Atten Percept Psychophys
DOI 10.3758/s13414-014-0633-3

The binding pool: A model of shared neural resources for distinct items in visual working memory

Garrett Swan · Brad Wyble

Abstract Visual working memory (VWM) refers to the ability to encode, store, and retrieve visual information. The two prevailing theories that describe VWM assume that information is stored either in discrete slots or within a shared pool of resources. However, there is not yet a good understanding of the neural mechanisms that would underlie such theories. To address this gap, we provide a computationally realized neural account that uses a pool of shared neurons to store information about one or more distinct stimuli. The binding pool model is a neural network that is essentially a hybrid of the slot and resource theories. It describes how information can be stored and retrieved from a pool of shared resources using a type/token architecture (Bowman & Wyble in Psychological Review 114(1), 38–70, 2007; Kanwisher in Cognition 27, 117–143, 1987; Mozer in Journal of Experimental Psychology: Human Perception and Performance 15(2), 287–303, 1989). The model can store multiple distinct objects, each containing binding links to one or more features. The binding links are stored in a pool of shared resources and, thus, produce mutual interference as memory load increases. Given a cue, the model retrieves a specific object and then reconstructs other features bound to that object, along with a confidence metric. The model can simulate data from continuous report and change detection paradigms and generates testable predictions about the interaction of report accuracy, confidence, and stimulus similarity. The testing of such predictions will help to identify the boundaries of shared resource theories, thereby providing insight into the roles of ensembles and context in explaining our ability to remember visual information.

Keywords Visual working memory · Computational model · Neural network · Working memory capacity · Slots and resources

G. Swan · B. Wyble (✉)
Psychology Department, Penn State University,
State College, PA, USA
e-mail: bwyble@gmail.com

G. Swan
e-mail: gsp.swan@gmail.com

Published online: 15 March 2014

Introduction

Whether we are completing a math problem or looking for our car keys on a cluttered desk, working memory enables success in a broad variety of tasks by foregrounding particular pieces of information for easy access. An important and as yet unresolved question is the format and mechanism of storage used by working memory. We explore this question with a focus on visual working memory (VWM), which specifically addresses the ability to store and retrieve visual pattern information and is distinct from a broader class of working memory capabilities (e.g., Baddeley & Hitch, 1974; Oberauer, 2009). Our goal is to construct a computational model of the neural circuits involved in VWM function that can simulate a broad range of existing data and provides a framework for understanding the functional mechanisms involved in visual memory storage.

We begin with a review of recent theory. Traditionally, VWM experiments have asked subjects to detect changes in sets of colored patches (Fig. 1a), and the results of such experiments have supported a theory that describes the structural nature of VWM in terms of discrete and independent slots (Luck & Vogel, 1997; Vogel, Woodman & Luck, 2001).

More recently, however, VWM experiments have recorded estimates of the precision of memory traces by using continuous variables, such as color (Fig. 1b), orientation, and spatial frequency (Bays, Catalao & Husain, 2009; Huang & Sekuler, 2010; Prinzmetal, Amiri, Allen & Edwards, 1998; Wilken & Ma, 2004). These experiments ask subjects to indicate, on a spectrum, exactly what feature value a particular stimulus had, and the variability of the responses reveals the precision of the retrieved memory trace. These paradigms reveal graded decrements in the quality of memory representations as set size increases. These results, in conjunction with change detection experiments involving complex stimuli (Alvarez & Cavanagh, 2004), have challenged theories about independent

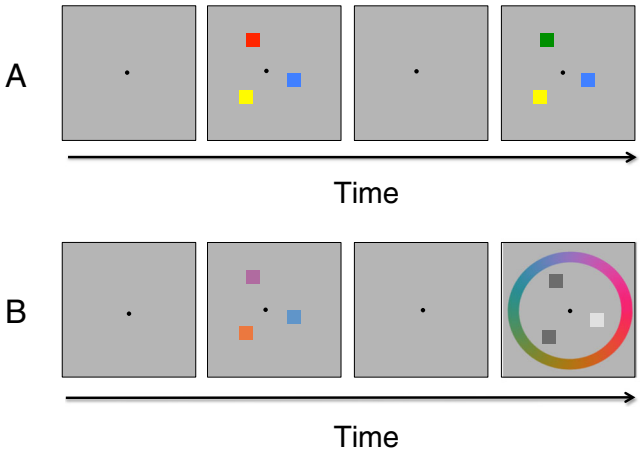

Fig. 1 Common visual working memory task designs. **a** In a whole-display change detection task, subjects see a set of stimuli (in this example, a red, blue, and yellow patch) and are asked to remember the items. The items disappear after an encoding duration (typically 100–500 ms), and following a retention interval of several hundred milliseconds, a new display is presented that may or may not contain a changed item. Subjects determine whether or not an item has changed. In this example, the red patch changed to green, so a correct response would be to respond "Change." **b** In a continuous report task, subjects see a set of stimuli and are asked to remember the hues. In this task, once the stimuli reappear, subjects are given a location cue (the light gray patch) to recreate that specific hue by selecting a color along the color wheel with a mouse click. The color wheel in this figure was generated using MemToolbox (Suchow, Brady, Fougnie & Alvarez, 2013; http://memtoolbox.org).

slots and support an alternative theory in which VWM involves a single fixed pool of resources that are divided across the items stored in a memory set. However, advocates of the slot model have suggested that these data do not disprove the slot theory and point out further evidence in favor of discrete representations (Awh, Barton & Vogel, 2007; W. Zhang & Luck, 2008).

This controversy continues, and the intensity of the debate between slot- and resource-based theorists has produced a progression of increasingly complex experimental designs (Bays, Gorgoraptis, Wee, Marshall & Husain, 2011a; Brady & Tenenbuam, 2013; Fougnie, Asplund & Marois, 2010; Lin & Luck, 2009; Orhan & Jacobs, 2013; Sims, Jacobs & Knill, 2012) and several mathematically descriptive models of VWM (Fougnie & Alvarez, 2011; Sims et al., 2012; van den Berg, Shin, Chou, George & Ma, 2012). Such models have provided valuable quantitative comparisons between variations of these theories that have challenged existing notions of slot-based models. Furthermore, computational models can also play a pivotal role in cognitive science by providing a formal specification of the mechanisms underlying a theoretical position. Creating such simulations can reveal crucial implicit assumptions of a theory, as well as the limits of a theory's explanatory power. Knowledge of such limitations makes it easier to investigate the role of more complex representations, such as ensembles, contextual effects, and configural information (Alvarez & Cavanagh, 2004; Jiang, Olson & Chun, 2000; Kahana & Sekuler, 2002[1]).

The approach that is taken here is complementary to these existing modeling approaches insomuch that elements of both the slot-based and shared resource models are incorporated in the proposed model. The proposed model can be considered a hybrid model of the slot-based and shared resource models, thereby contributing a mechanism by which these theories could be conjoined. Moreover, this model provides a mechanistic account of how discrete memory items can share neural resources in a neurally plausible simulation. We argue that performance across a variety of tasks cannot be adequately explained by slot or resource models. The binding pool model demonstrates that a broad spectrum of observed patterns of behavior can emerge through an interaction of these two kinds of limitations.

This model is a neural network simulation of a storage mechanism called the *binding pool*, in which memory traces are stored as a set of binding links within a distributed pool of neurons that is shared by multiple memory traces. The development of this model has been constrained by data from VWM paradigms that examine how changes in the complexity of a visual display affect the ability to detect changes (Keshvari, van den Berg & Ma, 2013) and to reconstruct a specific stimulus (Bays et al., 2009). The model also exhibits a number of inherent properties, such as the systematic interactions between stored items (Huang & Sekuler, 2010), evidence of binding-related errors (Wheeler & Treisman, 2002), the ability to retrieve one feature value of an object on the basis of another feature (Bays et al., 2011a), the impact of item similarity on retrieval precision (Sims et al., 2012), and the relationship between confidence and objective accuracy (Rademaker, Tredway & Tong, 2012).

We begin with a description of the capabilities of VWM that we consider as essential components of a model. We follow this with an informal description of the binding pool model, with a more formal description to follow in the Appendix. The remaining sections will detail existing data that the model explains and provide testable predictions about how object complexity, forgetting, similarity, repetition, and confidence affect performance on VWM tasks.

The functional constraints of visual working memory

In this context, functional constraints refer to the general behavioral capabilities of human subjects in VWM tasks that we consider necessary components of a general theory. We list here a core set of capabilities specified at an abstract level. Later in the article, specific empirical data will be used to constrain the model further.

[1] Kahana and Sekuler (2002) proposed what are essentially ensemble representations, although they were not named as such explicitly.

VWM is indexed and can store repetitions When presented with multiple items, subjects are able to selectively retrieve specific items presented at a given location (e.g., Wilken & Ma, 2004). Furthermore, while repetitions can be more difficult to encode than nonrepeated items (Kanwisher, 1987; Luo & Caramazza, 1996; Mozer, 1989), subjects are generally able to see and report multiple copies of the same item. The ability to encode repetitions is perhaps one of the most significant constraints on a model of memory, since it precludes models in which memories are stored within the same neurons that are used to process the sensory input (for further discussion, see Bowman & Wyble, 2007).

VWM is content addressable When presented with items containing multiple features such as lines with varying colors and orientations, subjects can retrieve, either the color of a line on the basis of its orientation, or the orientation on the basis of its color (Bays, Gorgoraptis, Catalao, Bays & Husain, 2011; van Lamsweerde & Beck, 2012). In computer science terminology, this is known as *content addressable* memory, which means that any subset of the contents of a memory trace can be used to retrieve the rest of it.

Multiple pieces of information stored in VWM produce mutual interference Storing multiple pieces of information into memory reduces the precision of the individual memory traces. This suggests either multiple that items compete for the same representational space (Bays et al., 2009; Wilken & Ma, 2004) or that multiple slots can be focused onto a small number of items (W. Zhang & Luck, 2008). Furthermore, this interference produces cross talk in that a stored item can pull the memory of another item toward its own value along a continuous feature dimension, such as spatial frequency (Huang & Sekuler, 2010).

VWM contents can be rapidly encoded and rapidly erased VWM storage is extremely mutable, such that a piece of information presented on the screen is available for retrieval almost immediately. Furthermore, stored information in visual memory seems to be almost entirely erasable, such that a given memory representation produces relatively little interference with a subsequent trial of a similar memory test (Huang & Sekuler, 2010).[2] This ability to discard information rapidly presumably allows us to perform complex tasks in rapid succession. If VWM contents could not be discarded easily, interference between stored representations of multiple stimuli might rapidly accrue and produce debilitating proactive interference.

[2] There is proactive interference in verbal working memory (Keppel & Underwood, 1962), but such interference does not seem to manifest strongly in visual working memory paradigms (Hartshorne, 2008; Lin & Luck, 2012).

Neurophysiological constraints of visual working memory

To provide a model that can be readily mapped onto brain mechanisms, the binding pool model was constructed through the use of simple neural elements that are similar to the elements described in early work on the Perceptron (Block, 1962; Rosenblatt, 1958). These simulated neurons can excite or inhibit one another and can store information by retaining a specific activity level across time. Maintenance of information through sustained activity might occur through either synaptic self-excitation or intracellular mechanisms (Hasselmo, Fransen, Dickson & Alonso, 2000). Another constraint of the model is that the strength of connections between neurons is not modifiable (at least within the timescale of a single trial of a memory experiment). Therefore, in this model, the only means by which information can be encoded and retrieved on the time scale of seconds is to modify the activity levels of neurons.

The binding pool model

The binding pool model is a simulation at the neural level that describes how information can be encoded in VWM.[3] The model will be described in an informal sense here and in greater detail in the Appendix. Also, the complete simulation code is available for download at http://wyblelab.com/research_repos/models/bindingpool/.

Informal description of the model

The binding pool model describes how neural resources can be dynamically allocated across one or more items in a memory display. Like the serial-order-in-a-box complex-span model (Oberauer, Lewandowsky, Farrell, Jarrold & Greaves, 2012), this model does not store the presence or absence of features; instead, it stores links between features. These stored links connect features (*types*) to object files (*tokens*[4]) using a

[3] The idea of a binding pool was first advanced in Bowman and Wyble (2007) in the context of encoding a sequence of items and was extended in later work (Wyble, Bowman & Nieuwenstein, 2009; Wyble, Potter, Bowman & Nieuwenstein, 2011). In these papers, the binding pool could only represent categorical stimuli and also lacked the ability to share neural resources between distinct items. The binding pool, as described here, uses representations that are shared between items and has the ability to represent features along a continuum. This new binding pool architecture can also represent categorical stimuli, and we assume that the binding pool described in the aforementioned models represents the same neural structure as this one.

