Author's personal copy

Cognitive Psychology 66 (2013) 157–211

Contents lists available at SciVerse ScienceDirect

Cognitive Psychology

journal homepage: www.elsevier.com/locate/cogpsych

Analogous mechanisms of selection and updating in declarative and procedural working memory: Experiments and a computational model

Klaus Oberauer *, Alessandra S. Souza, Michel D. Druey, Miriam Gade

University of Zurich, Department of Psychology – Cognitive Psychology, Binzmühlestrasse 14/22, 8050 Zürich, Switzerland

ARTICLE INFO

Article history:
Accepted 13 November 2012

Keywords:
Working memory
Task switching
Computational modeling
Chunking
Long-term memory
Attention

ABSTRACT

The article investigates the mechanisms of selecting and updating representations in declarative and procedural working memory (WM). Declarative WM holds the objects of thought available, whereas procedural WM holds representations of what to do with these objects. Both systems consist of three embedded components: activated long-term memory, a central capacity-limited component for building structures through temporary bindings, and a single-element focus of attention. Five experiments test the hypothesis of analogous mechanisms in declarative and procedural WM, investigating repetition effects across trials for individual representations (objects and responses) and for sets (memory sets and task sets), as well as set-congruency effects. Evidence for analogous processes was obtained from three phenomena: (1) Costs of task switching and of list switching are reduced with longer preparation interval. (2) The effects of task congruency and of list congruency are undiminished with longer preparation interval. (3) Response repetition interacts with task repetition in procedural WM; here we show an analogous interaction of list repetition with item repetition in declarative WM. All three patterns were reproduced by a connectionist model implementing the assumed selection and updating mechanisms. The model consists of two modules, an item-selection module selecting individual items from a memory set, or responses from a task set, and a set-selection module for selecting memory sets or task sets. The model codes the matrix of binding weights in the item-selection module as a pattern of activation in the set-selection module, thereby providing a mechanism for building chunks in LTM, and for unpacking them as structures into working memory.

© 2012 Elsevier Inc. All rights reserved.

* Corresponding author.
E-mail address: k.oberauer@psychologie.uzh.ch (K. Oberauer).

0010-0285/$ - see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.cogpsych.2012.11.001

Author's personal copy

1. Introduction

The information most relevant for our thoughts and actions often changes in a matter of seconds. To function efficiently, we need a working memory that maintains the most relevant representations at any moment in a highly available state, so that they control our ongoing cognitive activity. Working memory must be selective, excluding irrelevant and potentially distracting representations, and it needs to be updated rapidly.

Our mental activity is guided by *declarative* representations, which represent the objects of thoughts and actions (i.e., physical objects, events, symbols, people, and the relations between them), and by *procedural* representations, which specify what to do with the objects of thought (i.e., condition-action bindings, task sets). So far, the selection, short-term maintenance, and updating of these representations has been studied in different research traditions. Research on short-term and working memory has focused on how people maintain and manipulate the objects of thought. Research on attention and action control has focused on how people prepare for one or more tasks, and how they switch between tasks. Selecting a task and preparing it for immediate execution can be understood as holding a procedural representation of that task in working memory (Mayr & Kliegl, 2000; Meiran & Cohen-Kdoshay, 2012).

Both fields have identified important limitations of our cognitive abilities. Research in the tradition of short-term and working memory has uncovered constraints on how many objects of thought can be considered simultaneously (Halford, Cowan, & Andrews, 2007). Research in the action-control tradition has demonstrated limits on how many tasks can be prepared and executed in parallel (Logan & Gordon, 2001; Pashler, 1994; Vandierendonck, Liefooghe, & Verbruggen, 2010).

Here we propose a theoretical unification of these two research traditions, using *working memory* (WM) as an umbrella concept. Doing so implies extending the concept of WM to describe a system for temporarily making available not only representations of the objects of thoughts, but also representations of the intended (cognitive or overt) actions on these objects. The theoretical framework for this conceptual integration has so far been sketched only verbally (Oberauer, 2009). In this article we present a first attempt to implement the core ideas of this theory as a computational model.

