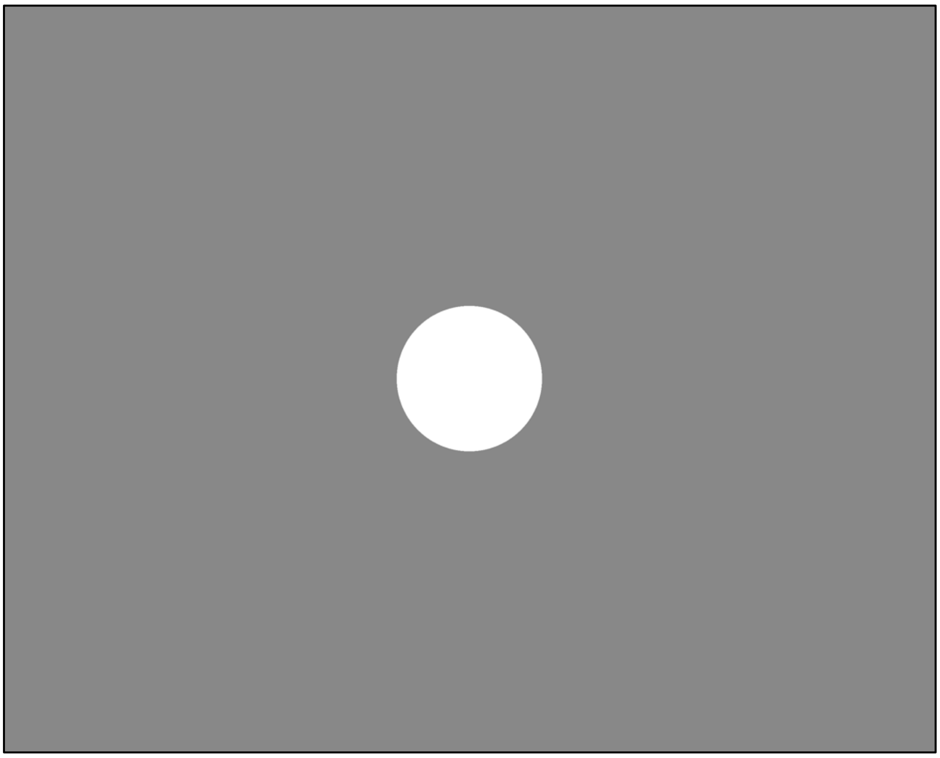


The following pictures are the feedback given after a response has been entered (correct/incorrect):



The following image is the displayed instructions at the beginning of each task:

The following image is the mask containing the foveated transparent circle:



Visual Paradigm Scripts:

```
%% MSC PROJECT %%
% KCL Neuroimaging MSc Project
% [created by: J P Monney  07/06/2019 - updated: 12/07/2019]

addpath(genpath('.'));
clear all;


%% 1. Dialog box to enter participant number & mode
main_directory = uigetdir('.','Select Main Output Directory');
image_directory = uigetdir('.','Select Image Directory');
prompt = 'Enter participant number';
participant = inputdlg(prompt);
participant = participant{1};
prompt = 'Enter mode: \n mode 0 = trial session \n mode 1 = test session';
mode = inputdlg(sprintf(prompt));
mode = mode{1};
mode = str2num(mode);

%---------------------------------------------------------------------------
%% Enter into trial mode
if mode == 0
    cgloadlib;
    % config_display(1,[xres yres]) % configure resolution
    % cgopen(1,8,0,1); %for fullscreen
    task_num = 0;
    edfFile = eyelink_jona(task_num); % 1.-4.

    Eyelink('StartRecording'); % 5.  Start recording eye position
    WaitSecs(0.1) % 6. Record a few samples before we actually start
displaying
    Eyelink('Message', 'SYNCTIME'); % 7.  Mark zero-plot time in data file

    cgopen(1000,1000,0,0,0);
    cgflip;
    cgloadbmp(107,'task4.bmp',1000,1000);
    cgloadbmp(108,'instructions1234.bmp',1000,1000);
    cgdrawsprite(107,0,0);
    cgflip;
    wait(1500);
    cgdrawsprite(108,0,0);
    cgflip;
    wait(4000);
    config_data( 'random.dat' );
    for i = 1:10
        x1 = randi(1000);
        mnist = getdata(x1, 2);
        label = getdata(x1, 1);
        display_test(label, mnist, i, image_directory);
    end

    % 10. STOP RECORDING the eyetracker & Close the file
    Eyelink('StopRecording');
    Eyelink('CloseFile');

    % 11. DOWNLOAD DATA into your favourite folder
    cd(main_directory);
```

```
    mkdir TRIAL;
    cd(TRIAL);

    try
        fprintf('Receiving data file ''%s''\n', edfFile );
        status=Eyelink('ReceiveFile');
        if status > 0
            fprintf('ReceiveFile status %d\n', status);
        end
        if 2==exist(edfFile, 'file')
            fprintf('Data file ''%s'' can be found in ''%s''\n', edfFile,
pwd );
        end
    catch
        fprintf('Problem receiving data file ''%s''\n', edfFile );
    end


else
%---------------------------------------------------------------------------

    %% 2. Creates an output folder for the participant's results
    cd(main_directory);
    participant_folder = strcat('SUBJECT_', participant);
    mkdir(participant_folder);
    cd(participant_folder);
    mkdir('TASK1');
    mkdir('TASK2');
    mkdir('TASK3');
    mkdir('TASK4');
    cd(image_directory);


    %% 3. Start Cogent
    cgloadlib;
    % config_display(1,[xres yres]) % configure resolution
    % cgopen(1,8,0,1); %for fullscreen
    cgopen(1000,1000,0,0,0);
    cgflip;


    %% TASK1 = sequence & connect eyetracker
    task_num = 1;
    edfFile = eyelink_jona(task_num); % 1.-4.