[4] This use of the type/token terminology derives from earlier work (Bowman & Wyble, 2007). Note that there is a difference between the uses of type and token here and in a philosophical/linguistic context, but we maintain the use of the terminology on the grounds that the underlying idea is preserved. Types represent features of objects that might be repeated across objects, and tokens represent instance-specific memory representations.

shared, amodal pool of nodes (*binding pool*). Essentially, the binding pool acts as a switchboard that can encode one or more binding links between the object files and features (Fig. 2). When multiple links are stored in the switchboard, there is an accumulation of cross talk between them, producing a general loss of memory precision (Wilken & Ma, 2004), a drop in the ability to detect changes (Keshvari et al., 2013), swap errors (Bays et al., 2009), and systematic patterns of interference between stimuli that manifest at retrieval (Huang & Sekuler, 2010).

Types

In this model, types are used to represent abstract, high-level features, such as color hue, line orientation, and location. To represent a continuous variable, like color, in a set of discrete neural units, each node represents a specific point in color space, and colors that fall between those values are interpolated by using multiple type nodes (see the Appendix for details). Furthermore, to implement the continuity of a representation, type nodes that represent similar feature values (i.e., two shades of red) share more overlapping connections with the binding pool (see below) than do nodes with more distal values. We assume that the same continuity exists for other feature dimensions (e.g., spatial frequency, color, location, etc.). For the sake of conceptual and computational parsimony, at this point we assume that distinct feature dimensions are fully separable, even though this is likely to be incorrect. Note that the distinction of what constitutes a single feature is assumed to reflect experience with particular kinds of stimuli (VanRullen, 2009). Thus, it is probable that extensive experience with specific feature pairings would reorganize perceptual representations to create new "hard-wired" feature bindings that can be represented as patterns of type node activity without requiring active binding.

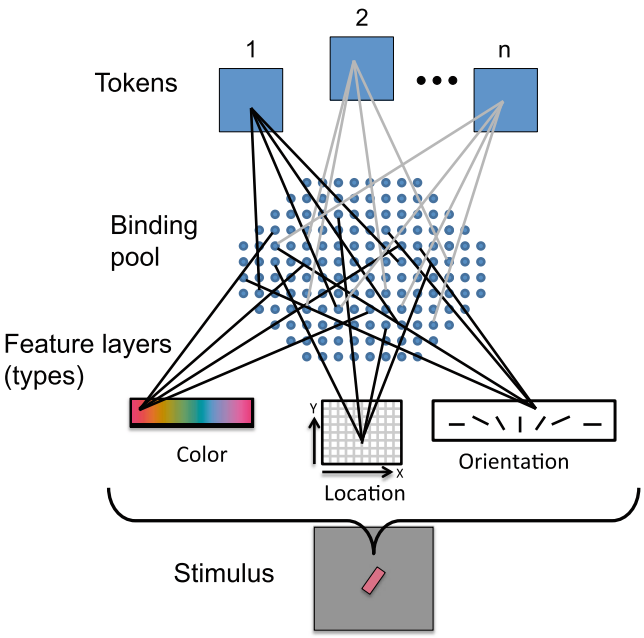

Fig. 2 In this illustration of the binding pool model, a diagonal purplish line is being stored (shown at the bottom of the figure). The features of the stimulus (color, location, and orientation) are separated into distinct feature layers, which we refer to as *types*. The types are connected to the *binding pool*, in which the type links are bound to an object representation, referred to as a *token*. In this example, the stimulus is being stored in token 1. However, if there were multiple stimuli, each would be assigned to a different token. The number of tokens shown here should not be construed as a capacity limit, since the model can encode more than three tokens per trial

Tokens

The ability to individuate different objects that might share features (especially when they share location) requires a form of indexing, and tokens provide this capability. We assume that at the presentation of a new stimulus display, the visual system identifies individual objects by detecting closed contours and creates tokens corresponding to each such object. Note that this initial allocation of token placeholders is grounded in the idea of place tokens as described by Marr (1976) and object files as described by Kahneman and Treisman (1984). For the purpose of parsimony, we abstract over the mechanisms required to detect and track such objects (e.g., Mihalas, Dong, von der Heydt & Niebur, 2011; Vul, Frank, Alvarez & Tenenbuam, 2010). Once allocated, each of these tokens can be linked to specific feature values (color, orientation, etc.) using the binding pool.

The binding pool

The *binding pool* is a set of nodes that encodes information about a visual stimulus using links that connect types to a single token. This type–token binding process provides an organizational structure for representing complex pieces of information. Unlike other theories of working memory that posit distinct representations of features and their bindings to objects (Wheeler & Treisman, 2002), the only form of information storage in the binding pool model are the binding links. Thus, in this model, features cannot be stored in an "unbound" state. Importantly, it is possible to represent repetitions of a feature by linking it to multiple tokens. In this theoretical framework, location is treated as a feature along with other features, such as color and orientation, and this means that the model can represent multiple items occurring at the same location by assigning each one to a distinct token. Once encoded, the links can subsequently be used to reconstruct which types are connected to a given token. Links within the binding pool are maintained in memory as self-sustained patterns of neural firing. Such activation-based

representations can be created and erased rapidly through excitation and inhibition of nodes.

Two kinds of capacity in the binding pool

The binding pool has distinct limitations on the quality and number of encoded items. These two forms of capacity interact in that, all else being equal, when more items are encoded into memory, each item's quality will be reduced.

The first such capacity limitation is related to the size of the binding pool, which is assumed to be a fixed, immutable property of the brain. The binding pool is a distributed form of memory that has much in common with preexisting ideas about mathematical abstractions of biological memory systems that are distributed (McClelland & Rumelhart, 1988; Murdock, 1983) or holographic in nature (Kanerva, 1993; Plate, 1995). In such models, a memory trace is not stored in a single neuron but is distributed across a large number of neurons. The more neurons that are available for a given representation, the greater the precision of the resultant memory reconstruction will be. In this sense, the capacity of the binding pool is not measured in terms of an absolute limit on the number of links, since the number of such links that might conceivably be represented is much larger than the number of stimuli in an experiment. Rather the capacity limit is observed in the way that memory for each individual item degrades as the number of stored links increases. As such, while there is no maximal limit to the number of links that can be created per se, there is a point at which so many links are stored that the amount of cross talk between them renders the memory representations indistinguishable from random noise.

The second capacity limitation is the number of items stored per trial, which is assumed to vary according to the parameters of the task, as well as intrinsic variations in attentional control from one moment to the next. The model stores individual items by linking them to tokens, and in this sense, tokens can be seen as analogous to slots. By fixing the number of tokens at three, the model would resemble a traditional slot model, with the added assumption that they share a binding pool. However, in order to best fit the data, the model assumes a variable number of items encoded per trial. In making this assumption, the model adopts the findings of Sims et al. (2012) and uses parameter fitting to characterize the distribution that describes this variability. We assume that the number of tokens encoded per trial is drawn from a uniform distribution (see the Appendix for details) that may be affected by task parameters, such as increased encoding duration, which seems to increase the number of items encoded (Vogel, Woodman & Luck, 2006) and to reduce the guessing rate (Bays et al., 2009). Thus, the limit on the number of encoded items is not fixed in the same way that the size of the binding pool is. A further assumption of this encoding limitation is that items that are not encoded have no influence on activity in the binding pool. This is an important assumption because it means that the ensemble effects we simulate below are the result of only the items that are encoded. In this article, we use one distribution of item-encoding limits for all of the simulations to reduce the number of parameters, but we expect that distribution to vary according to the task.

Connectivity between the three components of the model

The type nodes are connected to the binding pool with excitatory connections that are pseudorandomly determined. These connections are bidirectional such that a type node that excites a given binding pool node during encoding will also receive input from that same binding pool node during retrieval. Each type node is connected to a proportion of the nodes within the binding pool. These connections are pseudorandom for type nodes representing continuous features, in the sense that two neighboring type nodes in feature space (i.e., two shades of red) will share a greater proportion of connections than nodes that are farther away in type space. Tokens are connected to the binding pool in a similar fashion, except that they do not have a similarity gradient; their connectivity is fully random.

Encoding and retrieval of a single item

The model assumes that binding features together to form a coherent representation of a remembered object is an active process that creates a pattern of sustained activity within the binding pool to store links between features and locations, as indexed by tokens. We do not explicitly simulate the mechanisms required to understand the structure of the task or to interpret the visual cues on the visual display. Instead, we assume that a control signal is generated from task instructions and controls the flow of information within the model using gating signals (Wyble et al., 2009, provide an example of such a gating circuit). For example, during encoding, the activation values for the type nodes are fixed, and the activity levels of the binding pool nodes are modified to store information. During retrieval, the binding pool activity is held fixed, and the values of type nodes in one feature dimension is modified.

To encode a colored object at a specific location, the following steps occur in order. First, the color hue is translated into a pattern of activity across the set of type nodes corresponding to color, while location is represented on a different set of type nodes (see the Appendix for details). At the same time, a token node is allocated, corresponding to this object. Next, excitatory projections from each of these representations (color, location, and token) converge at the binding pool using the preestablished connectivity patterns (Fig. 3). These converging projections activate the subset of the binding pool nodes that receive input from all three of these representations. Those activated binding pool nodes have their activity level augmented, and this selective increase in activation stores the

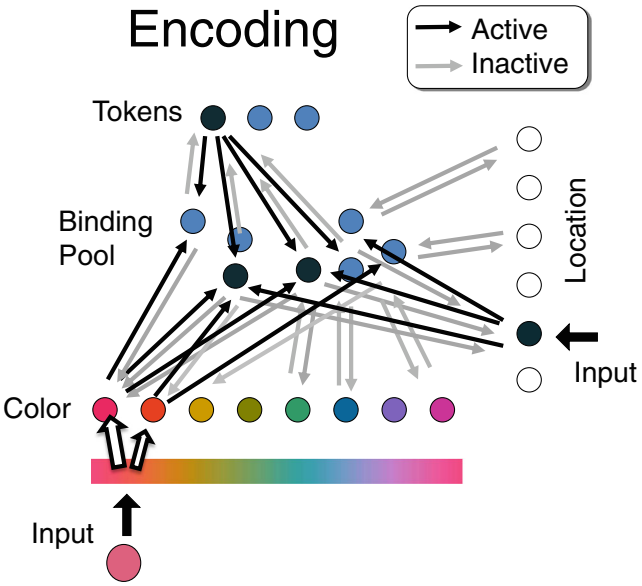

Fig. 3 In this illustration of encoding, the model is storing an item with a color and a location feature. The color and location are first translated into type node representations. The white arrows indicate the magnitude of activation during this conversion process (location, in this case, is categorical, so that the type node is either on or off). Then, these types provide input into the binding pool along with input from the active token. The binding pool nodes that receive convergent input from both types and the token are activated (shown in dark blue). The black lines indicate active connections at the time of encoding. The gray lines indicate connections that exist, but are not active

link between the features and the token. Once encoding is complete, the activated type nodes are no longer necessary, and the sensory system is free to process a different stimulus while holding the encoded stimuli in memory (Bowman & Wyble, 2007; Wyble et al., 2009).

Retrieval of information from the binding pool is assumed to occur in response to a retrieval cue. The first step in retrieval is to translate the retrieval cue into a pattern of activity across the appropriate type layer. For example, in the case of retrieving a color on the basis of its location, we first activate the location type node corresponding to the location of the retrieval cue. Next, this type node sends a retrieval cue to all of the binding pool nodes that are connected to it, triggering them to project their output to the token layer in an attempt to retrieve the token that was originally bound to that location. On average, since these binding pool nodes were strongly connected to that token during encoding, that same token will be more strongly active after retrieval. This process will reconstruct a noisy version of the token activity that was present at the time that particular stimulus was encoded (see Fig. 4).

Next, a decision process determines whether the token retrieval was successful by comparing the most active token to the next most active token. If the token retrieval was successful (see the Appendix for details), that token is projected back through the binding pool to retrieve the associated color. If the most active token node is not sufficiently different from the second most active token, the model will guess by retrieving a random feature value. In order to retrieve the probed feature, the model utilizes a similar procedure as for retrieving a token (Fig. 5). The active token node selectively activates a portion of the binding pool connected to it, and these active binding pool nodes project their activity to the type layer. The retrieved population vector of type nodes is then converted back into a continuous variable by using each type node as a vector in a Cartesian space. These vectors are then added together, and the resultant vector provides both a direction and a length (Fig. 6a; and see the Appendix). The direction is taken as the retrieved color, and the length is assumed to represent an internal metric of confidence in the retrieved value (see below).

Here is a highly simplified example to illustrate the concept of encoding and retrieval within the binding pool. For the sake of simplicity, we are assuming only one type dimension instead of two, although the same general process occurs in either case. Imagine that there are two type nodes (red and green), six binding pool nodes (a, b, c, d, e, and f) and two tokens (1 and 2).