The theoretical framework motivated two hypotheses (Oberauer, 2009). The first is that within WM we can distinguish between *declarative* WM, which provides access to declarative representations to be manipulated, and *procedural* WM, holding the procedural representations that determine what is done with the declarative representations. The second hypothesis is that the two sub-systems have an analogous structure and operate according to analogous principles. The purpose of the experimental work presented here is to test this second hypothesis, which we elaborate in the following section.

The first hypothesis, postulating a distinction between declarative and procedural WM, is motivated by observations suggesting that there are separate capacities for declarative and procedural representations in WM (for a review see Oberauer, 2009). For instance, research with dual-task paradigms has established a fairly robust bottleneck for response selection (Pashler, 1994), such that response selection for one task has to wait until response selection for the other task has been completed. This bottleneck suggests that people can hold only one task set at a time in procedural WM. At the same time, people have little difficulty selecting responses in choice tasks, and even switching between two tasks, while holding a list of items in declarative WM (Liefooghe, Barrouillet, Vandierendonck, & Camos, 2008; Logan, 2004, 2007). This initial evidence notwithstanding, the assumption of separate capacity limits for declarative and procedural WM still awaits thorough empirical testing. Therefore, at this point we are open to the possibility that a unitary WM handles declarative and procedural representations.

The two hypotheses are related by asymmetrical inferential links: Whereas evidence against analogous principles in declarative and procedural WM implies that they must be distinct to some extent, evidence for analogous principles is equally compatible with separate sub-systems and with a single WM system for both declarative and procedural representations. Conversely, if the first hypothesis turns out to be wrong and there is only a single unified WM system, it would imply that the evidence for analogous principles presented in this article needs to be re-interpreted as evidence for the broad

Author's personal copy

scope of the operating principles of a single WM system. This would by no means jeopardize our goal of unifying theories of capacity limits in memory, reasoning, and action.

The structure of the article is as follows: In the following section we explain the assumed analogous structures and processes of declarative and procedural WM in detail. We then present three experiments testing hypotheses derived from this framework with the help of the heuristic of analogies between declarative and procedural WM. Subsequently we introduce the computational model of declarative and procedural WM and present simulations that reproduce the data of the first three experiments. Finally, we use the model to derive new predictions, and test these predictions with two further experiments.

2. The structure of declarative and procedural working memory

In the theoretical framework that motivates our work (Oberauer, 2009), both declarative and procedural WM consist of three embedded components (see Fig. 1). The most encompassing component of both systems is the *activated part of long-term memory* (green structures in Fig. 1). Declarative and procedural representations in long-term memory (LTM) can be temporarily activated by perceptual input or by spreading activation from other representations. Activation primes these representations, implying that they are relatively easy to retrieve into the more central components of the WM system (depicted in blue in Fig. 1). The more central components hold the small subset of activated representations that are actually selected for the ongoing task.

The central component of declarative WM is called the *region of direct access*. The elements held in the direct-access region are bound to positions in a common mental coordinate system, thus forming a structure such as an ordered set or a mental model. For instance, a list of five digits can be held in the region of direct access by binding each digit to a position on a single dimension that represents their ordinal list position. The central component of procedural WM is called the *bridge*. It holds a small set of condition-action bindings that are integrated into a *task set*, that is, a set of mutually exclusive con-