    Eyelink('StartRecording'); % 5.  Start recording eye position
    WaitSecs(0.1) % 6.  Record a few samples before we actually start
displaying
    Eyelink('Message', 'SYNCTIME'); % 7.  Mark zero-plot time in data file

    cgloadbmp(101,'task1.bmp',1000,1000);
    cgloadbmp(102,'instructions1234.bmp',1000,1000);
    cgdrawsprite(101,0,0);
    cgflip;
    wait(1500);
    cgdrawsprite(102,0,0);
    cgflip;
    wait(4000);
    config_data( 'sequence.dat');
    for i = 1:3 %in test, i = 1:100
        mnist = getdata(i, 2);
```

```
        label = getdata(i, 1);
        TASK1_output.trial(i) = display_sequ(participant, label, mnist,
i,...
            participant_folder, image_directory, main_directory);
    end
    cd(main_directory);
    cd(participant_folder);
    save('TASK1_output.mat', 'TASK1_output'); %save the struct to a .mat
    cd(image_directory);

    Eyelink('StopRecording'); % 10. STOP RECORDING the eyetracker & Close
the file
    Eyelink('CloseFile'); % 11. DOWNLOAD DATA into your favourite folder
    cd(main_directory);
    cd TASK1;
    try
        fprintf('Receiving data file ''%s''\n', edfFile );
        status=Eyelink('ReceiveFile');
        if status > 0
            fprintf('ReceiveFile status %d\n', status);
        end
        if 2==exist(edfFile, 'file')
            fprintf('Data file ''%s'' can be found in ''%s''\n', edfFile,
pwd );
        end
    catch
        fprintf('Problem receiving data file ''%s''\n', edfFile );
    end

    cgloadbmp(123,'break.bmp',1000,1000);
    cgdrawsprite(123,0,0);
    cgflip;
    kp = 0;
    while ~kp
        [ks,kp] = cgkeymap;
    end

    %% TASK2 = volatility
    task_num = 2;
    edfFile = eyelink_jona(task_num); % 1.-4.

    Eyelink('StartRecording'); % 5.  Start recording eye position
    WaitSecs(0.1) % 6.  Record a few samples before we actually start
displaying
    Eyelink('Message', 'SYNCTIME'); % 7.  Mark zero-plot time in data file

    cgloadbmp(103,'task2.bmp',1000,1000);
    cgloadbmp(104,'instructions1234.bmp',1000,1000);
    cgdrawsprite(103,0,0);
    cgflip;
    wait(1500);
    cgdrawsprite(104,0,0);
    cgflip;
    wait(4000);
    config_data( 'volatility.dat');
    for i = 1:3 %in test, i = 1:100
        mnist = getdata(i, 2);
        label = getdata(i, 1);
        TASK2_output.trial(i) = display_volatility(participant,label,
mnist, i,...
            participant_folder, image_directory, main_directory);
    end
```

```
    cd(main_directory);
    cd(participant_folder);
    save('TASK2_output.mat', 'TASK2_output');
    cd(image_directory);

    Eyelink('StopRecording'); % 10. STOP RECORDING the eyetracker & Close
the file
    Eyelink('CloseFile'); % 11. DOWNLOAD DATA into your favourite folder
    cd(main_directory);
    cd TASK2;
    try
        fprintf('Receiving data file ''%s''\n', edfFile );
        status=Eyelink('ReceiveFile');
        if status > 0
            fprintf('ReceiveFile status %d\n', status);
        end
        if 2==exist(edfFile, 'file')
            fprintf('Data file ''%s'' can be found in ''%s''\n', edfFile,
pwd );
        end
    catch
        fprintf('Problem receiving data file ''%s''\n', edfFile );
    end

    cgloadbmp(123,'break.bmp',1000,1000);
    cgdrawsprite(123,0,0);
    cgflip;
    kp = 0;
    while ~kp
        [ks,kp] = cgkeymap;
    end

    %% TASK3 = higher/lower
    task_num = 3;
    edfFile = eyelink_jona(task_num); % 1.-4.

    Eyelink('StartRecording'); % 5.  Start recording eye position
    WaitSecs(0.1) % 6.  Record a few samples before we actually start
displaying
    Eyelink('Message', 'SYNCTIME'); % 7.  Mark zero-plot time in data file

    cgloadbmp(105,'task3.bmp',1000,1000);
    cgloadbmp(106,'instructions1234.bmp',1000,1000);
    cgdrawsprite(105,0,0);
    cgflip;
    wait(1500);
    cgdrawsprite(106,0,0);
    cgflip;
    wait(4000);
    config_data( 'random.dat');
    prev_label = 0;
    for i = 1:3 %in test, i = 1:100
        x2 = randi(1000);
        mnist = getdata(x2, 2);
        label = getdata(x2, 1);
        TASK3_output.trial(i) = display_compare(participant, label, mnist,
prev_label,...
            i, participant_folder, image_directory, main_directory);
        prev_label = label;
    end
    cd(main_directory);
    cd(participant_folder);
```

```
    save('TASK3_output.mat', 'TASK3_output');
    cd(image_directory);

    Eyelink('StopRecording'); % 10. STOP RECORDING the eyetracker & Close
the file
    Eyelink('CloseFile'); % 11. DOWNLOAD DATA into your favourite folder
    cd(main_directory);
    cd TASK3;
    try
        fprintf('Receiving data file ''%s''\n', edfFile );
        status=Eyelink('ReceiveFile');
        if status > 0
            fprintf('ReceiveFile status %d\n', status);
        end
        if 2==exist(edfFile, 'file')
            fprintf('Data file ''%s'' can be found in ''%s''\n', edfFile,
pwd );
        end
    catch
        fprintf('Problem receiving data file ''%s''\n', edfFile );
    end

    cgloadbmp(123,'break.bmp',1000,1000);
    cgdrawsprite(123,0,0);
    cgflip;
    kp = 0;
    while ~kp
        [ks,kp] = cgkeymap;
    end

    %% TASK4 = random scanning
    task_num = 4;
    edfFile = eyelink_jona(task_num); % 1.-4.