Through hardwired connections, red is connected to nodes a, c, d, e. Green is connected to nodes b, c, d, f. Token 1 is connected to nodes a, b, c, d, and token 2 is connected to nodes c, d, e, f. Encoding a link between red and token 1 would cause nodes a, c, d (i.e., the subset of nodes connected to both of them) to increase their activation by 1.0. To retrieve the type linked to token 1, those nodes that are connected to it and are active (i.e., a, c, d) would project their activity back

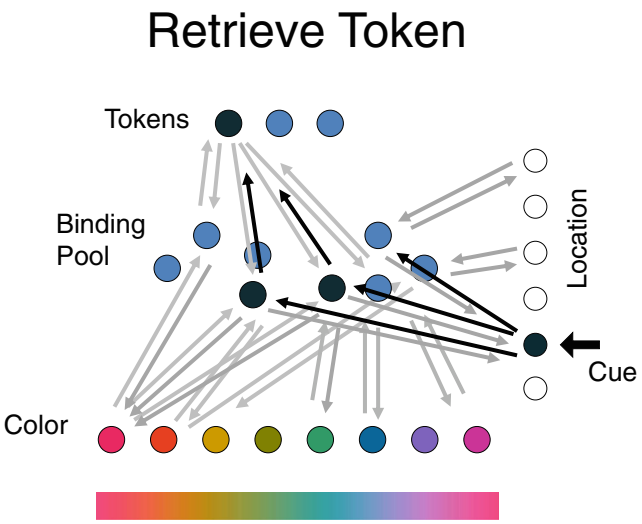

Fig. 4 In order to retrieve the probed feature from the cued feature, the model first retrieves the associated token representation. A retrieval cue activates a type node (in this case, corresponding to a location), which projects to the binding pool. The convergent input from these binding pool nodes reactivates nodes in the token layer, and, on average, the token that had been originally bound to this location will receive the most activity. The retrieved token is then used to retrieval the probed feature (i.e., color; see Fig. 5)

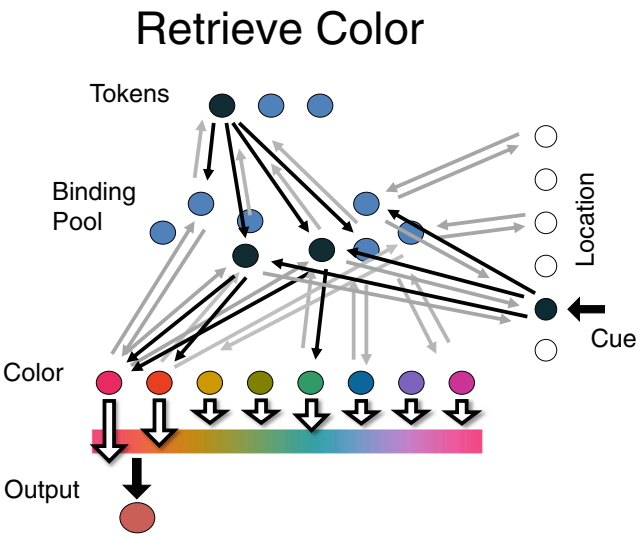

Fig. 5 The retrieved token, in conjunction with the location cue (Fig. 4), is then used to retrieve the color feature bound to that token. The white arrows indicate the magnitude of retrieved activity of the type layer in the form of a population vector. The mean of the population vector corresponds to the retrieved feature (see Fig. 6a and the Appendix for more detail)

down to the red and green types. Red would receive 3.0 activation units, and green would receive 2.0 activation units.

Likewise the model could reconstruct which token was associated with red by activating those nodes that are connected to red (again, a, c, d). These nodes would project their activity up to the tokens so that token 1 would receive 3.0 activation units and token 2 would receive 2.0 activation units.

This simple example illustrates the underlying process of encoding and retrieval within the model; however, the model that we use has Ω=800 binding pool units, which provides a representational capacity that matches the performance subjects on the tasks we fit below.

Retrieval confidence

Because retrieval of a single memory trace occurs across a set of type nodes, the variability of those type nodes provides an inherent metric of the quality of the memory trace, and this can be interpreted as a confidence measurement. This variability can be quantified as the length of the resultant vector when the average across the type nodes is taken to reconstruct a continuous value (Fig. 6b). This confidence metric is highly correlated with the true error in the reconstructed value and, thus, can serve as a measure of confidence. This confidence metric provides a secondary threshold that can be used during change detection as described below.

Encoding and retrieval of multiple items

When multiple items are presented on the display, it is assumed that the visual system assigns tokens to each discrete

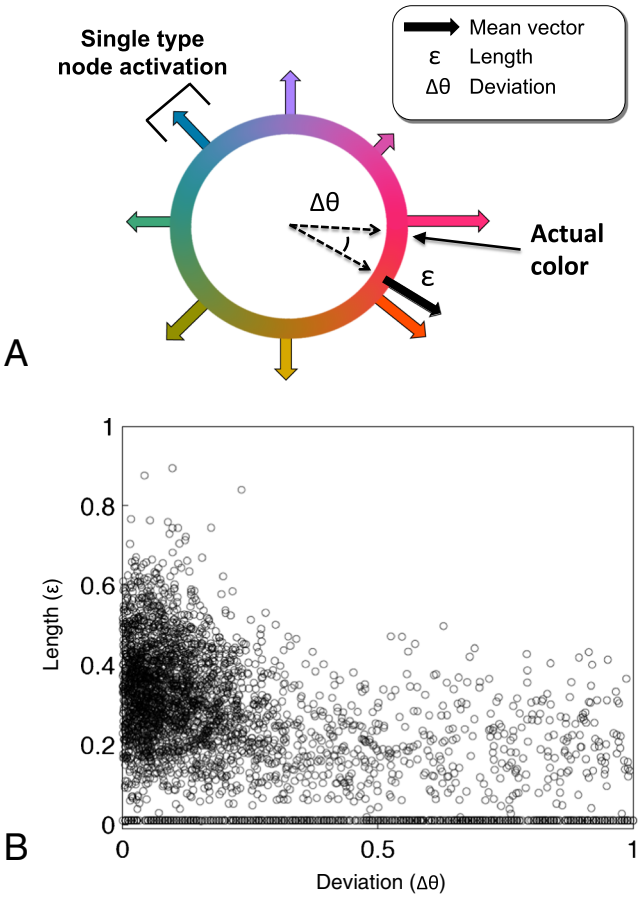

Fig. 6 Illustration of the properties of the retrieved mean vector. **a** The length of each arrow around the color wheel corresponds to the activation level of a type node. The vectors are added together to produce a composite vector that has a length and an angle. This figure illustrates the deviation ($\Delta\theta$) of the retrieved color from the color of the original stimulus. This angle is interpreted as the retrieved feature value, and the length of the vector (ε) is interpreted as retrieval confidence. **b** A scatterplot of the relationship between the length and deviation for trials of set size 4. Note that there is a cluster near zero deviation that corresponds to a correct retrieval of a token and the corresponding feature. There is also a more broadly distributed band of responses at a lower confidence level, corresponding to trials on which the item that was a token was retrieved but the corresponding color value had not been encoded. Finally, at the very bottom of the scatterplot, the line of dots indicates cases in which token retrieval failed entirely and the model guessed randomly, with a confidence of 0. In this plot, deviation values, which normally range from 0 to 180, have been normalized to the range [0 1]

stimulus. It is further assumed that during encoding, the visual system samples information from different objects serially.[5] Each sample produces one encoding, as described above, by activating a token corresponding to the object and the type nodes that represent the pairing of the location and a feature of

[5] We assume that encoding of multiple stimuli could occur in parallel when stimuli are presented rapidly in sequence (i.e., at 100-ms separation) at the same location. This is the conclusion of earlier models that use the binding pool (Bowman & Wyble, 2007; Wyble et al., 2009).

that object. The number of items sampled per trial varies, reflecting presumed fluctuations in the attentional state of the subject (Sims et al., 2012). Importantly, the tokens have overlapping projections to the binding pool, which means that they share binding pool nodes. As a consequence, storing a link to a given token will affect links bound to other tokens, and the size of this interference will grow as the number of stored links increases. Therefore, the retrieved feature value of any stimulus that is stored in the binding pool will be affected by all of the other links stored in the binding pool. When averaged across trials with randomly varying feature values, the result is a distribution that has increasing variance as set size increases (Fig. 7). Note that this increase in variance is apparent only for a no-change condition, in which the deviation is measured between the original stimulus and the retrieved stimulus (Fig. 8). For a changed probe stimulus, the new value is drawn randomly, and therefore, the deviation between the probe and the retrieved value is also uniformly random, no matter how precisely the original item was stored.

Model simulations

Computational models are best used when they are fit against existing data and then used to generate testable predictions using the same parameters. We will classify three types of data for this purpose.

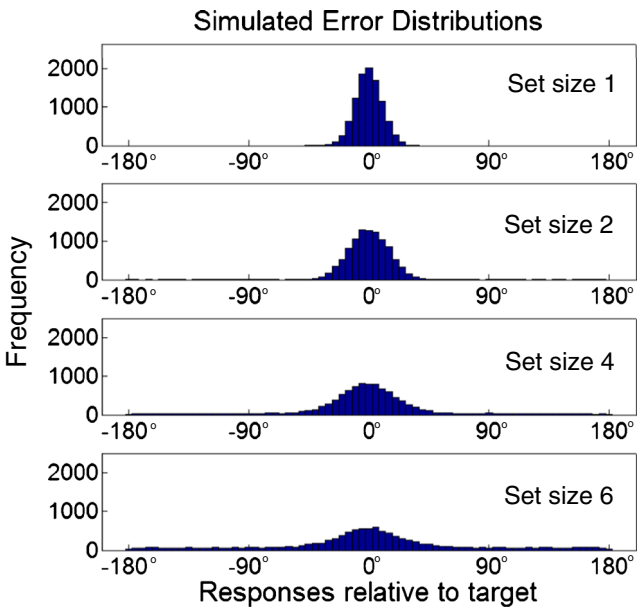

Fig. 7 Histograms of simulated, retrieved color value in degrees, centered at 0° relative to the color that was encoded, taken from the retrieved population mean. This and each further simulation (unless otherwise noted) were collected over 10,000 trials. From the top to the bottom, the histograms indicate set sizes of 1, 2, 4, and 6

1. *Fitted data* are those that the model can simulate accurately because we have parameterized the model to exhibit them. These parameter values are then used to produce simulations of the inherent properties and predictions.
2. *Inherent properties* are known properties of VWM that the model exhibits as a natural consequence of its design.
3. *Predictions* are simulations generated by the model for experiments that have not yet been run. Specifying these predictions in advance of collecting the data ensures that the predictions are genuinely *de novo*.

Fitted data

We fit the model against two data sets: change detection as demonstrated by Keshvari et al. (2013) and continuous report by Bays et al. (2009). To fit the data listed below, there are 11 parameters that are adjusted using a grid search that calculates the mean squared error at each parameter combination. These two data sets will be fit with one parameter set, and those parameters will then be used to simulate the inherent properties and the predictions.

Change detection performance decreases as set size changes

In change detection tasks, an array of stimuli are shown to a subject, removed for some time, and then replaced with a second array that could contain one change that is to be detected. In the model, this task can be simulated by retrieving each of the items originally stored in the array and comparing them against their spatially corresponding elements in the probe display. In addition to the color value, the confidence score associated with each item is also retrieved. In order for the model to produce a "change" response, the retrieved value must be sufficiently different from the original color, and the confidence of the retrieval must be sufficiently high.

Data from change detection experiments can be analyzed in terms of both hits and false alarms, as well as by the proportion of hits relative to the magnitude of a change along a feature dimension, such as color. The hit and false alarm data illustrates the loss of change discriminability as set size increases (Fig. 9a). The analysis by change magnitude illustrates that change detection is easier for larger changes (Fig. 9b). In the binding pool model, the latter effect is a natural outcome of the loss of precision with increasing memory load, because the storage of more links within the binding pool reduces the chance that a small change will be detectable as a result of the reduced precision. Furthermore, as the number of items increases, the chance that an item will be stored at all decreases, which reduces the chance of detecting a change of even the largest magnitude.