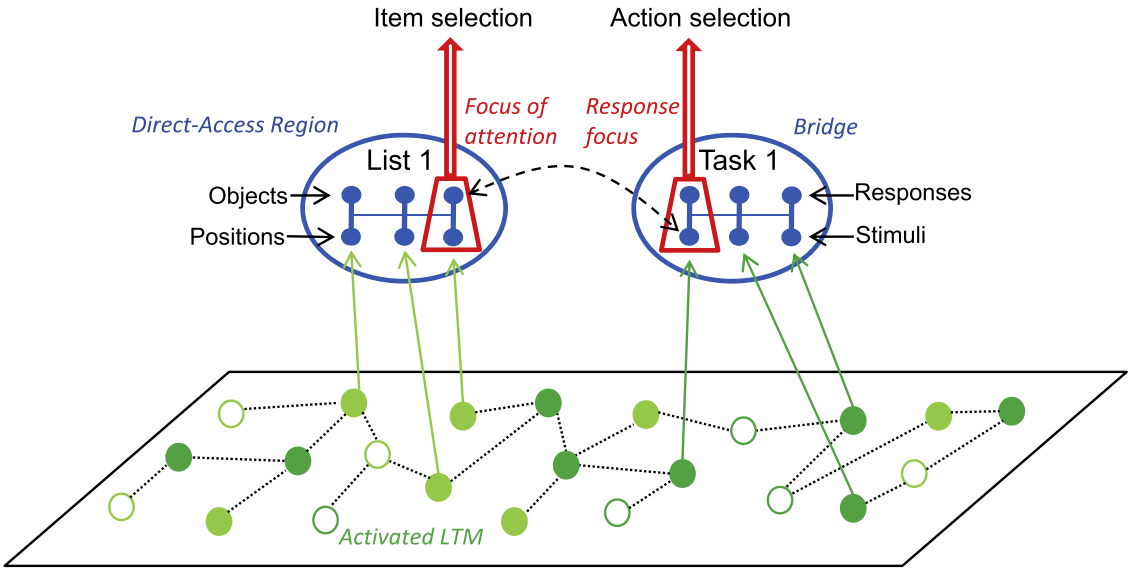

Fig. 1. The architecture of declarative and procedural WM. The network of activated long-term memory is depicted in green (light green: declarative representations; dark green: procedural representations). Representations in the region of direct access or in the bridge, together with their temporary bindings, are shown in blue (note that, although appearing as separate nodes in the figure for graphical reasons, these representations are not separate copies of those in long-term memory; rather they are identical to the representations in activated long-term memory with which they are connected by arrows). The focus of attention and the response focus (in red) select one object and one action, respectively. The object is selected by using the position it is currently bound to as a cue; the action is selected by using the stimulus it is bound to as a cue. When actions respond to representations in working memory (rather than to perceptual stimuli), the object selected in the focus of attention serves as cue for the response-selection focus; this link between the two foci is shown as a black broken-lined arrow.

Author's personal copy

dition specifications, each of which is bound to one of a set of actions, which in turn are bound to (expected) outcomes. For instance, a task set could specify that whenever a digit is even, the person should press the left button, and for every odd digit, they should press the right button; both responses would be further linked to an expected outcome, for instance the disappearance of the digit from the computer screen.

The third component in each WM subsystem serves to single out one elementary representation that is selected for action (red components in Fig. 1). In declarative WM, one element from the set in the direct-access region is retrieved into the *focus of attention* and thereby selected as the object of the next cognitive operation. Analogously, in procedural WM, one action from the set of alternative actions included in the task set currently held in the bridge is brought into the *response focus* and thereby selected as the next action to be carried out.

For instance, imagine a person who holds a list of five digits in memory, and wants to determine whether the third digit is odd or even. Because digits are highly relevant for the task, all nine digits will be activated in LTM. The five digits of the list would be bound to their list positions in the region of direct access. To select the third digit, the focus of attention is directed to the third position, which then serves as cue to the digit currently bound to that position. Thus, the digit itself is brought in the focus of attention. Thereby, that digit is selected as the input for the odd–even decision. At this point declarative WM communicates with procedural WM: The content of the focus of attention is matched against the condition specifications of the task set currently held in the bridge. This task set binds the category of even digits to the action of giving a left response, and the odd-digit category to a right response. The digit in the focus of attention is classified as belonging to one of the parity categories, and that category serves as a cue to the response bound to it in procedural WM. The response bound to this cue is brought into the response focus, and is thereby selected for execution.

Table 1 presents an overview of the structures of declarative and procedural WM, lining up side by side the analogous structural components, their analogous functions and features, and analogous empirical phenomena providing support for these assumed features and functions. Some of the phenomena are new predictions – marked (P) in the table – that are motivated by the analogy and have not yet been tested. The experiments presented in this article provide a test of three of these predictions, each illustrating processes attributed to one of the three components. First, the time costs of switching to a new task set, or to a new memory set, serve to illuminate analogous processes of updating the bridge and the direct-access region, respectively. Second, we will investigate the analogy

Table 1
Analogous functions and features of declarative and procedural WM, and corresponding phenomena.