    Eyelink('StartRecording'); % 5.  Start recording eye position
    WaitSecs(0.1) % 6.  Record a few samples before we actually start
displaying
    Eyelink('Message', 'SYNCTIME'); % 7.  Mark zero-plot time in data file

    cgloadbmp(107,'task4.bmp',1000,1000);
    cgloadbmp(108,'instructions1234.bmp',1000,1000);
    cgdrawsprite(107,0,0);
    cgflip;
    wait(1500);
    cgdrawsprite(108,0,0);
    cgflip;
    wait(4000);
    config_data( 'random.dat' );
    for i = 1:3 %in test, i = 1:100
        x1 = randi(1000);
        mnist = getdata(x1, 2);
        label = getdata(x1, 1);
        TASK4_output.trial(i) = display_rand(participant, label, mnist,
i,...
            participant_folder, image_directory, main_directory);
    end
    cd(main_directory);
    cd(participant_folder);
    save('TASK4_output.mat', 'TASK4_output');
    cd(image_directory);
    cgloadbmp(123,'break.bmp',1000,1000);
    cgdrawsprite(123,0,0);
```

```
    cgflip;

    Eyelink('StopRecording'); % 10. STOP RECORDING the eyetracker & Close
the file
    Eyelink('CloseFile'); % 11. DOWNLOAD DATA into selected folder
    cd(main_directory);
    cd TASK4;
    try
        fprintf('Receiving data file ''%s''\n', edfFile );
        status=Eyelink('ReceiveFile');
        if status > 0
            fprintf('ReceiveFile status %d\n', status);
        end
        if 2==exist(edfFile, 'file')
            fprintf('Data file ''%s'' can be found in ''%s''\n', edfFile,
pwd );
        end
    catch
        fprintf('Problem receiving data file ''%s''\n', edfFile );
    end


    %% 4. Update genpath, show result files, close cogent
    addpath(genpath(main_directory));
    cd(main_directory);
    cd(participant_folder);

    kp = 0;
    while ~kp
        [ks,kp] = cgkeymap;
    end
end

% cgtracker('shut');
cgshut
```

Task 1 Script:

```
function [trial] = display_sequ(participant, label, mnist, i,...
    participant_folder, image_directory, main_directory)
% Does the exact same job as display_rand
% KCL Neuroimaging MSc Project
% [created by: J P Monney  27/05/2019 - updated: 23/06/2019]

cd(image_directory);

cgloadbmp(1,mnist,400,400)
cgloadbmp(2,'circle.bmp',2000,2000)
cgloadbmp(3,'cross.bmp',1300,1300)
cgloadbmp(4,'task_instr.bmp',1000,1000)
cgloadbmp(5,'digit_dial.bmp',1000,1000)
cgloadbmp(6,'correct.bmp',1000,1000)
cgloadbmp(7,'incorrect.bmp',1000,1000)
cgloadbmp(9,'white.bmp',1000,1000)
cgloadbmp(10,'black.bmp',1000,1000)
cgloadbmp(11,'ready.bmp',1000,1000)
cgtrncol(2,'n')


% Uses space bar press as signal for ready to start task
[ks,kp]=cgkeymap; % to re-initialize accidental button press
kp=0;
cgdrawsprite(11,0,0);
cgflip;
while ~kp
   [ks,kp]=cgkeymap;
end


tic;
timerV = 0;
eye_used = 0;
xres = 1280; % horizontal resolution
yres = 1024; % vertical resolution
loop_count = 1; %to create coordinates cell
coordinates = cell(1); %to store gaze data
init = 1;
[ks,kp]=cgkeymap;
kp=0;
tmess = [ 'trial_start' num2str(i)]; % 8. MESSAGE - send & log start of
each trial
Eyelink('Message', tmess);
while  ~kp & (timerV < 3)
   [x,y]=cgmouse; % tracks mouse in lieu of gaze [for now...]
   if Eyelink('NewFloatSampleAvailable') > 0
        % get the sample in the form of an event structure
        evt = Eyelink('NewestFloatSample');
        x = evt.gx(eye_used+1);
        y = evt.gy(eye_used+1);

        if x ~= el.MISSING_DATA && y ~= el.MISSING_DATA
            x = x - xres/2; % +1 as we're accessing MATLAB array
            y = yres/2 - y;
        end
   end
   if init ==1
       cgdrawsprite(3,0,0);
```

```
        cgflip;
        r1 = 20; % fixation cross centre radius
        D1 = ((((x - 0)^2)+((y - 0)^2))^0.5);
        while D1 > r1
            [x,y]=cgmouse;
            D1 = ((((x - 0)^2)+((y - 0)^2))^0.5); % recalculate
        end
        init = 2; % stop showing cross
        start = toc;
        [ks,kp]=cgkeymap;
        kp=0;
    end
    [ks,kp]=cgkeymap;
    cgdrawsprite(10,0,0);
    cgdrawsprite(1,0,0);
    cgdrawsprite(2,x,y);
    cgflip;
    finish = toc;
    timerV = (finish - start);
    coordinates{loop_count,1} = x;
    coordinates{loop_count,2} = y;
    coordinates{loop_count,3} = finish;
    loop_count = loop_count +1;
end
tmess = [ 'trial_end' num2str(i)];
Eyelink('Message', tmess);


            % if (timerV < 0.5)
            %     cgdrawsprite(10,0,0);
            %     cgscale(30);
            %     cgfont('Arial',1);
            %     cgpencol(0,1,0);
            %     timerV = num2str(timerV);
            %     elapsed_time = strcat('Time ', timerV, ' s.');
            %     cgtext(elapsed_time,0,0);
            %     cgflip;
            %     wait(1000);
            % else
            %     cgdrawsprite(10,0,0);
            %     cgscale(30);
            %     cgfont('Arial',1);
            %     cgpencol(1,0,0);
            %     timerV = num2str(timerV);
            %     elapsed_time = strcat('Time ', timerV, ' s.');
            %     cgtext(elapsed_time,0,0);
            %     cgflip;
            %     wait(1000);
            % end

% Explains to participant how to validate their answer
cgdrawsprite(4,0,0);
cgflip;
wait(1000);

% Waits for participant to press space button to validate gaze location
[ks,kp]=cgkeymap;
kp=0;
while ~kp
    [ks,kp]=cgkeymap;
    cgdrawsprite(5,0,0);
    cgflip;
```

```
    [x,y] = cgmouse;
    % [x,y]=cgtracker('eyedat'); % tracks gaze
end

    cgdrawsprite(9,0,0);
    cgflip;
    wait(100);

% Display correct or incorrect
if label == 0
    r = 81;
    D = ((((x - 9)^2)+((y - 370)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 1
 r = 81;
    D = ((((x - 235)^2)+((y - 300)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 2
 r = 81;
    D = ((((x - 380)^2)+((y - 110)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 3
 r = 81;
    D = ((((x - 380)^2)+((y + 130)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
```

```
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 4
 r = 81;
    D = ((((x - 235)^2)+((y + 320)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 5
 r = 81;
    D = ((((x - 9)^2)+((y + 400)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 6
 r = 81;
    D = ((((x + 215)^2)+((y + 320)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 7
 r = 81;
    D = ((((x + 360)^2)+((y + 130)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
```

```
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 8
 r = 81;
    D = ((((x + 360)^2)+((y - 110)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 9
 r = 81;
    D = ((((x + 215)^2)+((y - 300)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

end

%Create output file and save label and mnist file data
output_file = strcat(participant, '_TASK1_', int2str(i), '_',...