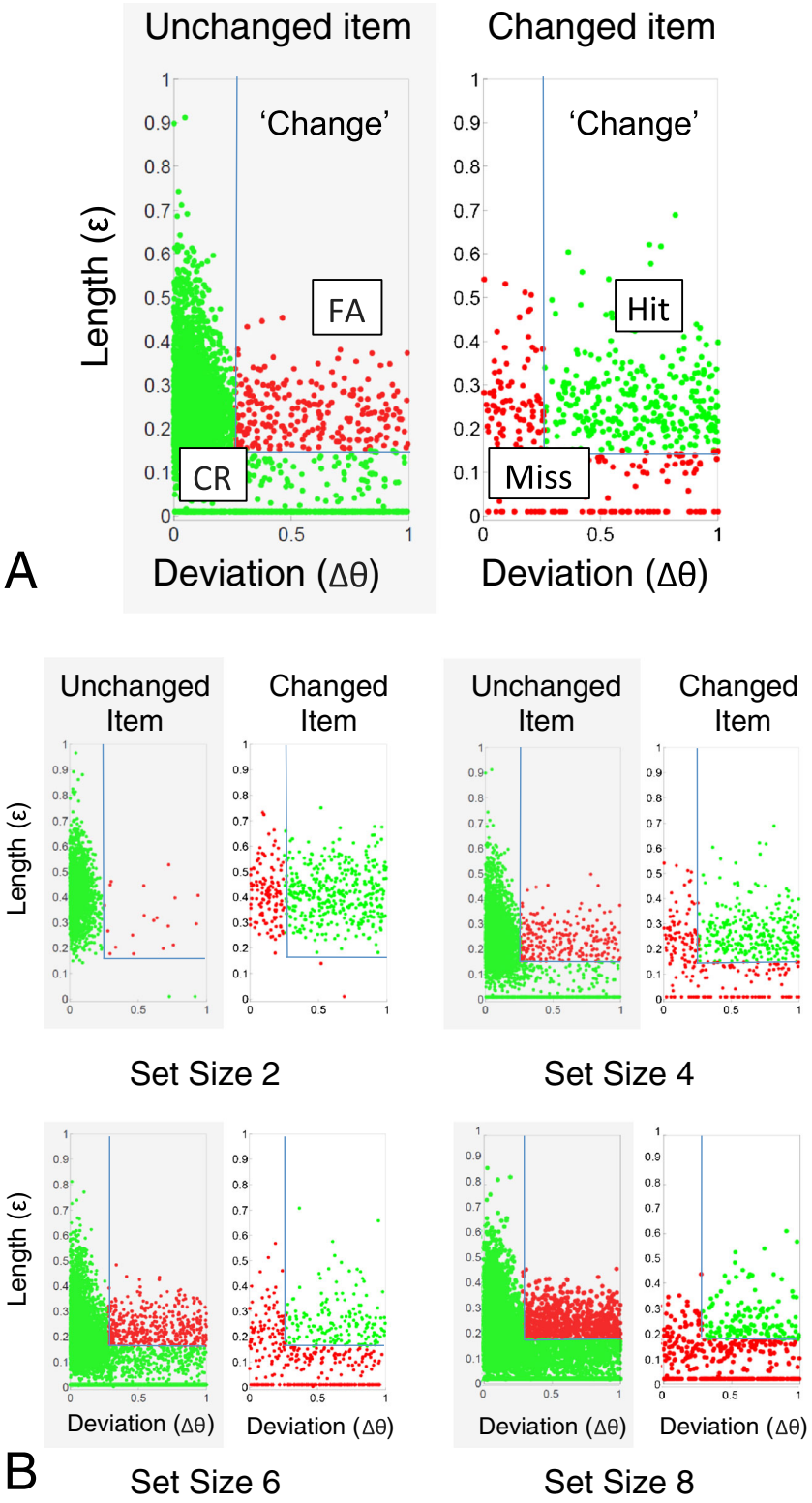

Fig. 8 Scatterplots showing the relationship between the length of a retrieved vector (i.e., confidence) and the deviation between the retrieved color and the actual color at each location in the probe display. **a** The left scatterplot indicates the response for an unchanged item in the probe display, while the right plot indicates the response for a changed item. The deviations are uniformly distributed for the changed item because the changed feature value was selected from a uniform distribution. The thresholds for deviation and length are indicated as lines, and green dots indicate correct responses (CR = correct rejection, FA = false alarm). This example is for set size 4. **b** Scatterplots for set sizes 2, 4, 6, and 8. Note that as set size increases, the confidence decreases, and the deviation increases. Also, the thresholds change systematically as a function of set size according to the parameters (see the Appendix for details)

Memory precision decreases as set size changes

As the number of elements stored in memory increases, the ability to accurately report a specific feature value of the items in a continuous report task decreases. For example, if shown a patch containing a particular color, subjects can be asked to indicate the specific hue of that patch on a color wheel. These responses can be scored as errors relative to the true color, resulting in a histogram of errors across trials (Wilken & Ma, 2004). W. Zhang and Luck (2008) further proposed that such

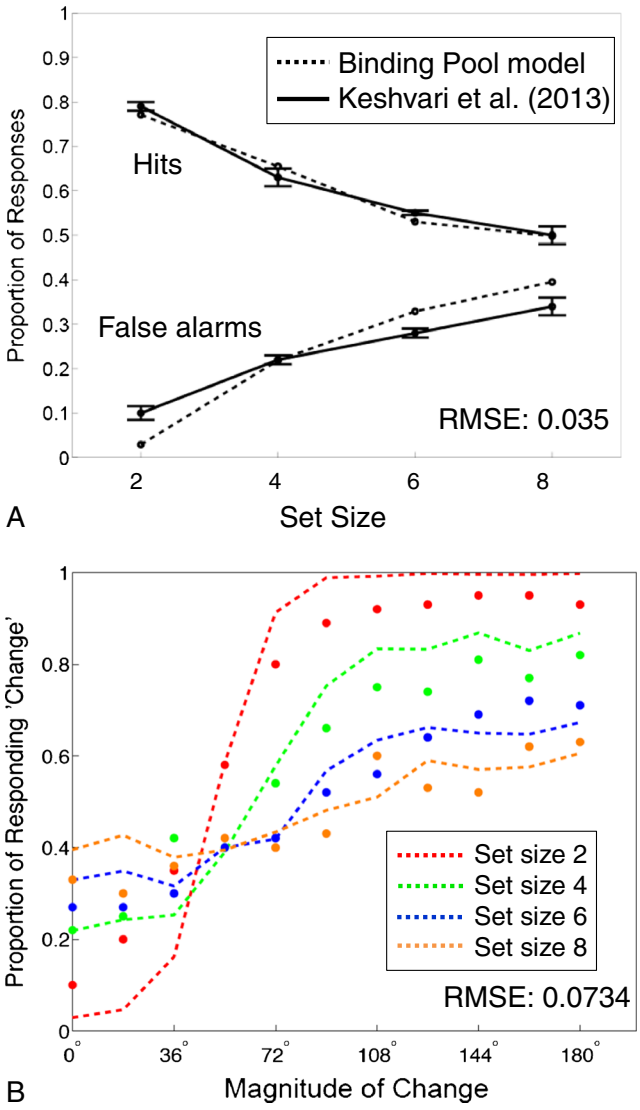

Fig. 9 Change detection simulations. **a** Proportion of hits versus false alarms of the binding pool model compared with data from Keshvari et al. (2013). The RMSE value is the average RMSE of the hit and false alarm rate. In this figure, the largest standard error for simulations is .0071 computed as the averaged output of 15 simulation batches, each of which ran 750 trials. **b** Proportion of reporting a "change" given the difference in degrees between an item in the memory array and the probe display. The actual datapoints from Keshvari et al. are represented as dots. The largest standard error value for any simulated data point is .0268 (error bars omitted for clarity)

responses could be divided into correct retrievals and random guesses through the use of a mixture model, producing separate estimates of the percentage of correctly reported items and the precision of those responses. Bays et al. (2009) extended this idea further by dividing responses into three groups: correct retrievals, swaps (i.e., cases of retrieving the wrong item on the basis of a cue), and random guesses. The overall pattern from such data is that as set size increases, the precision of correctly retrieved items decreases, while the proportion of swaps and guesses increases.

The responses of the model can likewise be divided with a mixture model into the same three response types (see the

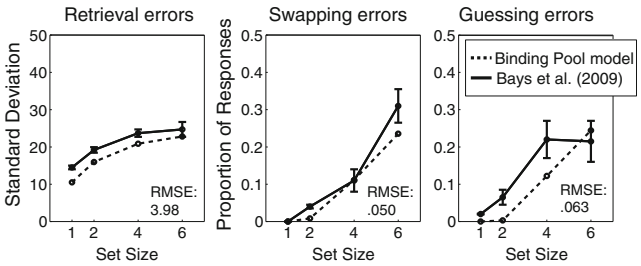

Fig. 10 Continuous report simulation. The model simulations were fitted to data from Bays et al. (2009), in the condition with a stimulus presentation of 100 ms. Across 20 simulated subjects, the largest standard error value was .33 for standard deviation and .001 for swaps and guesses

Appendix for details). For the purpose of fitting the data, we use data from Bays et al. (2009), Experiment 1, in the condition for which the stimuli were presented for 100 ms. In the simulated results the standard deviation of the error distribution of correctly retrieved items increase as a function of set size (Fig. 10). Furthermore, because of the cross talk between stored representations, each stored representation pulls other representations toward itself. This attraction causes some retrievals to be classified as swap errors by the mixture model. For a larger set size, this attraction is increased, increasing the likelihood that an item will be classified as a swap. Finally, guessing occurs when a stimulus is not encoded at all, due to the trial-by-trial variability in the number of encoded items.[6] As set size increases, it is more likely that a stimulus will not be encoded and the model will guess.

Inherent properties

The model exhibits behaviors that match findings already in the literature. These are data that were not used explicitly when fitting the parameters but are, nevertheless, observed as inherent properties of the model.

Retrieval based on nonlocation cues

Retrieval can occur without using the location of the object as a cue. For example, if several oriented, colored bars are presented for encoding, subjects can retrieve the orientation of a particular bar given a color cue, even when that bar is presented at the center of the screen (Bays, Gorgoraptis et al., 2011; Gorgoraptis et al., 2011; van Lamsweerde & Beck, 2012). This capability illustrates that VWM is content-addressable, which means that an item representation has some chance of being retrieved by any of its features. This property is inherent in the design of the binding pool model in that multiple types can be linked to a single token and retrieval of that token can be triggered from any of the type layers.

[6] Another source of guessing errors could be a decay of information within the binding pool, but the loss of information is currently outside of the scope of the model.

Featural cross talk

When presented with two stimuli along the same dimension, such as spatial frequency, the retrieved values of those stimuli will shift their representations toward each other (Huang & Sekuler, 2010). The binding pool model exhibits a similar distortion when multiple items are encoded, and this is a natural consequence of the fact that there is overlap between token connections (Fig. 11). If two items are stored, some of the binding pool nodes linked to each token will be shared with the other token bound in that trial. Therefore, when retrieving item 1, item 2 will also be partially retrieved, and this will affect the retrieved vector representation. Furthermore, this effect is increased in magnitude when the stimuli are farther apart from one another in feature space.

Retrieval precision is improved when variability is decreased

When the variability of one feature in a set of presented items is decreased (e.g., the colors are chosen from a more restricted region of color space), the ability to detect changes in that feature is improved (Lin & Luck, 2009; Sims et al., 2012). Sims et al. considered this as evidence that memory encoding can be represented using an information-theoretic framework in which a predefined number of bits of information are allocated across a number of items. Improved performance in the homogeneous feature condition results from the allocation of those bits across a smaller range of feature values. The binding pool model exhibits the same quality, although it does not use an explicit information-theoretic framework. Instead, this effect emerges because the precision of a retrieved value is improved when all other values of that feature reside in nearby locations in feature space. This is due, in part, to the featural cross talk, which changes as a function of the difference between two stored feature values within a single feature dimension. To illustrate this point, a simulation of the condition described in Sims et al. is shown in Fig. 12.