Declarative WM	Procedural WM
Activated declarative LTM	**Activated procedural LTM**
Temporary strengthening of declarative representations and their associations:	Temporary strengthening of representations of (cognitive) actions and their associations:
Repetition priming	*Priming of stimulus–response associations*
Intrusion of extralist items	*Task-congruency effect*
List-congruency effect (P)*	
No capacity limit	No capacity limit
No set-size effect of memory set currently not selected for access	*No set-size effect of task set currently not selected for action (P)*
Direct-Access Region	**Bridge**
Selection of a memory set	Selection of a task set
Memory-set switch cost (P)*	*Task-set switch cost*
Capacity limit	Capacity limit
Set-size effect for speed of access to an element of the set	*Set-size effect for access to a response of the task set (Hick's law)*
Focus of Attention	**Response Focus**
Selection of a single chunk as the object of an operation	Selection of a single response for execution
Object-repetition benefit	*Response-repetition benefit*
Object-repetition cost after memory-set switch (P)*	*Response-repetition cost after task-set switch*

Note: Structural components are printed in bold; their functions and features are described in normal font; corresponding empirical phenomena are printed in italics. As yet untested predictions are marked by (P), predictions tested in this article are marked by (P*).

Author's personal copy

between task-congruency and list-congruency effects, which might be understood as intrusions from activated LTM. Third, we will investigate the analogy between response-repetition effects and object-repetition effects, which speak to the mechanisms of the attentional foci in procedural and declarative WM. We next characterize the components of the WM system in more detail and derive the predictions to be tested.

2.1. Activated long-term memory

Activated LTM consists of representations and their associations that are temporarily primed, which means that their use is facilitated (Cowan, 1995). Activated representations are relatively easy to retrieve into the more central component of WM (i.e., the direct-access region and the bridge). Activated associations are temporarily strengthened, such that the associated elements activate each other more strongly. Priming of declarative representations is reflected in repetition priming (McKone, 1998) as well as in semantic priming (Jones, Kintsch, & Mewhort, 2006). Priming of procedural representations is manifest in response priming (Lien & Proctor, 2002), in facilitated use of recently used cognitive operations or task sets (Waszak, Hommel, & Allport, 2003; Woltz & Was, 2007), and in priming of stimulus–response mappings (Horner & Henson, 2009; Rothermund, Wentura, & De Houwer, 2005).

Facilitated retrieval also shows in less advantageous forms when activated representations intrude into task-related processes although they are irrelevant and potentially misleading. Intrusions of declarative representations in activated LTM can be found in many tests of immediate memory. For instance, recognition probes matching an element of a recently encoded but currently not relevant list are harder to reject than probes not matching any recently encoded list (Berman, Jonides, & Lewis, 2009; Monsell, 1978; Oberauer, 2001). In procedural WM, recently used stimulus–response associations tend to intrude when the same stimulus later must be given a different response, thereby contributing to negative-priming effects (Rothermund et al., 2005).

Here we investigate the task-congruency effect and its counterpart in declarative WM, the list-congruency effect, as phenomena that could be explained as intrusions from activated LTM. The effect of task congruency in the task-switch paradigm (Rogers & Monsell, 1995) reflects the intrusion of a task set relevant in the experimental context but not for the current trial: Response times are slowed when the task set for the current trial demands a different response to the target stimulus than the alternative task set. This effect has been attributed to the persistent activation of components of the alternative task set in LTM (Meiran & Kessler, 2008; Yamaguchi & Proctor, 2011). In the present experiments we ask whether an analogous list-congruency effect is observed in declarative WM.