    result, '_', int2str(label), '.mat');
cd(main_directory);
cd(participant_folder);
cd('TASK1');
save(output_file, 'participant', 'label', 'mnist', 'result',
'coordinates');
trial.participant = participant;
trial.label = label;
trial.file = mnist;
trial.result = result;
trial.coordinates = coordinates;
cd(image_directory);

end
```

Task 2 Script:

```
function [trial] = display_volatility(participant, label, mnist, i,...
    participant_folder, image_directory, main_directory)
% Does the exact same job as display_rand
% KCL Neuroimaging MSc Project
% [created by: J P Monney  27/05/2019 - updated: 23/06/2019]

cd(image_directory);

cgloadbmp(1,mnist,400,400)
cgloadbmp(2,'circle.bmp',2000,2000)
cgloadbmp(3,'cross.bmp',1300,1300)
cgloadbmp(4,'task_instr.bmp',1000,1000)
cgloadbmp(5,'digit_dial.bmp',1000,1000)
cgloadbmp(6,'correct.bmp',1000,1000)
cgloadbmp(7,'incorrect.bmp',1000,1000)
cgloadbmp(9,'white.bmp',1000,1000)
cgloadbmp(10,'black.bmp',1000,1000)
cgloadbmp(11,'ready.bmp',1000,1000)
cgtrncol(2,'n')


% Uses space bar press as signal for ready to start task
[ks,kp]=cgkeymap; % to re-initialize accidental button press
kp=0;
cgdrawsprite(11,0,0);
cgflip;
while ~kp
   [ks,kp]=cgkeymap;
end


tic;
timerV = 0;
eye_used = 0;
xres = 1280; % horizontal resolution
yres = 1024; % vertical resolution
loop_count = 1; %to creat coordinates cell
coordinates = cell(1); %to store gaze data
init = 1;
[ks,kp]=cgkeymap;
kp=0;
tmess = [ 'trial_start' num2str(i)]; % 8. MESSAGE - send & log start of
each trial
Eyelink('Message', tmess);
while  ~kp & (timerV < 3)
   [x,y]=cgmouse; % tracks mouse in lieu of gaze [for now...]
   if Eyelink('NewFloatSampleAvailable') > 0
        % get the sample in the form of an event structure
        evt = Eyelink('NewestFloatSample');
        x = evt.gx(eye_used+1);
        y = evt.gy(eye_used+1);

        if x ~= el.MISSING_DATA && y ~= el.MISSING_DATA
            x = x - xres/2; % +1 as we're accessing MATLAB array
            y = yres/2 - y;
        end
   end
   if init ==1
       cgdrawsprite(3,0,0);
```

```
        cgflip;
        r1 = 20; % fixation cross centre radius
        D1 = ((((x - 0)^2)+((y - 0)^2))^0.5);
        while D1 > r1
            [x,y]=cgmouse;
            D1 = ((((x - 0)^2)+((y - 0)^2))^0.5); % recalculate
        end
        init = 2; % stop showing cross
        start = toc;
        [ks,kp]=cgkeymap;
        kp=0;
    end
    [ks,kp]=cgkeymap;
    cgdrawsprite(10,0,0);
    cgdrawsprite(1,0,0);
    cgdrawsprite(2,x,y);
    cgflip;
    finish = toc;
    timerV = (finish - start);
    coordinates{loop_count,1} = x;
    coordinates{loop_count,2} = y;
    coordinates{loop_count,3} = finish;
    loop_count = loop_count +1;
end
tmess = [ 'trial_end' num2str(i)];
Eyelink('Message', tmess);

% Explains to participant how to validate their answer
cgdrawsprite(4,0,0);
cgflip;
wait(1000);

% Waits for participant to press space button to validate gaze location
[ks,kp]=cgkeymap;
kp=0;
while ~kp
    [ks,kp]=cgkeymap;
    cgdrawsprite(5,0,0);
    cgflip;
    [x,y] = cgmouse;
    % [x,y]=cgtracker('eyedat'); % tracks gaze
end

    cgdrawsprite(9,0,0);
    cgflip;
    wait(100);

% Display correct or incorrect
if label == 0
    r = 81;
    D = ((((x - 9)^2)+((y - 370)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end
```

```
elseif label == 1
 r = 81;
    D = ((((x - 235)^2)+((y - 300)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 2
 r = 81;
    D = ((((x - 380)^2)+((y - 110)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 3
 r = 81;
    D = ((((x - 380)^2)+((y + 130)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 4
 r = 81;
    D = ((((x - 235)^2)+((y + 320)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 5
```

```
 r = 81;
    D = ((((x - 9)^2)+((y + 400)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 6
 r = 81;
    D = ((((x + 215)^2)+((y + 320)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 7
 r = 81;
    D = ((((x + 360)^2)+((y + 130)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 8
 r = 81;
    D = ((((x + 360)^2)+((y - 110)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 9
 r = 81;
    D = ((((x + 215)^2)+((y - 300)^2))^0.5);
```

```
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

end

%Create output file and save label and mnist file data
output_file = strcat(participant, '_TASK2_', int2str(i), '_',...