Binding errors

Binding features to locations creates the possibility for binding errors—situations in which a remembered feature is erroneously linked to the wrong object or location. Such errors represent failures to correctly maintain the index between features and locations and can be observed in the data of Wheeler and Treisman (2002), in which subjects were impaired at spotting a change if it involved a location swap between two items on the screen, rather than replacing those two colors with different colors. The model produces this same pattern in a simulation of these two conditions. This pattern occurs because a retrieved value will be biased toward the locations of other stored values. Therefore, when the probe display replaces a given object's color with a another stored object's color, there is a decreased chance that the change will be noticed, relative to a condition in which that item in the probe display was changed to a feature value that is not currently stored in memory. Thus, the chance of detecting a change is slightly reduced in the location swap condition, relative to the condition in which the colors are replaced with new ones. Because the Wheeler and Treisman paradigm is different from the Keshvari et al. (2013) paradigm that was used for data fitting, in that subjects expect two items to be changed in the former case, new thresholds for change detection are required. Therefore, to demonstrate that the binding effect is robust, rather than a consequence of parameter fitting, the presence of binding errors was confirmed statistically over a range of parameters using a grid-based search of the four change detection threshold parameters (see the Appendix) that compared the proportion of hits between novel-color and

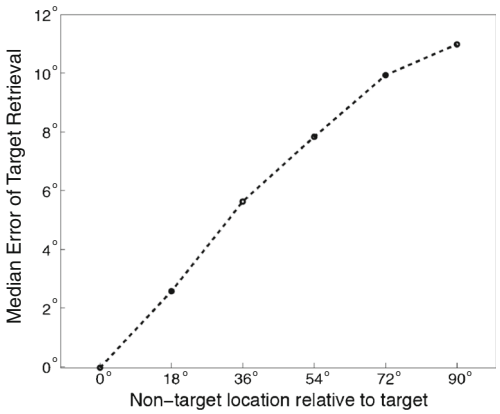

Fig. 11 A simulation of the interference effect originally demonstrated by Huang and Sekuler (2010). Two items are stored in the binding pool, with the first item being fixed at 180° on a continuous scale and the second item presented at various distances (0 to 90 in increments of 18°) relative to it on the feature dimension

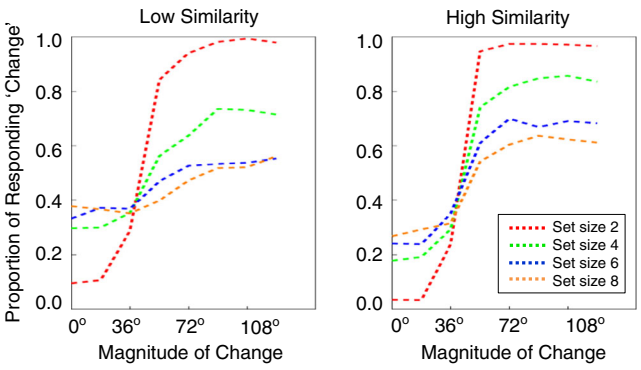

Fig. 12 In a simulation of the effect found by Sims et al. (2012), the model is more precise in being able to spot a changed stimulus value when the items it has encoded are more similar to one another in the retrieved feature dimension

color-swap conditions. With a sampling of 4000 different combinations of threshold parameters, performance in the new-color condition was higher than performance in the color-swap for both set size 3 (sign test: $p < .0001$) and set size 6 (sign test: $p < .0001$).

The effect of confidence on continuous report

Subjects in VWM tasks have an inherent sense of the quality of their memory, and they can use this knowledge to accurately choose items that have higher precision (Fougnie, Suchow & Alvarez, 2012). Retrieval confidence can also be reported explicitly by subjects. In an experiment by Rademaker et al. (2012), items reported with high confidence had increased precision and dramatically decreased proportions of guessing and swap errors, relative to low-confident items.

The output of the model can be subjected to a similar analysis by binning responses into quantiles according to confidence (see the Appendix for details). This analysis produces results that are in good qualitative agreement with those of Rademaker et al. (2012). First, the model simulates an increased proportion of highly confident responses for smaller set sizes (Table 1). Second, the model simulates that precision decreases with decreased confidence and that swaps are relatively rare in the highest confidence bin (Fig. 13). This last point is especially important because it is not certain to what degree the swap errors as found by a mixture model analysis are genuine cases in which the subject reports the incorrect item. In partial answer to this question, the simulations are in agreement with the data that these swap results are primarily from retrieval noise, rather than highly confident errors. Indeed, the model simulates that subjects' awareness of the accuracy of their responses is actually quite good and that their highly confident responses will almost always be centered at the correct location in feature space at the time of retrieval for the kind of tasks simulated here. However, as we describe below, the model does predict that highly confident swap errors should be observed in certain experimental contexts.

Table 1 Relative proportions of responses for set size given low, medium, and high confidence

Confidence	Set Size 1	Set Size 2	Set Size 4	Set Size 6
Low	0	.03	.51	.72
Medium	.01	.62	.45	.25
High	.99	.35	.04	.03

Note. This simulation and that shown in Fig. 13 were simulated with 30,000 trials. Note that the proportion of high-confident responses drops dramatically with set size.

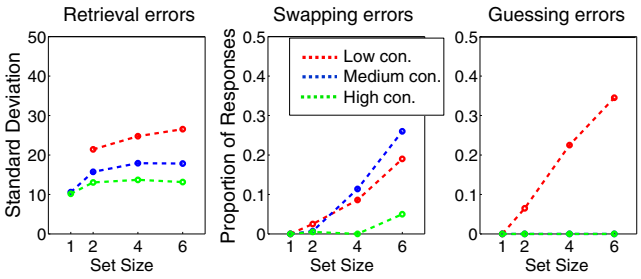

Fig. 13 A simulation of a continuous report task with responses divided by simulated confidence. Confidence levels are binned by the length of the retrieved vector into tertiles. As was found by Rademaker et al. (2012), high confident responses are rarely swaps or guesses. In this figure, "con." refers to confidence

Predictions

To test the explanatory limits of the model and to generate new insights into the mechanisms underlying retrieval, we list a series of testable predictions concerning the interaction of similarity, repetitions, and confidence in continuous report tasks.

Prediction 1: Loss of precision due to set size is a product of interference with other stored items

Simulations of memory precision for items have found strong support for variable precision encoding, in which the stored representations vary in their precision from trial to trial as a function of set size, among other influences (Fougnie et al., 2012; van den Berg et al., 2012). In the binding pool model, there is an inherent loss of precision as more items are stored, but this loss of precision is not due to variations in precision at the time of encoding. Rather, the loss is primarily due to the interference between stored binding links that is expressed at retrieval. Consequently, the model predicts that if subjects can be successfully instructed to forget one or more items in a memory set, the precision of the remaining items would increase to nearly the levels they would have had if the now forgotten items had not been encoded in the first place.[7] Forgetting is accomplished in the model by resetting the binding pool nodes connected to a to-be-forgotten token back to an activation of 0. In simulations of 11 pseudosubjects with 5,000 trials each, retrieving one of two stored colors produced standard deviations of 14.95 (*SE* of 0.09). Eliminating the memory trace of one of those items allowed the nonforgotten

[7] Williams, Hong, Kang, Carlisle & Woodman (2013) demonstrated an intentional forgetting effect very similar to this prediction in a directed forgetting paradigm. In their task, participants stored 2 color patches, and were then asked to forget one, which improved performance on the remaining patch, although not to the same level as if the forgotten item had not been stored, as predicted here. We learned of Williams et al. (2013) after this paper had been submitted for publication, so we retain this simulation as a prediction rather than an inherent property.

item to be retrieved with a standard deviation of 12.1 (*SE* of 0.13). Storing just a single item resulted in a standard deviation of 10.26 (*SE* of 0.13). Note that storing two items and dropping one would never produce the same standard deviation as storing just one item, because of the overlap between tokens in their connections to the binding pool. Thus, the model also predicts that it is impossible to completely erase a single item while leaving others intact; there will always be a small amount of residual interference remaining from the erased item.

Prediction 2: Memory precision is reduced when storing more complex objects

The size of the binding pool provides a fundamental limit on the amount of information that can be stored in memory regardless of whether that information is grouped within an object or distributed across objects. Therefore, adding new feature dimensions to the stimuli in a VWM task produces stronger constraints on which binding pool neurons can be used to store information about that object. Consequently, storing information about stimuli with more features will produce a loss of precision about each of the stored features, relative to simpler objects, especially if the subject does not know in advance which features will be tested. The empirical data pertaining to this prediction are complex and somewhat contradictory. Luck and Vogel (1997) found that memory for complex items (defined as objects with multiple features, such as color and orientation) were no worse than memory for single-feature objects. On the other hand, more recent studies by Fougnie et al. (2012) and Oberauer and Eichenberger (2013) have found evidence that adding extra features to an object reduced precision and change detection accuracy, respectively.

The binding pool model suggests that a likely reason for this discrepancy is the amount of information stored per feature dimension. In the model, the specificity of a stored representation within a feature dimension can be modulated by adjusting the sparsity of a representation within a type layer. A sparser type representation allows a more specific feature value to be encoded, but it also reduces the number of binding pool nodes that can contribute to the retrieval precision of information about that object across all other feature dimensions. To see why, consider that only binding pool units that are connected to an active type node are able to participate in encoding information about an object. Therefore, there is a predicted trade-off between encoding precision within a single dimension and precision across all other dimensions.

This explanation suggests a specific reason why Luck and Vogel (1997) failed to find an effect of object complexity, which is that the subjects encoded some features, such as shape and texture at a coarse-grained level, even for the simple objects. For example, even when subjects are not expected to report on the shape of a colored square, there is likely to be some shape information that is stored automatically. Thus, subjects may have been able to perform adequately in a change detection paradigm without sacrificing performance when there are very few possible values that a feature might have by relying on this coarse-grained representation. However, if subjects are required to retrieve a more specific feature value to perform a memory task (e.g., one of eight possible values vs. one of four possible values), the model predicts that this requirement will reduce the quality of the stored value for all other feature dimensions. In support of this explanation, both Fougnie et al. (2012) and Oberauer and Eichenberger (2013) placed greater demands on the specificity of retrieved representations than did Luck and Vogel (1997).

Prediction 3: The effect of similarity and repetition on continuous report

The binding pool model encodes similarity within the type dimension and, therefore, can make predictions about the effect of similarity on performance in continuous report tasks. In fact, the model can simulate a gradient of similarity up to and including complete overlap (i.e., repetitions). In this prediction, the similarity between colors in a memory array of size 4 is adjusted so that items occur at regular intervals in the color wheel from 0° separation (i.e., four copies of the same color) up to 90°. As a baseline comparison, the retrieved distribution of values is compared with encoding of a single color.

The simulations reveal two predictions. First, as the set of colors becomes more similar, the retrieved color becomes more precise (Fig. 14). Second, repeating the same color 4 times at encoding produces precisions that are substantially more variable than encoding a single copy of that color. The increased variability of encoding four copies of the same color is due to the

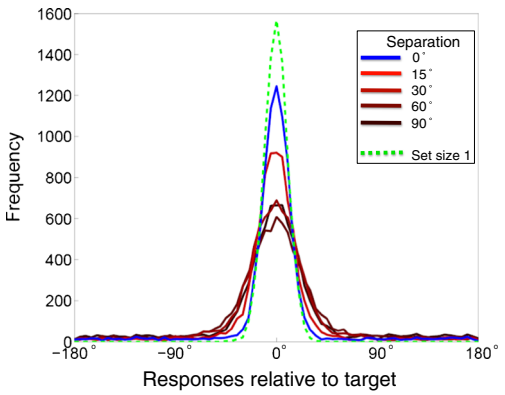

Fig. 14 A simulation of response precision as a function of the similarity of stored stimuli. The set size is held constant at four, and the stimuli are distributed in feature space at fixed intervals from 0° to 90°. As a baseline comparison, the green line represents response precision when only a single item is encoded. Note that the precision of a repeated item is worse than the precision of one encoded item, and this is due to the probability of encoding failure in the repeated item case

trial-to-trial variability in the number of colors encoded, which causes the model to fail to encode some of the colors on some trials, and such encoding failures produce random guessing.

A failure to confirm this prediction (i.e., finding that four repeated colors have the same precision as encoding a single color, or perhaps an even higher precision) would highlight the importance of particular kinds of ensemble representations that are not included in the current model. For example, it may be the case that a repetition is treated like a feature value, such that a repeated color is encoded by creating a single token that is bound to the repeated color, the repetition feature, and a number of different locations. An analogy to this theory can be found in the language production literature, in which it is thought that double-letter is a feature that can be bound to the wrong letter, such that the word "tomorrow" is frequently misspelled as "tommorow" (Caramazza & Miceli, 1990).

Prediction 4: The influence of similarity on confidence

The binding pool model also predicts that since retrieval confidence is determined by the variability of the retrieved type layer, similarity should increase confidence. Furthermore, four repeated copies of the same item should lead to less confidence in report than a single item (Fig. 15). A failure to confirm this prediction would again point to the existence of mechanisms for explicitly encoding repetitions.

Prediction 5: The effect of repetitions and similarity on report of other items

The binding pool model represents the interference between items that are stored producing both general decrements in precision (Wilken & Ma, 2004) and attraction of one retrieved representation toward another (Huang & Sekuler, 2010). Consequently, the binding pool model is able to make predictions about how the relative proximity of two items to one another will affect a third item.

This prediction will also make use of the model's ability to represent and encode repetitions. In this prediction, three stimuli (referred to as A, B, and C) with feature values on a linear dimension, such as spatial frequency, will be encoded. The dependent value will be the retrieval of C, while the proximity of A and B is varied. At one extreme, A and B are relatively far apart from one another, while at the other, they are completely overlapping (Fig. 16). However, in all cases, the mean of A and B remains the same.

The model makes two predictions about this paradigm. First, as A and B become more proximal, the magnetic effect they produce on stimulus C increases, even though the mean of A and B remains constant. Second, there is no discontinuity in this function as A and B become repetitions. Thus, the model effectively predicts a null effect of repetitions on a third stimulus. As in predictions 4 and 5, finding a qualitatively different result when A and B are repetitions would suggest that repetitions are encoded explicitly.