2.2. The direct-access region and the bridge

Information retrieved from LTM becomes the content of one of the more central components of the WM system, which means that they are given a privileged state in controlling our thoughts and actions. The central component of declarative WM, the region of direct access, serves as a blackboard for the construction and manipulation of structural representations, such as lists, spatial arrays, or mental models. Contents in the direct-access region are bound to context representations by which they can be accessed directly – for instance, digits in a memory list can be bound to their serial position, or to a location in space, such that these contexts serve as temporary retrieval cues. Focusing attention on the context automatically brings the content bound to it into the focus of attention, thereby selecting it as the object of the next cognitive operation (Bialkova & Oberauer, 2010). In the same way, the central component of procedural WM, the bridge, serves to construct and manipulate structural representations of intended (mental) actions, often called task sets. A task set consists of a set of stimuli or stimulus categories bound to corresponding responses.[1] Each stimulus (category) serves as a

[1] Most contemporary studies on task sets use categorization tasks, such as classifying digits as odd or even. We assume that for those tasks, stimulus categories, rather than individual stimuli, are bound to the corresponding responses. Not all tasks are categorization tasks, however—in fact, the earliest study on task switching did not involve categorization tasks (Jersild, 1927)—and when the stimuli requiring the same response do not form a category, task sets will consist of bindings between individual stimuli and responses.

Author's personal copy

cue to the response bound to it, and focusing attention on a stimulus matching one of the stimulus (categories) automatically brings the response bound to it into the response focus. Thus, the task set in the bridge functions as a "prepared reflex" (Cohen-Kdoshay & Meiran, 2009; Hommel, 2000; Logan, 1978) that enables immediate execution of a response to a stimulus.

The procedural representation of a task set, established in the bridge to operate as a "prepared reflex", must be distinguished from a declarative representation of the task rules, such as a propositional representation of the instructions. This distinction has been made in many theories of practice (Anderson, 1987), and is also incorporated in theories of action control (e.g., ECTVA distinguishes between propositional task representations in "working memory" and what we would call procedural task representations in the TVA component of the system; Logan & Gordon, 2001). Declarative representations of task rules consist of concepts bound to roles in a propositional schema; procedural representations consist of inputs (e.g., stimuli to respond to) bound to responses.

The direct-access region and the bridge have limited capacity. We assume that the capacity limit arises from interference between multiple bindings that must be maintained simultaneously. In the direct-access region, these are bindings between contexts and contents, and in the bridge, these are bindings between inputs and responses. The capacity limit becomes manifest in set-size effects on latencies: The time to select one declarative representation from the content of the direct-access region into the focus of attention increases with the number of elements from which it is selected (Oberauer, 2002). The time to select one response for execution increases with the number of response options in the current task set (Hick, 1952; Schneider & Anderson, 2011).

The WM system can flexibly swap representations between the central components and activated LTM. This enables smooth switching between alternating task sets (Monsell, 2003) and alternating memory sets (Oberauer, 2005). When switching between two task sets in procedural WM, the old set is removed from the bridge and the new set is retrieved into the bridge from activated procedural LTM (Mayr & Kliegl, 2000). Switching between memory sets in declarative WM means that the current set in the direct-access region is replaced by another set retrieved from activated declarative LTM. Research on selection of memory sets has shown that the set size of the currently selected set affects response times whereas the set size of the currently not selected set does not (Oberauer, 2002, 2005; Oberauer & Göthe, 2006).

Switching between task sets incurs a cost in response times and accuracy; this cost is typically reduced when people are given some preparation time between a task cue and the stimulus to be processed according to that task (Monsell, 2003). We predicted an analogous pattern for switching between memory lists. We obtained initial evidence for list-switch costs, and their reduction with a preparation interval, in another set of experiments (Souza, Oberauer, Gade, & Druey, 2012); in the present Experiment 2 we aim to replicate these effects, and in addition ask how the preparation interval modulates the congruency effect and object-repetition effects in declarative WM.