    result, '_', int2str(label), '.mat');
cd(main_directory);
cd(participant_folder);
cd('TASK2');
save(output_file, 'participant', 'label', 'mnist', 'result',
'coordinates');
trial.participant = participant;
trial.label = label;
trial.file = mnist;
trial.result = result;
trial.coordinates = coordinates;
cd(image_directory);

end
```

Task 3 Script:

```
function [trial] = display_rand(participant, label, mnist, i,...
    participant_folder, image_directory, main_directory)
% Takes the selected MNIST digit and its value - performs task
%1. Fixation cross (press space bar when ready)
%2. Displays alpha hole over MNIST digit
%3. Answer validation instructions (press space bar when staring at digit)
%4. Gives feedback on answer
% KCL Neuroimaging MSc Project
% [created by: J P Monney  27/05/2019 - updated: 23/06/2019]

cd(image_directory);

cgloadbmp(1,mnist,400,400)
cgloadbmp(2,'circle.bmp',2000,2000)
cgloadbmp(3,'cross.bmp',1300,1300)
cgloadbmp(4,'task_instr.bmp',1000,1000)
cgloadbmp(5,'digit_dial.bmp',1000,1000)
cgloadbmp(6,'correct.bmp',1000,1000)
cgloadbmp(7,'incorrect.bmp',1000,1000)
cgloadbmp(9,'white.bmp',1000,1000)
cgloadbmp(10,'black.bmp',1000,1000)
cgloadbmp(11,'ready.bmp',1000,1000)
cgtrncol(2,'n')



% Uses space bar press as signal for ready to start task
[ks,kp]=cgkeymap; % to re-initialize accidental button press
kp=0;
cgdrawsprite(11,0,0);
cgflip;
while ~kp
   [ks,kp]=cgkeymap;
end


tic;
timerV = 0;
eye_used = 0;
xres = 1280; % horizontal resolution
yres = 1024; % vertical resolution
xcross = randi([-300,300]); % random placement of cross
ycross = randi([-300,300]);
loop_count = 1; %to creat coordinates cell
coordinates = cell(1); %to store gaze data
init = 1;
[ks,kp]=cgkeymap; % to re-initialize accidental button press
kp=0;
tmess = [ 'trial_start' num2str(i)]; % 8. MESSAGE - send & log start of
each trial
Eyelink('Message', tmess);
while  ~kp & (timerV < 3) % either button press or max 3 seconds
   [x,y]=cgmouse; % tracks mouse in lieu of gaze [for now...]
   if Eyelink('NewFloatSampleAvailable') > 0
        % get the sample in the form of an event structure
        evt = Eyelink('NewestFloatSample');
        x = evt.gx(eye_used+1);
        y = evt.gy(eye_used+1);
```

```
            if x ~= el.MISSING_DATA && y ~= el.MISSING_DATA
                x = x - xres/2; % +1 as we're accessing MATLAB array
                y = yres/2 - y;
            end
        end
        if init ==1
            cgdrawsprite(3,xcross,ycross);
            cgflip;
            r1 = 20; % fixation cross centre radius
            D1 = ((((x - xcross)^2)+((y - ycross)^2))^0.5);
            while D1 > r1
                [x,y]=cgmouse;
                D1 = ((((x - xcross)^2)+((y - ycross)^2))^0.5); % recalculate
            end
            init = 2; % stop showing cross
            start = toc;
            [ks,kp]=cgkeymap;
            kp=0;
        end
        [ks,kp]=cgkeymap;
        cgdrawsprite(10,0,0); % black BG
        cgdrawsprite(1,0,0); % MNIST digit
        cgdrawsprite(2,x,y); % transparent circle
        cgflip;
        finish = toc;
        timerV = (finish - start);
        coordinates{loop_count,1} = x;
        coordinates{loop_count,2} = y;
        coordinates{loop_count,3} = finish;
        loop_count = loop_count +1;
    end
    tmess = [ 'trial_end' num2str(i)];
    Eyelink('Message', tmess);

    % Explains to participant how to validate their answer
    cgdrawsprite(4,0,0);
    cgflip;
    wait(1000);

    % Waits for participant to press space button to validate gaze location
    [ks,kp]=cgkeymap; % to re-initialize accidental button press
    kp=0;
    while ~kp
        [ks,kp]=cgkeymap;
        cgdrawsprite(5,0,0);
        cgflip;
        [x,y] = cgmouse;
        % [x,y]=cgtracker('eyedat'); % tracks gaze

    end
    cgdrawsprite(9,0,0);
    cgflip;
    wait(100);


    % Display correct or incorrect
    if label == 0
        r = 81;
        D = ((((x - 9)^2)+((y - 370)^2))^0.5);
        if (D < r)
            cgdrawsprite(6,0,0);
            cgflip;
```

```
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 1
 r = 81;
    D = ((((x - 235)^2)+((y - 300)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 2
 r = 81;
    D = ((((x - 380)^2)+((y - 110)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 3
 r = 81;
    D = ((((x - 380)^2)+((y + 130)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 4
 r = 81;
    D = ((((x - 235)^2)+((y + 320)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
```

```
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 5
 r = 81;
    D = ((((x - 9)^2)+((y + 400)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 6
 r = 81;
    D = ((((x + 215)^2)+((y + 320)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 7
 r = 81;
    D = ((((x + 360)^2)+((y + 130)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 8
 r = 81;
    D = ((((x + 360)^2)+((y - 110)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
```

```
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

elseif label == 9
 r = 81;
    D = ((((x + 215)^2)+((y - 300)^2))^0.5);
    if (D < r)
        cgdrawsprite(6,0,0);
        cgflip;
        wait(1000);
        result = 'correct';
    else
        cgdrawsprite(7,0,0);
        cgflip;
        wait(1000);
        result = 'incorrect';
    end

end

%Create output file and save label and mnist file data
output_file = strcat(participant, '_TASK4_', int2str(i), '_',...