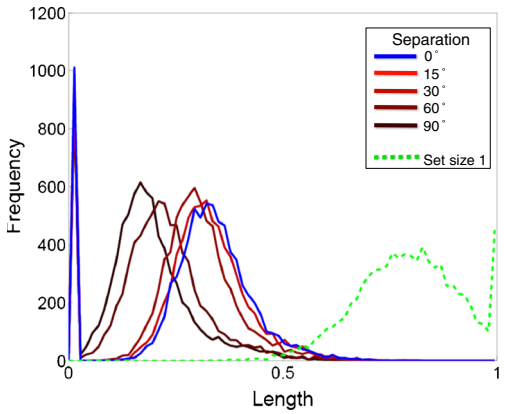

Fig. 15 Histograms of length values (i.e., simulated confidence) pooled over multiple trials for various sets of item similarity with set size 4.The colored lines represent the same degrees of item separation as in Fig. 14. Note that the model predicts that confidence will be much greater for a single item than for four items, regardless of whether they are similar or repeated

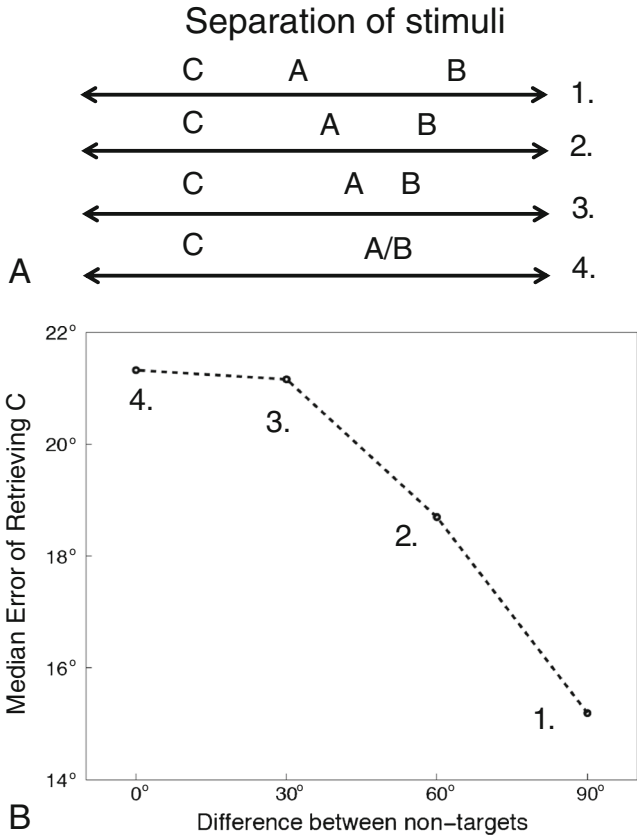

Fig. 16 Illustration of the experimental paradigm for prediction 5. **a** A, B, and C represent stimulus features for three encoded objects on a continuous dimension such as spatial frequency. The proximity of A and B in feature space are varied up to and including the case where the two stimuli are identical (i.e., #4). The effect of their interference is measured on stimulus C. **b** The predicted magnet effect on C is depicted. Numbers 1 through 4 indicate which data points in (**b**) correspond to scenarios depicted in (**a**)

Prediction 6: Highly confident swap errors are possible

In the specific experimental paradigms being simulated here, the binding pool simulations suggest that swap errors occur primarily when confidence is medium or low, and this result agrees with data from Rademaker et al. (2012). These low-confident swap errors occur when the correct token is retrieved by a location cue but the retrieval of the color associated with that token is strongly pulled toward another stored feature value. This conflict between multiple representations adds variability to the type layer, which reduces the confidence measure.

However, the model predicts that, in certain circumstances, it should be possible for subjects to make swap errors with high levels of confidence due to the retrieval of the wrong token. Thus, when the second step of the retrieval process commences, this incorrect token is used to retrieve the feature associated with it, and the model might confidently generate the wrong feature. Such errors would occur in situations when two stimuli are very proximal in the feature dimension that is used to cue retrieval of the stimulus. For example, in the Rademaker et al. (2012) experiment, if stimuli were presented sequentially to avoid configural effects and were presented at closer locations, a significant proportion of highly confident swaps should be observed.

General discussion

The binding pool model provides a constructive proof that it is possible to create, using simple neural elements, a hybrid model of VWM with a resource pool indexed by tokens that exhibits behaviors that are characteristic of human subjects in a variety of tasks. Tokens, which resemble the classical notion of slots, are bound to features and play an essential part in storing and retrieving information. This model represents a significant advance over previous neural models of VWM in its ability to explain data from both change detection and continuous report tasks using a single set of parameters.

As the size of a simulated memory set increases, the binding pool model exhibits a quantitative match to the patterns of interference observed in human behavior. In the model, this relationship between set size and performance results from two distinct limitations within the model. First, we simulate a trial-to-trial variability in the amount of items that are encoded on a given trial, as a function of the short encoding duration used in many VWM tasks. Thus, as set size increases, there is an increasingly large chance that any randomly selected item will have not been encoded, which increases the proportion of guesses in a continuous report task and the proportion of misses in a change detection task. The second limitation is the interference between links stored within the binding pool: As the number of stored links increases, so does the interference. This interference also manifests directly as the within-trial "magnet" effect observed by Huang and Sekuler (2010). This interference also produces a decrease in precision and an increase in swap errors with set size in continuous report tasks. In change detection tasks, this interference produces a reduced ability to correctly discriminate between change and no-change trials, as well as a flatter curve of change detection probability as a function of the magnitude of a change (Keshvari et al., 2013).

The binding pool model also provides an inherent measure of confidence about the quality of retrieved representations by measuring the variability of type node activity after retrieval. The greater the variability is (i.e., the more type nodes are co-active), the lower the confidence is. This derived sense of confidence is suggested to play a role in determining whether an item has changed in a change detection task. This simulated measure of confidence is able to accurately predict whether a given response is likely to be accurate, in terms of both precision and the likelihood of guesses or swap errors. This property of the model stands in agreement with recent data that compared subjective confidence ratings with objective accuracy (Rademaker et al., 2012).

Two sources of retrieval errors

Another contribution of the binding pool model is to characterize the multiple forms of errors that may arise in the retrieval process. One source of error is a failure to retrieve the correct token, which gives rise to highly confident swap errors. Another error is that of a noisy retrieval process given that the correct token was retrieved. These two types of errors are predicted to occur at different rates depending on the confusability of the cue that elicits retrieval. Furthermore, the model predicts that confidence measurements may be a clear way to discriminate between these two types of errors.

Another likely source of error, which is not addressed in the model, is decay of active memory traces (Fougnie et al., 2012). Such decay could be incorporated either by having active binding pool nodes lose their attractor states spontaneously or by allowing activation values of binding pool nodes to drift over time. Either implementation would cause a gradual erosion of precision with a time constant dependent on the rate of change.

Wholistic encoding

Treisman and Schmidt (1982) computed the probability that multiple features from the same object would be stored together, and they found evidence of independence, which is to say that correct retrieval of one feature of an object made it no more likely that another feature of that same object would be reported as belonging to it. Subsequent explorations of the same question have found the same answer (Bays, Wu & Husain, 2011b; Vul & Rich, 2010). The binding pool model does not inherently predict either dependence or independence between co-occurring features, since this is a property of how the objects and their features are selected for encoding. Using the architecture described above, independence between

features could be simulated by assuming that multiple features from an object are sampled independently as feature/location/token triplets. Thus, an oriented colored bar would be encoded by linking its orientation/location pairing to a token and then linking color/location to the same token. A random sampling process across all possible pairings of feature and location for all objects in a display would provide the observed independence between stored feature values. In the simulations presented here, this issue is not directly relevant, since we are simulating tasks in which there is only one explicit feature/location pair per object. Thus, this is an issue left for future work. However the model does make a clear prediction that the more features that are encoded per object, the lower the precision will be. This prediction can be surprisingly difficult to test, because it is not straightforward to deduce what all of the features of an object are. For example, when a square color patch is encoded, even when subjects are not asked to report the shape explicitly, some amount of shape information is probably encoded to at least a coarse degree. Surprise memory tests may be one means to address such questions about what information is incidentally encoded beyond what is explicitly required by the task (e.g. Rock, Linnet, Grant & Mack 1992).

Comparison with other models

There are numerous accounts of VWM and WM at varying levels of mechanistic specificity. While it is beyond the scope of this article to review this body of work exhaustively, there are several comparisons that are particularly revealing in terms of clarifying theoretical positions on the underlying structure of VWM.

Some models characterize VWM performance by fitting distributions that describe the precision of stored items, rather than fixing the precision at a specific value (Fougnie et al., 2012; Keshvari et al., 2013; Sims et al., 2012; van den Berg et al., 2012). The model by Sims et al. suggests that there is variability in the number of encoded items from one trial to the next. On this point, our models agree, because we find that in fitting the data from Bays et al. (2009), it is essential to include substantial variability in the number of encoded items from trial to trial to account for the pattern of guesses as a function of set size. Sims et al. also found that an information-theoretic encoding scheme is necessary to account for the changes in retrieval precision when the variability of the encoded items is varied. Our model demonstrates this behavior as well, even without an explicit information-theoretic allocation of bits. In the binding pool model, this interaction between featural similarity and retrieval precision is a natural consequence of the mutual interference between stored pieces of information: Items that are more different from one another produce stronger interference as measured by the paradigm described by Huang and Sekuler (2010).

Another set of statistical VWM models find superior matches with observed data when the precision distributions are allowed to vary as a function of set size (Keshvari et al., 2013; van den Berg et al., 2012). Such models behave in a similar manner as a hybrid resource/slot account, such as this one, in the sense that there is a specific number of items that are encoded, which may be less than the number of items on the display, and the precision of encoding is stronger when fewer items are encoded. In the binding pool model, such variability is inherent in the memory storage mechanism because multiple pieces of information share the binding pool and mutually interfere with one another at the time of retrieval. The more items that are stored on a given trial, the lower the precision will be for all of them. Furthermore, in the binding pool model, the retrieved precision varies both between trials as a function of set size and between items within a trial. This variability arises from three sources. First, there is trial-to-trial variability in the number of items encoded. Second, there is variability in the random initialization of connections between different layers of the model that affects the degree of interference between stored representations (e.g., if two tokens happen to share a greater fraction of the binding pool, they will exhibit greater interference). This form of variability is expected to occur only between subjects. The third source of variance arises from the specific combinations of stimulus values that are randomly chosen, because the amount of interference between two or more items is a product of their similarity in feature space. This form of variability would occur between subjects and would reflect their varying levels of familiarity with the stimulus dimension(s) being tested. Thus, the binding pool model is in agreement with accounts of variable precision and provides a mechanistic explanation as to how some of that variability might arise.

Another class of models has been using a similar method of fitting distributions to understand the structure of memory encoding in terms of ensembles of features. In the binding pool model, features of an object are represented at the simplest possible level: We use a separate population code for each feature of each object. In contrast to these simple representations, Brady and Alvarez (2011) and Brady and Tenenbuam (2013) proposed that representations are hierarchical in the sense that some representations correspond to individual object features and other representations correspond to groups of object features. For example, if groups of red and blue circles of varying sizes are stored in memory, there are memory traces for the individual circles, but also superordinate representations of the average sizes of the two groups of circles. Orhan and Jacobs (2013) argued that VWM attempts to discover the underlying structure of the items being stored and represents information as clusters of possible groups of items along with the probabilities of these clusters being accurate representations.

The binding pool model has no explicit representations of higher order ensembles, although it can, nevertheless, explain some of the data typically associated with ensembles. In this article, we simulate the within-trial magnet effect described by

Huang and Sekuler (2010) as an inherent property of a model in which multiple items are stored within a shared pool of neural resources. Such interference arises despite the fact that the items are represented as independent features at the time of encoding. Thus, the model clarifies that some ensemble effects can be accounted for by a model in which neural resources are shared between items. However, the binding pool model currently has no ability to accommodate some of the higher order ensemble effects involving complex configurations of items (e.g., Jiang et al., 2000). Looking forward, incorporating feature representations that use hierarchical ensembles (Brady & Alvarez, 2011), probabilistic clusters (Orhan & Jacobs, 2013), or two-dimensional templates (Brady & Tenenbuam, 2013) into the binding pool model has the potential to build neurally plausible ensemble models of VWM.

In addition to models of the statistical properties of VWM representations, there are other models that focus on possible neurophysiological mechanisms of information storage. One such account (Wei, Wang & Wang, 2012) proposes that each item in memory is stored as a distinct bump-attractor (cf. K. Zhang, 1996) and multiple such attractors can coexist within a feature domain. This model explains the limits of VWM as a function of attractors that inadvertently combine together when memory is overloaded. This model provides an elegant explanation of VWM capacity as an emergent property of neural dynamics in a feature layer (similar to our type layer). A similar idea, recently proposed by Franconeri, Alvarez and Cavanagh (2013), describes memory capacity limits in terms of cortical maps. Selection of objects within those maps requires the inhibition of surrounding areas. Therefore, there is a natural limit on the number of items that can be encoded as a function of the size of these maps, because the inhibited surrounds of multiple items collide as the system becomes overloaded.