2.3. The focus of attention and the response focus

Memory items in the region of direct access of declarative WM can be accessed for processing through the context that they are bound to; the context serves as a cue to the item. For instance, a digit in a list can be accessed by a representation of its serial position; an element in a spatial array can be accessed by a representation of its spatial location. Selectively accessing an individual item for processing means that it becomes the content of the focus of attention. Likewise, in procedural WM, selecting an individual response for execution means that it becomes the content of the response focus. Like item selection in declarative WM, response selection in procedural WM can be described as cued access: When a perceived stimulus, or an item in the focus of declarative WM, matches one of the stimulus (categories) of the task set in the bridge, then this stimulus (category) cues the response bound to it in the task set. This brings the selected response into the response focus.

One piece of evidence for a single-item focus of attention in declarative WM comes from the *object-switch cost* (or object-repetition benefit). Repeatedly using the same item in a memory set is faster than switching to a different item (Garavan, 1998; Oberauer, 2003; Verhaeghen & Hoyer, 2007). For instance, when people hold a list of digits in WM and carry out a series of arithmetic operations on selected digits, they do so faster when successive operations are applied to the same list item than

Author's personal copy

when they need to switch to another item (Oberauer, 2003). Likewise, when counting the number of triangles and squares presented sequentially, incrementing the same running count as on the previous step (e.g., a triangle following a triangle) is faster than switching to the other count (e.g., a triangle following a square; Garavan, 1998). This object-repetition benefit has been explained by assuming that the last-used object of processing (i.e., the last-processed digit, or the last-updated count) remains in the focus of attention after an operation is completed. Thus, when the same object is needed again for the next operation, it is already selected. In contrast, when a different object is needed, the focus first needs to switch to another object (Garavan, 1998; Oberauer, 2002).[2]

An analogous benefit of repetition has been observed for responses in successive speeded choice trials (e.g., Bertelson, 1965). At first glance, this response-repetition benefit could be interpreted in the same way as the object-repetition benefit: The response focus tends to hold on to the selected response after its execution, such that when the next trial requires the same response, that response is already selected, but when it requires a different response, the response focus must switch.

However, further research has revealed a more complicated picture of response-repetition effects. When the stimuli mapped to the same response are difficult to subsume under a common stimulus category (e.g., "1" and "P" are assigned to a left key press, whereas "2" and "Q" are assigned to a right key press), no response-repetition benefit is observed for transitions between different stimuli assigned to the same response (Campbell & Proctor, 1993; Smith, 1968). Moreover, when people alternate between two tasks, each of which involve the same response alternatives (e.g., classifying digits by their parity or by their magnitude, and using left and right button presses for both classifications), a response-repetition benefit is observed only when the task remains the same across two successive trials. When the task switches, response repetition instead incurs a cost (Kleinsorge, 1999; Rogers & Monsell, 1995). This finding is important for theories of the mechanisms of response selection, reviewed in the Discussion of Experiments section. Based on our hypothesis of analogous mechanisms in declarative and procedural WM, we predict that a corresponding pattern of effects should be observed in declarative WM. The first two experiments in this article test this prediction.

Within our framework, switching between task sets in procedural WM is equivalent to switching between memory sets or lists in declarative WM. So far, experiments showing item-repetition benefits have only compared access to the same versus other items of the *same* list. Here, we investigate repeated access to the same item either after a repetition or after a switch of the list containing that item. We predict an item-repetition benefit when the list is repeated, but an item-repetition cost when the list is switched.

The remainder of this article is organized as follows. We first report two experiments testing the predictions outlined above with the list-switch paradigm, followed by a task-switch experiment closely modeled after our list-switch paradigm, so that we can compare list switching and task switching without confounds by methodological differences. We next introduce a computational model implementing the theoretical framework outlined above, and show how it reproduces the pattern of results in the first three experiments. We also apply the model to benchmark data from the standard task-switch paradigm. Finally, we test three new predictions from the model with two additional experiments. In Table 2 we present an overview of the experiments conducted, indicating their goals, critical manipulations, and main results.

3. Experiment 1

In this and the following experiment, we asked participants to hold in memory two lists, each consisting of three digits. On each trial, participants had to retrieve the item bound to one position of one of the lists as input for a speeded arithmetic operation. Two successive trials could involve a repetition of the list or a switch to the other list. The list-switch cost is assessed as the difference in response times between list repetitions and list switches.