    result, '_', int2str(label), '.mat');
cd(main_directory);
cd(participant_folder);
cd('TASK4');
save(output_file, 'participant', 'label', 'mnist', 'result',...
    'coordinates');
trial.participant = participant;
trial.label = label;
trial.file = mnist;
trial.result = result;
trial.coordinates = coordinates;
cd(image_directory);



end
```

NIfTI Conversion & Analysis Script:

```
% % FOR TESTING WITH RANDOM MATRICES
%
% matrix = zeros(100,100,100,30);
%
% for i = 1:30
%     x = randi([19 22],1,5);
%     y = randi([29 32],1,5);
%     z = randi([29 32],1,5);
%     matrix(x,y,z,i) = 1;
% end
%
% nii = make_nii(matrix);
% save_nii(nii, 'testNIFTI.nii');
% view_nii(nii);

%% created on 11.08.19 by J.MONNEY updated 20.08.19

clear all;
task = 'Task3'; % example task
task_messages = [(task) '_messages'];
digit_num = 0; % change this variable for the digit desired
load('/Users/JonA/Documents/MSc Project/DATA/processedData_COMBINED.mat');
cd('/Users/JonA/Documents/MSc Project/DATA/niftis/Task 3/AAA');

for subj = 1%:numel(fieldnames(AllData.Matrices.MATLABcoordinates))
    subject = fieldnames(AllData.Matrices.MATLABcoordinates);
    subject = subject{subj};

    [timestamp, trial] =
size(AllData.Matrices.MATLABcoordinates.(subject).(task).X);
    for b = 1:trial
        trialNum = ['trial' num2str(b)];
        label = AllData.Raw.(subject).(task_messages).Labels.(trialNum);
        if label == digit_num
            matrix(:,1) =
AllData.Matrices.MATLABcoordinates.(subject).(task).X(:,b);
            matrix(:,2) =
AllData.Matrices.MATLABcoordinates.(subject).(task).Y(:,b);

            for i = 1:size(matrix)
                V = isequal(matrix(i,1), 0); % finds all entries equal to 0
                w = isequal(matrix(i,2), 0);
                if V == 0 % all entries not equal to 0 are copied into
matrix2
                    matrix2(i,1) = matrix(i,1) + 640; % adjusts matlab
coordinates
                else
                    matrix2(i,1) = matrix(i,1);
                end
                if w == 0
                    matrix2(i,2) = matrix(i,2) + 512; % adjusts matlab
coordinates
                else
                    matrix2(i,2) = matrix(i,2);
                end
            end
            n_trials = length(matrix2);
            if (n_trials < 255)
```

```
                matrix2(n_trials:255,:) = 0; % makes all trials 3s. long
(85hz x 3s = 255 time steps)
            end
            matrix2 = matrix2(1:255,:); %trims matrix2 if longer than 255
time steps
            matrix2 = round(matrix2, -1); % rounds to nearest integer
            matrix2 = matrix2/10;
            t_length = length(matrix2); % number of entries
            Box = zeros(128,102,t_length); % creates base for nifti
            radius = 7; % adds a radius of 7  to gaze location

            for i = 1:t_length % gives value of 1 to where gaze is
                for m = 1:128
                    for n = 1:102
                        if (m > matrix2(i,1) - radius) && (m < matrix2(i,1)
+ radius) &&...
                                (n > matrix2(i,2) - radius) && (n <
matrix2(i,2) + radius)
                            Box(m,n,i) = 1;
                        elseif (matrix2(i,1) == 0) && (matrix2(i,2) == 0)
                            Box(m,n,i) = 0;
                        end
                    end
                end
            end
            nii = make_nii(Box);
            %view_nii(nii);
            filename = ['NIFTI_' num2str(subj) '_' num2str(b) '_'
num2str(label) '.nii'];
            save_nii(nii, filename);
        end
        clearvars -except AllData b subj digit_num subject task
task_messages;
    end
end


%
% nii = load_nii('stestNIFTI.nii');
% view_nii(nii);


%%   Display a 3d view of the cannonical scan path - not great...