With respect to the terminology we use here, both of these models store information at the type level. That is to say, they use a localized, sparse, nonoverlapping representation in which there is a one-to-one correspondence between clusters of activated neurons and the items being stored. The binding pool model, in contrast, has distinct neural structures for maintaining information (e.g., the binding pool) and for representing features in a form that is accessible to other brain regions (e.g., the type layers). When other brain areas need to access information stored in the binding pool, it must be transferred back to the type layer first (see also Oberauer, 2009, for a similar account).

We suggest that this distinction between storage and representation is crucial because these other accounts (Franconeri et al., 2013; Wei et al., 2012) would have difficulty exhibiting some of the key properties of working memory that we discuss above. For example, such models have no ability to store repetitions because, as two items grow more similar, it would be predicted that one of the items would disappear from memory or the two would merge into a single representation that loses any information about the existence of two discrete items. Contrary to this prediction, data show that increasing similarity between stored items actually improves memory performance (Sims et al., 2012). Furthermore, while the model of Wei et al. provides a straightforward account of storing multiple single-feature items, without a binding mechanism, it lacks the ability to store conjunctions between features and other features or features and locations.

The role of tokens in cognitive function

In the binding pool model, tokens provide a form of indexing that can store representations of complex objects, not unlike the idea of object files (Treisman & Gelade, 1980). We speculate further that these tokens fulfill a crucial role in allowing higher order cognitive functions to make use of the representations stored in VWM. Assuming that such higher order functions might include, for example, the ability to compare stored representations against one another or to recombine elements of stored representations into higher order representations, it would seem important that these higher order functions can easily access stored information regardless of the specific features. The binding pool provides an answer to this functional requirement in that the ability to bind tokens to arbitrary features means that tokens serve as pointers to collections of features. Superordinate levels of cognition could operate on the tokens themselves and, thereby, have indirect access to the types that are bound to the tokens. This level of abstraction allows a level of computational flexibility that begins to resemble symbolic forms of processing. We suggest that this capability is crucial for bridging the gap between visual pattern information and higher order forms of cognition.

Conclusion

The binding pool model that we describe here provides one potential theory about how working memory resources are distributed across multiple items. The model suggests that a hybrid resource model is computationally feasible, given simple neural mechanisms. Moreover, the model specifies a number of issues that any comprehensive model of VWM would have to address, including such capabilities as repetition encoding and content addressable memory. The contribution of the model to the debates regarding slots, resources, and ensembles is to provide clarity, in the form of a formally specified implementation of a hybrid slot-resource theory. This implementation can generate predictions that will drive further empirical inquiry in pursuit of a more finely specified and more strongly constrained theory of VWM.

Author Note G.S. was responsible for a majority of the computational work and graphics. B.W. was responsible for a majority of the writing and the original conception of the model. The authors would like to thank Howard Bowman for his input during the development of this model that has been ongoing for several years. We would also like to thank Rich Carlson, John Collins, Mark Nieuwenstein, Adam Reeves, Michael Mozer, Jeremy Wolfe and Christopher Stevens for helpful comments. This study was supported in part by an NSF grant (BCS-1331073) to B.W.

Appendix

This section provides a formal description of the model.

Neuronal dynamics

All simulated neurons are rate-coded and time steps are discrete, such that neurons update their activation levels in a single step at encoding and retrieval phases, as specified below. Trials are completely independent of one another.

Here is the pseudocode for storing and retrieving information in a single trial:

1. Set up connections/determine encoding capacity
2. For each stimulus to be encoded
   a. Stimulus selection
   b. Convert feature values to type representations
   c. Activate token
   d. Increment activation of binding pool neurons receiving input from both types and token
3. To retrieve a stimulus
   a. Convert cue feature value to type representation
   b. Retrieve token
      i. If token retrieval is successful, retrieve associated type representation
      ii. Otherwise, guess by randomly selecting a retrieved feature value.
   c. Retrieve type representation
   d. Convert retrieved type representation to probed feature value
4. For a change detection task
   a. For each stimulus in the probe display, compare retrieved feature value to probe feature value
      i. If confidence and deviation thresholds are exceeded, then the model reports a change for this stimulus
   b. If at least one change was detected, then the model classifies this trial as a change trial
   c. Categorize this trial as a hit, miss, false alarm, or correct rejection.
5. For a continuous report task
   a. The set of retrieved feature values from a group of trials is passed through a mixture model (Bays et al., 2009).

1. Set up connections/determine encoding capacity for a trial

Each token and type node is connected to the binding pool by selecting a proportion (α = .45) of the total set of binding pool nodes at random. There are no restrictions on the number of types and tokens that can be linked to the same binding pool node. Connections have a weight of 1.0. In tasks with a continuous report dimension, the type connectivity is structured to provide a similarity gradient by incorporating structured overlap between type nodes that represent proximal feature values. For example, type nodes 1 and 2 should have more overlapping connections than type nodes 1 and 3. To do this, we increased the proportion of overlapping connections between any two type nodes by $\vartheta^{\text{distance}}$, with distance as the separation between two type nodes. Therefore, with ϑ = .3, type nodes 1 and 2 will share an extra 30 % of their connections, and type nodes 1 and 3 will share an extra 9 % of their connections, and so on.

For the experiments simulated here, we assume that the limited encoding duration constrains the number of items that can be encoded from the display and that this varies from trial to trial. The number of items that are encoded on a trial is drawn from a uniform distribution over the range specified by parameters ψ and ω, equal to 2 and 7, respectively. Note that this limit reflects limited encoding time rather than the capacity of memory.

2. Encoding

2a. Stimulus selection

At the start of each trial, a set of stimuli is drawn from a uniform distribution [1 360] with replacement. The range represents the possible feature values for the stimuli. This distribution is modified for the similarity and magnet effect simulations as described below.

2b. Convert feature values to type representations

These feature values are converted to type node representations. In a continuous feature dimension, each type node corresponds to a range of 36°, since there are 360° and 10 type nodes (specified as the parameter ρ). The two type nodes

with values closest to the chosen stimulus value are selected, one with a value below and one with a value above the *stimulus* value as shown below:

$$type_{floor\left(\frac{Stimulus}{36}\right)} = 1 - \frac{Stimulus}{36} - \left(floor\left(\frac{Stimulus}{36}\right)\right) \quad (1)$$

$$type_{ceil\left(\frac{Stimulus}{36}\right)} = \frac{Stimulus}{36} - \left(floor\left(\frac{Stimulus}{36}\right)\right) \quad (2)$$

where *floor* and *ceil* correspond to rounding down and rounding up, respectively. *Stimulus* refers to a feature value in the range (1 to 360). If the type node selection is 0, then that type node is set to 10 instead. If the stimulus value falls exactly on the value of a given type node (i.e., 36, 72, etc.), then only one type node is active with a value of 1.0. In the case of feature dimensions with discrete values instead of continuous values, such as widely separated locations used in these simulations, the model assigns one value per type node with activation values of 1.0.

2c. Activate token

Each stimulus is assumed to be represented by a distinct token, and its activity level is set to 1.0.

2d. Increment activation of binding pool neurons

Once the type and token activities have been determined, the binding pool activity is adjusted according to the following formula for each binding pool node B_β:

$$B_\beta = B_\beta + Z_t N_{t,\beta} \sum_{f=1}^{n} X_f L_{f,\beta} \sum_{g=1}^{n} Y_g M_{g,\beta} \quad (3)$$

where B is the set of binding pool nodes in the binding pool, indexed by β. Z_t is the currently active token activation level, X is type layer 1, and Y is type layer 2. L is the connection matrix between the type layer 1 and the binding pool, and M represents the same matrix for the other set of type nodes. N is the connection matrix between the token layer and the binding pool with indices t (the currently active token) and β. n represents the number of type nodes in each type layer. After encoding, the activity of all of the neurons in the binding pool is normalized so that the total sum of binding pool activity is 1.0.

3. Retrieval

3a. Convert cue feature value to type representation

The cue is converted to a type representation in type layer 2 as described in 2b.

3b. Retrieve token

The type representation for layer 2 is projected through the binding pool to reconstruct the token associated with it according to the following equation. This is done once for each token Z_t:

$$Z_t = \sum_{\beta=1}^{n} B_\beta N_{t,\beta} \sum_{f=1}^{n} X_f L_{f,\beta} \sum_{g=1}^{n} Y_g M_{g,\beta} \quad (4)$$

3b(i). Successful token retrieval

A successful token retrieval occurs when the difference between the most active token node and the next most active token node is greater than the token individuation threshold of ς.

3b(ii). Unsuccessful token retrieval

If there is an unsuccessful token retrieval, the model will randomly select a value from the range [0 360], and the length of the retrieved vector (i.e confidence) is fixed at 0.

3c. Retrieve type representation

The Z_t token is then used to retrieve activity in type 1 layer using the following equation:

$$X_f = Z_t \sum_{\beta=1}^{n} B_\beta L_{f,\beta} N_{t,\beta} \sum_{g=1}^{n} Y_g M_{g,\beta} \quad (5)$$

where X represents the retrieved type layer as a population vector given the current pattern of activity across the other set of type nodes (Y) and given the current token t .

3d. Convert retrieved type representation to probed feature value

The reconstructed type node activity across the entire set of type nodes is then converted back to a single value in the range [1 360] by representing each type node as a vector at evenly spaced intervals from this range. To convert the population code into a single output vector, the type activation vectors are

converted into Cartesian coordinates and then summated, using these formulas:

$$C_x = \sum_{f=1}^{n} X_f \cos(f * 36) \tag{6}$$

$$C_y = \sum_{f=1}^{n} X_f \sin(f * 36) \tag{7}$$

with C_x and C_y corresponding to the summated x and y values in Cartesian coordinates, respectively; X_f is the length of activation; and f* 36 is the corresponding degree value of the type nodes in the retrieved feature dimension.

The resulting vector in Cartesian coordinates is converted back into polar coordinates to produce a single output vector consisting of an angle in the range [1 360] and a length, using these formulas:

$$\varepsilon = \sqrt{{C_x}^2 + {C_y}^2} \tag{8}$$

$$\widehat{\theta} = \tan^{-1}\left(\frac{C_y}{C_x}\right) \tag{9}$$

The retrieved angle $\widehat{\theta}$ is interpreted as the model's feature selection, and the length ε is interpreted as the confidence, of retrieval.

4. Change detection

To simulate a change detection task, the model incorporates a dual-threshold process such that a change is declared only when a detected change is sufficiently large and the retrieval process is sufficiently confident about the retrieved value.

4a. Compare retrieved stimuli with probe stimuli

For each stimulus in the probe display a deviation is computed by taking the absolute value of the difference between the retrieved value for that location and the probe stimulus value. The length of the vector of the retrieved stimulus is taken as the confidence. If both the deviation and the confidence are above threshold, that particular stimulus is declared as having changed. The two thresholds are fit against the data from Keshvari et al. (2013) and vary linearly with set size.

4b. Detecting a change trial

If any single stimulus in a trial is determined to have changed, the trial is classified as a change trial, regardless of the classifications of the other stimuli.

4c. Categorize trial type

The model's behavior on a given trial can be interpreted as one of a hit, a miss, a false alarm, or a correct rejection. Note that if the model erroneously detects a change for an unchanged stimulus in a change trial, the trial is recorded as a hit, even if the stimulus that actually changed was misclassified as an unchanged item.

5. Continuous report

To simulate a continuous report task, the model uses the angle of the output vector from retrieval as the reported stimulus value for a given trial. To divide these responses into precision (measured as the standard deviation of a von Mises distribution), swaps (measured as a proportion), and guesses (measured as a proportion), a mixture model is used, as specified by Bays et al. (2009):

$$p\left(\widehat{\theta}\right) = (1 - \gamma - \beta)\phi_\sigma\left(\widehat{\theta} - \theta\right) + \gamma\frac{1}{2\pi} + \beta\left(\frac{1}{m}\right)\sum_{i}^{m}\phi_\sigma\left(\widehat{\theta} - \theta_i^*\right) \tag{10}$$

where $\widehat{\theta}$ is the reported color (in radians), θ is the target color, γ is the proportion of trials on which the subject guesses, ϕ_σ represents the Von Mises distribution with a mean of zero and standard deviation σ, β represents the proportion of retrieval errors or nontarget selections, and m as the number of nontargets. θ_i* is the nontarget color (in radians).

Simulations of Inherent properties

Featural cross talk

To simulate the "magnet" effect of features pulling their representations toward each other, as in Huang and Sekuler (2010), we fixed the set size to two items and simulated three conditions in which the degree separation of the two items was 0, 45, and 90. Ten thousand trials were simulated.