[2] In this article, we use the terms object-repetition effects, item-repetition effects, and digit-repetition effects to refer to the same phenomenon. We speak of objects to refer to individual tokens of a type of items (e.g., an individual digit in a memory list), and we speak of items to refer to the type (e.g., all digits with the same numerical value). We use digit-repetition effects as the more concrete term for item-repetition effects in the context of specific experiments using digits as items.

Author's personal copy

Table 2
Overview of experiments.

Goals	Design	Results
Experiment 1		
Test for analogous effects to task switching in a list switch paradigm	List-switch paradigm (two lists, three items) Item overlap between lists: 1. No overlap 2. Overlap: same list-position 3. Overlap: different list-positions	1. List-switch cost 2. List-congruency benefit 3. Item-repetition benefit in list-repetition trials; item-repetition cost in list-switch trials 4. Reduction of item-repetition by list-repetition interaction for congruent trials
Experiment 2		
Replicate Exp. 1 with different preparation intervals	As Exp. 1, with temporally separated list cue and item cue: CTIs of 0 and 1 s	1. Reduced list-switch cost at long CTI 2. List-congruency benefit hardly affected by CTI 3. Item-repetition by list-repetition interaction unaffected by CTI
Experiment 3		
Investigate whether discrepancies between task-switch and list-switch results are due to ancillary methodological choices	Task-switch paradigm (two tasks, three responses) Response overlap between tasks: 1. Shared response to the same stimulus between tasks 2. Shared response to different stimuli between tasks CTIs of .3 and 1 s	1. Task-switch cost, reduced at long CTI 2. Task-congruency benefit, unaffected by CTI 3. Response-repetition benefit in task-repetition trials; response-repetition cost in task-switch trials 4. Reduction of this interaction by congruency, but not by longer CTI
Experiment 4		
Test the model prediction of a congruency benefit when long-term learning of position-item associations is held constant	List-switch paradigm (two out of four lists) – Across four lists, every item occurs twice at the same position – Two randomly selected lists were used in each run of trials – For the two lists in a run, only one item occurred at the same position in both lists	1. List-switch cost 2. List-congruency benefit for the repeated item of the two lists used in a run
Experiment 5		
Test the model prediction of item-repetition benefit for repeated items in different positions within the same list	List switch paradigm (two lists) Item overlap: 1. No overlap 2. Item-repetition in different positions in the same list 3. Item-repetition in different positions in different lists	1. List-switch cost 2. Item-repetition benefit for repeated items in the same list 3. Item-repetition cost for repeated items in different positions in different lists

Author's personal copy

To assess the effect of congruency, we included a list composition in which the same digit occurred in both lists in the same position, for example [2 5 3] and [7 6 3]. Here the third list position is linked to the same digit in both lists. Thus, the digit to be selected in response to a cue to the third position is the same regardless of lists. This is the situation analogous to a congruent stimulus–response mapping in two tasks, where a stimulus is linked to the same response in both tasks (see Fig. 2A). When the two lists had the same digit in the same position, we classified all trials requiring access to the digit in that position as *congruent* trials. Trials using any other position were regarded as *incongruent*, because those positions were bound to different digits in the two lists. When the two lists shared no digit in the same list position, all trials were incongruent.

To assess item-repetition effects, we used a list composition in which one digit was an element of both lists, in different list positions. For instance, the two lists could be [2 5 3] and [9 6 5]. This enabled us to investigate digit repetitions and digit switches in conjunction with list repetition, and in conjunction with list switches (see Fig. 2B), avoiding confounds with congruency.