nii = load_nii('/Users/JonA/Documents/MSc Project/DATA/niftis/Task 1/Early
1/con_0001.nii');
nii = nii.img;
%nii = nii(:,:,1:102);
nii = double(nii);
nii = permute(nii,[2 1 3]);

[fa, va] = isosurface(nii);
[fb, vb] = isosurface(nii, 0.3);
[fc, vc] = isosurface(nii, 0.4);
digit = imread('/Users/JonA/Documents/MSc Project/trial/1/29.bmp');
digit = imresize(digit, [102, 102]);
figure;
hold on
% imshow(digit);
p1 = patch('Faces', fa, 'Vertices', va);
```

```
p2 = patch('Faces', fb, 'Vertices', vb);
p3 = patch('Faces', fc, 'Vertices', vc);
p1.FaceAlpha = 0.2; p1.FaceColor = [0, 0, 1]; p1.EdgeAlpha = 0;
p2.FaceAlpha = 0.2; p2.FaceColor = [0, 1, 0]; p2.EdgeAlpha = 0;
p3.FaceAlpha = 0.2; p3.FaceColor = [1, 0, 0]; p3.EdgeAlpha = 0;
view(-10, 60);
hold off


%% Compressed image


nii = load_nii('/Users/JonA/Documents/MSc Project/DATA/niftis/Task
3/1/con_0001.nii');

compressed = nii.img;
Compressed = zeros(128,102);
for x = 1:length(compressed(:,1,1))
    for y = 1:length(compressed(1,:,1))
        for z = 1:length(compressed(1,1,:))
            check = isnan(compressed(x,y,z));
            if check == 1
                compressed(x,y,z) = 0;
            end
            if (x < 15) && (y < 15)
                compressed(x,y,z) = 0;
            end
        end
        Compressed(x,y) = sum(compressed(x,y,25:end));
        if (x < 15) && (y < 15)
            Compressed(x,y) = 0;
        end
    end
end
Compressed = Compressed';
figure;
imagesc(Compressed);


%%

imagerS = compressed;
imagerS = double(imagerS);

for x = 1:length(imagerS(:,1,1))
    for y = 1:length(imagerS(1,:,1))
        for z = 1:length(imagerS(1,1,:))
            if isnan(imagerS(x,y,z))
                imagerS(x,y,z) = 0;
            end
        end

    end
end

for x = 1:length(imagerS(:,1,1))
    for y = 1:length(imagerS(1,:,1))
        for z = 1:length(imagerS(1,1,:))

            imagerSs(x,y,1,z) = imagerS(x,y,z);
        end
```

```
    end
end


volViewer(imagerSs);



%% Look at difference heatmaps - still need to do stats in spm


nii1 = load_nii('/Users/JonA/Documents/MSc Project/DATA/niftis/Task 1/Early
3/con_0001.nii');
nii2 = load_nii('/Users/JonA/Documents/MSc Project/DATA/niftis/Task 1/Late
3/con_0001.nii');
img1 = nii1.img;
img2 = nii2.img;

for x = 1:length(img1(:,1,1))
    for y = 1:length(img1(1,:,1))
        for z = 1:length(img1(1,1,:))
            check1 = isnan(img1(x,y,z));
            if check1 == 1
                img1(x,y,z) = 0;
            end
            if (x < 15) && (y < 15)
                img1(x,y,z) = 0;
            end
            check2 = isnan(img2(x,y,z));
            if check2 == 1
                img2(x,y,z) = 0;
            end
            if (x < 15) && (y < 15)
                img2(x,y,z) = 0;
            end
        end
    end
end

img_delta = img1 - img2;
nii_delta = make_nii(img_delta);
% view_nii(nii_delta);

compressed = img_delta;
Compressed = zeros(128,102);
for x = 1:length(compressed(:,1,1))
    for y = 1:length(compressed(1,:,1))
        for z = 1:length(compressed(1,1,:))
            check = isnan(compressed(x,y,z));
            if check == 1
                compressed(x,y,z) = 0;
            end
        end
        Compressed(x,y) = sum(compressed(x,y,25:end));
        if (x < 15) && (y < 15)
            Compressed(x,y) = 0;
        end
    end
end

Compressed = Compressed';
```

```
figure;
imagesc(Compressed);
```

MDP model Script:

```
% Rosalyn Moran & Berk Mirza
% $Id: based on ...DEM_demo_MDP_reading.m 6866 2016-09-05 09:19:42Z karl $

clear all

rng('shuffle')


% first level (visual)
%==========================================================================

% prior beliefs about initial states
%--------------------------------------------------------------------------
no_digits   =   10;
no_wheres   =   8;
D{1}        =   ones(no_digits,1);      % what:  2 digits
D{2}        =   zeros(no_wheres,1);     % where: {'1'..,'4'}
D{2}(end)   =   1;

% probabilistic mapping from hidden states to outcomes: A
%--------------------------------------------------------------------------
Nf1    = numel(D);
for f = 1:Nf1
    Ns(f) = numel(D{f});
end

mdp.Aname = {'what_orientation','where'};
mdp.Bname = {'what_digit','where'};
No_o_what = 3;

% 2 outcomes ......
%outcomes - mini what {'horiz','vert','diagonal'} H - V - D
%A{WHAT}(what,what_next_level_ie_digit,where)

%       1 is blank, 2 is vertical (|) 3 is horizontal (_)
%            _
%           |_|
%           |_|

% Position      (1)
%           (2)(3)(4)
%           (5)(6)(7)
%              (8) === Fixation cross
%         [3 2 3 2 2 3 2]

  One   = [1 1 1 2 1 1 2] ;
  Two   = [3 1 3 2 2 3 1] ;
  Three = [3 1 3 2 1 3 2] ;
  Four  = [1 2 3 2 1 1 2] ;
  Five  = [3 2 3 1 1 3 2] ;
  Six   = [3 2 3 1 2 3 2] ;
  Seven = [3 1 1 2 1 1 2];
  Eight = [3 2 3 2 2 3 2];
  Nine  = [3 2 3 2 1 1 2];
  Zero  = [3 2 1 2 2 3 2];
```

```
  digit_whats = [One ;Two ;Three; Four; Five; Six ;Seven; Eight; Nine;
Zero];

% digit identity outcome
A{1} = zeros(No_o_what,Ns(1),Ns(2));
for i = 1:no_digits
    for p = 1:Ns(2)
        if p ~= Ns(2)
            A{1}(digit_whats(i,p),i,p) = 1;
        elseif p == Ns(2)
            A{1}(1,i,p) = 1;
        end
    end
end

% A{2} where: {'1',...,'8'}
%----------------------------------------------------------
A{2} = zeros(Ns(2),Ns(1),Ns(2));
for i = 1:Ns(2)
    A{2}(i,:,i) = 1;
end

% controlled transitions: B{f} for each factor
%--------------------------------------------------------------------------
for f = 1:Nf1
    B{f} = eye(Ns(f));
end

% controllable fixation points: move to the k-th location
%--------------------------------------------------------------------------
for k = 1:Ns(2)
    B{2}(:,:,k) = 0;
    B{2}(k,:,k) = 1;
end
Np         = size(B{2},3);
U          = ones(1,Np,Nf1);
U(:,:,2)   = 1:Np;

% MDP Structure
%--------------------------------------------------------------------------
mdp.T = 10;                          % Number of time steps the agent can
explore the scene
mdp.A = A;                          % Likelihood matrix
mdp.a{1} = ones(size(A{1})) + rand(size(A{1})) ;   % generative model
likelihood
mdp.a{2} = A{2}.*exp(32) ;                        % I KNOW MY WHERES FROM
MY WHERE's !!!