Binding errors

To simulate binding errors from Wheeler and Treisman (2002), the four threshold parameters for change detection were sampled with a grid search using 4,000 simulated trials for each location in the grid for both the color-swap and novel-color conditions. The parameter values used in the grid search spanned a range of ±25 % relative to each of the four threshold parameters specified below (i.e., for the length threshold and deviation threshold). The mean hit rates are computed for set sizes 3 and 6. Critically, for each comparison between the binding color-swap and novel-color condition conditions, the

same parameter values were used for each condition. For this simulation, the color space was not treated as continuous, since Wheeler and Treisman used categorically separate colors. Instead, each type node corresponded to a distinct color.

The effect of confidence on continuous report

Trials are broken into three groups depending on a length threshold. The length threshold was determined by taking tertiles of the combined length values of set sizes 1, 2, 4, and 6, which are the typical within-trial set size manipulation. Thirty thousand trials were simulated.

Parameters

The model has a total of 11 parameters, 10 of which were free to vary in fitting the data from Bays et al. (2009) and Keshvari et al. (2013). The type layer size remained fixed at ρ=10 through all fitting procedures.

Parameter	Description	Value
Binding pool size (Ω)	The amount of available nodes for storing links; as the size increases, so does the representational fidelity of stored information.	800
Type layer size (ρ)	The amount of type nodes in each feature space	10
Connection sparsity (α)	The proportion of active connections between the each layer and the binding pool	0.45
Type connection overlap (ϑ)	The proportion of added overlap between neighboring type nodes	0.30
Encoding capacity [ψ ω]	The number of stimuli encoded per trial is drawn from a uniform distribution with these limits	[2 7]
Token individuation threshold (ς)	If the ratio of the maximum retrieved token activity to the average retrieved token activity does not exceed this threshold, then the model will initiate a "guess" by choosing a feature value at random.	0.016
Length threshold for change detection (κ_s and λ_s)	Two parameters (baseline and set size slope) define the thresholds for detecting change based on the length of the retrieved vector (i.e., confidence).	$\boldsymbol{\Phi_L}$ = 0.005*Setsize + 0.09
Deviation threshold for change detection (κ_d and λ_d)	Two parameters define the thresholds for detecting change based on the deviation of the retrieved value from the probe value.	$\boldsymbol{\Phi_d}$ = 0.038 * Setsize + 0.16

References

Alvarez, G. A., & Cavanagh, P. (2004). The capacity of visual short-term memory is set both by visual information load and by number of objects. *Psychological Science, 15*(2), 106–111.

Awh, E., Barton, B., & Vogel, E. K. (2007). Visual working memory represents a fixed number of items regardless of complexity. *Psychological Science, 18*(7), 622–628.

Baddeley, A. D., & Hitch, G. J. (1974). Working memory. *The Psychology of Learning and Motivation: Advances in Research and Theory, 8,* 47–89.

Bays, P. M., Catalao, R. F. G., & Husain, M. (2009). The precision of visual working memory is set by allocation of a shared resource. *Journal of Vision, 9*(10), 1–11.

Bays, P. M., Gorgoraptis, N., Wee, N., Marshall, L., & Husain, M. (2011a). Temporal dynamics of encoding, storage, and reallocation of visual working memory. *Journal of Vision, 11*(10), 1–15.

Bays, P. M., Wu, E. Y., & Husain, M. (2011b). Dynamic updating of working memory resources for visual objects. *Journal of Neurosciences, 31*(23), 8502–8511.

Block, H. D. (1962). The perceptron: a model for brain functioning. *Reviews of Modern Physics, 34*(1), 123–135.

Bowman, H., & Wyble, B. (2007). The simultaneous type, serial token model of temporal attention and working memory. *Psychological Review, 114*(1), 38–70.

Brady, T. F., & Alvarez, G. A. (2011). Hierarchical encoding in visual working memory: ensemble statistics bias memory for individual items. *Psychological Science, 22*(3), 384–392.

Brady, T. F., & Tenenbuam, J. B. (2013). A probabilistic model of visual working memory: incorporating higher order regularities into working memory capacity estimates. *Psychological Review, 120*(1), 85–109.

Caramazza, A., & Miceli, G. (1990). The structure of graphemic representations. *Cognition, 37,* 243–297.

Fougnie, D., & Alvarez, G. A. (2011). Object features fail independently in visual working memory; evidence for a probabilistic feature-store model. *Journal of Vision, 11*(12), 1–12.

Fougnie, D., Suchow, J. W., & Alvarez, G. A. (2012). Variability in the quality of working memory. *Nature Communication, 3,* 1229.

Fougnie, D., Asplund, C. L., & Marois, R. (2010). What are the units of storage in visual working memory? *Journal of Vision, 10*(12), 1–11.

Franconeri, S. L., Alvarez, G. A., & Cavanagh, P. (2013). Flexible cognitive resources: competitive content maps for attention and memory. *Trends in Cognitive Sciences, 17*(3), 134–141.

Gorgoraptis, N., Catalao, R. F. G., Bays, P. M., & Husain, M. (2011). Dynamic updating of working memory resources for visual objects. *The Journal of Neuroscience, 31*(23), 8502–8511.

Hartshorne, J. K. (2008). Visual working memory capacity and proactive interference. *PLoS one, 3*(7), e2716.

Hasselmo, M. E., Fransen, E., Dickson, C., & Alonso, A. A. (2000). Computational modeling of entorhinal cortex. *Annals of the New York Academy of Sciences, 911*(617), 418–446.

Huang, J., & Sekuler, R. (2010). Distortions in recall from visual memory: two classes of attractors at work. *Journal of Vision, 10,* 1–27.

Jiang, Y., Olson, I. R., & Chun, M. M. (2000). Organization of visual short-term memory. *Journal of Experimental Psychology: Learning, memory, and cognition., 26*(3), 683–702.

Kahana, M. J., & Sekuler, R. (2002). Recognizing spatial patterns: a noisy exemplar approach. *Vision Research, 42*(18), 2177–2192.

Kahneman, D., & Treisman, A. (1984). Changing views of attention and automaticity. In R. Parasuraman & R. Davies (Eds.), *Varieties in Attention* (pp. 29–61). New York: Academic Press.

Kanerva, P. (1993). Sparse distributed memory and related models. *Associate Neural Memories: Theory and Implementation*, 50–73

Kanwisher, N. G. (1987). Repetition blindness: type recognition without token individuation. *Cognition, 27,* 117–143.

Keppel, G., & Underwood, B. J. (1962). Proactive inhibition in short-term retention of single items. *Journal of Verbal Learning and Verbal Behavior, 1*(3), 153–161.

Keshvari, S., van den Berg, R., & Ma, W. J. (2013). No evidence for an item limit in change detection. *PLoS Computational Biology, 9*(2), e1002927.

Lin, P., & Luck, S. J. (2009). The influence of similarity on visual working memory representations. *Visual Cognition, 17*(3), 1–15.

Lin, P., & Luck, S. J. (2012). Proactive interference does not meaningfully distort visual working memory capacity estimates in the canonical detection task. *Frontiers in Psychology, 3*(2), 1–9.

Luck, S. J., & Vogel, E. K. (1997). The capacity of visual working memory for features and conjunctions. *Nature, 390*(6657), 279–281.

Luo, C. R., & Caramazza, A. (1996). Temporal and spatial repetition blindness: effects of presentation mode and repetition lag on the perception of repeated items. *Journal of Experimental Psychology: Human Perception and Performance, 22*(1), 95–113.

Marr, D. (1976). Early processing of visual information. *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences, 275*(942), 483–519.

McClelland, J. L., & Rumelhart, D. E. (1988). *Explorations in parallel distributed processing*. Cambridge: The MIT Press.

Mihalas, S., Dong, Y., von der Heydt, R., & Niebur, E. (2011). Mechanisms of perceptual organization provide auto-zoom and auto-localization for attention to objects. *Proceedings of the National Academy of Sciences of the United States of America, 108*(18), 7583–7588.

Mozer, M. C. (1989). Types and tokens in visual letter perception. *Journal of Experimental Psychology: Human Perception and Performance, 15*(2), 287–303.

Murdock, B. B., Jr. (1983). A distributed memory model for serial-order information. *Psychological Review, 90*(4), 316–338.

Oberauer, K. (2009). Design for a working memory. *Psychology of Learning and Motivation, 51*(9), 45–100.

Oberauer, K., & Eichenberger, S. (2013). Visual working memory declines when more features must be remembered for each object. *Memory & Cognition*, May 2013

Oberauer, K., Lewandowsky, S., Farrell, S., Jarrold, C., & Greaves, M. (2012). Modeling working memory: an interference model of complex span. *Psychonomic Bulletin & Review, 19*(5), 779–819.

Orhan, A. E., & Jacobs, R. A. (2013). A probabilistic clustering theory of the organization of visual short-term memory. *Psychological Review, 120*(2), 297–328.

Plate, T. A. (1995). Holographic reduced representations. *IEEE Transactions on Neural Network, 6*(3), 623–641.

Prinzmetal, W., Amiri, H., Allen, K., & Edwards, T. (1998). Phenomenology of attention: 1. Color, location, orientation, and spatial frequency. *Journal of Experimental Psychology: Human Perception and Performance, 24*(1), 261–282.

Rademaker, R. L., Tredway, C. H., & Tong, F. (2012). Introspective judgments predict the precision and likelihood of successful maintenance of visual working memory. *Journal of Vision, 12*(13), 1–13.

Rock, I., Linnet, C. M., Grant, P., & Mack, A. (1992). Perception without attention: results of a new method. *Cognitive Psychology, 24*, 502–534.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review, 65*(6), 386–408.

Sims, C. R., Jacobs, R. A., & Knill, D. C. (2012). An ideal observer analysis of visual working memory. *Psychological Review, 119*(4), 807–830.

Suchow, J. W., Brady, T. F., Fougnie, D., & Alvarez, G. A. (2013). Modeling visual working memory with the MemToolBox. *Journal of Vision, 13*(10):9, 1–8.

Treisman, A. M., & Gelade, G. (1980). A feature-integration theory of attention. *Cognitive Psychology, 12*(1), 97–136.

Treisman, A. M., & Schmidt, H. (1982). Illusory conjunctions in the perception of objects. *Cognitive Psychology, 14,* 107–141.

van den Berg, R., Shin, H., Chou, W., George, R., & Ma, W. J. (2012). Variability in encoding precision accounts for visual short-term memory limitations. *Proceedings of the National Academy of Sciences of the United States of America, 109*(22), 8780–8785.

van Lamsweerde, A. E., & Beck, M. R. (2012). Attention shifts or volatile representations: what causes binding deficits in visual working memory? *Visual Cognition, 20*(7), 771–792.

VanRullen, R. (2009). Binding hardwired versus on-demand feature conjunctions. *Visual Cognition, 17*(1–2), 103–119.

Vogel, E. K., Woodman, G. F., & Luck, S. J. (2001). Storage of features, conjunctions, and objects in visual working memory. *Journal of Experimental Psychology: Human Perception and Performance, 27*(1), 92–114.

Vogel, E. K., Woodman, G. F., & Luck, S. J. (2006). The time course of consolidation in visual working memory. *Journal of Experimental Psychology: Human Perception and Performance, 32*(6), 1436–1451.

Vul, E., Frank, M. C., Alvarez, G. A., & Tenenbuam, J. B. (2010). Explaining human multiple object tracking as a resource-constrained approximate inference in a dynamic probabilistic model. *Advances in Neural Information Processing Systems, 22,* 1–9.

Vul, E., & Rich, A. N. (2010). Independent sampling of features enables conscious perception of bound objects. *Psychological Science, 21*(8), 1168–1175.

Wei, W., Wang, X. J., & Wang, D. (2012). From Distributed resources to limited slots in multiple-item working memory: a spiking network model with normalization. *Journal of Neuroscience, 32,* 11228–11240.

Wheeler, M. E., & Treisman, A. M. (2002). Binding in short-term visual memory. *Jorunal of Experimental Psychology: General., 131*(1), 48–64.

Wilken, P., & Ma, W. J. (2004). A detection theory account of change detection. *Journal of Vision, 4,* 1120–1135.

Williams, M., Hong, S. W., Kang, M., Carlisle, N. B., & Woodman, G. E. (2013). The benefit of forgetting. *Psychonomics Bulletin & Review., 19*(6).

Wyble, B., Bowman, H., & Nieuwenstein, M. (2009). The attentional blin provides episodic distinctiveness: sparing at a cost. *Journal of Experimental Psychology: Human Perception and Performance, 35*(3), 787–807.

Wyble, B., Potter, M. C., Bowman, H., & Nieuwenstein, M. (2011). Attentional episodes in visual perception. *Journal of Experimental Psychology. General, 140*(3), 488–505.

Zhang, W., & Luck, S. J. (2008). Discrete fixed-resolution representations in visual working memory. *Nature, 452*(7192), 233–235.

Zhang, K. (1996). Representation of spatial orientation by the intrinsic dynamics of the head-direction cell ensemble: a theory. *Journal of Neuroscience, 16,* 2112–2126.