mdp.B = B;                          % Transition matrix
mdp.D = D;                          % Prior beliefs over initial states
mdp.U = U;                          % Policies

mdp.alpha = 128;      % deterministic actions
mdp.chi   = exp(-16); % 1/64; breaking too early
mdp.eta   = 1;        % learning rate

mdp_L1 = spm_MDP_check(mdp);

clear A B D mdp
% second level (decision)
%==========================================================================
```

```matlab
% prior beliefs about initial states (in terms of counts_: D and d
%--------------------------------------------------------------------------

D{1} = ones(no_digits,1);      % what:   { '1' '2' '3'....'0'} MNIST - 2
digits for now
D{2} = zeros(no_digits+1,1);   % report(where):   { '1' '2'
'3'....'0'...'undecided'} MNIST - 2 digits for now
D{2}(end) = 1;
No1  = no_digits  ;
No2  = 3; % undecided - right - wrong
Ng   = 1; %% one outcome at top level undecided - right - or wrong

% probabilistic mapping from hidden states to outcomes: A

% o1: BELIEF IN DIGIT - PRIOR FOR LOWER LEVEL
A{1} = zeros(No1,no_digits,no_digits+1);
% if you've reported then the outcome is that digit
for i = 1:No1
    A{1}(i,i,:) = 1;
end

% No2: outcomes  right - wrong - undecided
A{2}   =  zeros(No2, no_digits,no_digits+1) ;

for i = 1:no_digits
    A{2}(2,:,i) = 1; % wrong - most digits
    A{2}(1,i,i) = 1; % right- just one corresponding digit
    A{2}(2,i,i) = 0; % right- just one corresponding digit
end
A{2}(3,:,no_digits+1)  =  1; % state N is undecided
a{1} = exp(32)*spm_softmax(log(A{1}+exp(-8)));
a{2} = exp(32)*A{2};

% controlled transitions: B{f} for each factor
%--------------------------------------------------------------------------
for f = 1
    B{1} = eye(no_digits);
end

% % control states B(2): report  { '1' '2' '3'....'0'...'undecided'} MNIST
- 2 digits for now
%--------------------------------------------------------------------------
B{2} = zeros(no_digits+1,no_digits+1,no_digits+1);

  % pick a digit or last state is remain undecided
for k = 1:no_digits+1
      B{2}(k,:,k) = 1;
end
Nf2        = 2;
Np         = size(B{2},3);
U          = ones(1,Np,Nf2);
U(:,:,2)   = 1:Np;

% % allowable policies (specified as the next action) U
          % stay on current page and report sad
%
% priors: (utility) C
%--------------------------------------------------------------------------
C{1}  = zeros(No1,1); %% no preference for digits
C{2}  = zeros(No2,1);
C{2}(1,:) =  4;                % the agent expects to be right
```

```
C{2}(2,:) = -4;                 % and not wrong


% MDP Structure
%--------------------------------------------------------------------------
%mdp.link = zeros(Nf1,length(A2)); %% hidden states at level 1, outcomes at
level 2
 % link the 'predicted digit'

mdp.T = 32;                      % number of moves
mdp.U = U;                      % allowable policies
mdp.A = A;                      % observation model
mdp.a = a;
mdp.B = B;                      % transition probabilities
mdp.C = C;                      % preferred outcomes
mdp.D = D;                      % prior over initial states

mdp.Aname    = {'digit_response','feedback'};
mdp.Bname    = {'what','decision'};
mdp          = spm_MDP_check(mdp);

 mdp.MDP   = mdp_L1 ;
 % link the 'predicted digit'
 mdp.link = [1 0; 0 0];%% link output of A_level2{1} to hidden state 1 at
level 1

 % illustrate learning over trials
 %==========================================================================
N       = 500;
s(1,:) = ceil(rand(1,N)*10);

 for i = 1:N
     i
     test_digit = s(1,i);

     % True initial state % digit 2/undecided ... tell the agent to think
     % it's the real number
     mdp.s              = [s(1,i) no_digits+1]';
     mdp.D{1}           = ones(no_digits,1);
     mdp.D{2}           = zeros(no_digits+1,1);
     mdp.D{2}(end)      = 1;
     mdp.MDP.D{2}       = zeros(no_wheres,1) ;
     mdp.MDP.D{2}(end)   = 1 ;

     MDP_fin                        = spm_MDP_VB_X(mdp);
     mdp.MDP.a                      = MDP_fin.mdp(end).a;

     Result{i}.o = MDP_fin.o;
     Result{i}.s = MDP_fin.s;
     Result{i}.R = MDP_fin.R;
     M = size(MDP_fin.mdp(1).o,2);
     for j = 1:M
         Result{i}.lower(j).o = MDP_fin.mdp(j).o;
         Result{i}.lower(j).a = MDP_fin.mdp(j).a;
         Result{i}.lower(j).u = MDP_fin.mdp(j).u;
     end

 end
Result.test_digit = s(1,:);
save('Result.mat','Result')
```