In this and the following experiments, memory lists remained constant throughout a block of trials. This is untypical for experiments in research on (declarative) working memory, but matches the usual procedure in task-switch experiments, where the task sets are defined by instruction at the beginning and not changed during the experiment. We believe that these methodological differences merely reflect differences in research tradition. Theories of working memory agree that the contents of WM can be new stimuli as well as information from LTM. Therefore, there is no theoretical reason to investigate WM only with memory sets given through new stimuli on every trial. The few studies directly comparing fixed sets (retrieved from LTM) and varied sets (presented at the start of each trial) found

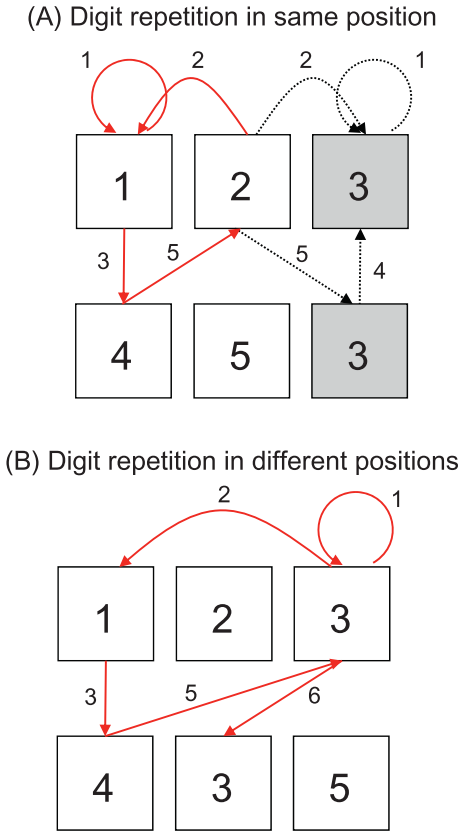

Fig. 2. Conditions in the design of Experiments 1 and 2. Panel A shows the list composition with a repeated digit in the same position; panel B shows the composition with a repeated digit in different positions. Transition conditions are: (1) List repetition, position repetition, digit repetition. (2) List repetition, position switch, digit switch. (3) List switch, position repetition, digit switch. (4) List switch, position repetition, digit repetition. (5) List switch, position switch, digit switch. (6) List switch, position switch, digit repetition. Positions of congruent items are shaded. Transitions to incongruent items are marked by red continuous lines; transitions to congruent items are marked by black broken lines.

Author's personal copy

comparable effects, such as the increase of retrieval times with set size (Risse & Oberauer, 2010; Sternberg, 1969), and the object-repetition benefit, which is a central topic of the present experiments (Risse & Oberauer, 2010). Therefore, we used fixed memory sets to render our results more directly comparable with task-switch experiments.

In sum, Experiment 1 tests three hypotheses derived by analogy to well-established findings in task-switching studies: (1) There is a cost for switching between two memory lists. (2) Item retrieval is faster for congruent list positions, which are occupied by the same item in both lists, than for incongruent positions occupied by different items. (3) Repeatedly accessing the same digit in the same list as on the preceding trial is faster than switching to another digit in the same list (in line with the often reported object-switch cost), but when the list is switched, item repetition incurs a cost.

3.1. Method

3.1.1. Participants

Twenty-five students from the University of Zurich took part in a one-hour session in exchange for 15 CHF or course credit. One participant was excluded because of experimenter error, and another because of exceptionally low accuracy (.41, compared to an average of .96, SD = .03, for the remainder of the sample), leaving $N = 23$ for analysis.

3.1.2. Materials and procedure

The experiment consisted of three blocks, which participants completed in counterbalanced order. For each block, two lists of three digits each were created at random for each participant. For a given participant the lists remained the same within the block. The three blocks differed by their list composition. In the *no-overlap* block, which served merely as a control condition replicating conditions of previous experiments, the two lists shared no digits. In the *different-position* block, the two lists shared one digit in different positions, as in [4 2 6], [1 9 4]. In the *same-position* block, the two lists had one digit in the same list position in common, for example [4 2 6] [4 1 9].

Within each block, participants completed two practice runs, followed by 16 test runs. Fig. 3 illustrates the flow of events in a run. Each run started with the display of three rectangular frames in a row on the screen. The digits of the first list were displayed sequentially in red, one in each frame, proceeding from left to right at a pace of 1 s per digit. They were immediately followed by the three digits of the second list, displayed in blue, occupying the same three frames from left to right at the same pace. After presentation of the lists, participants worked in self-paced mode through a series of 13 arithmetic operations. Operations ranged from −3 to +3 (excluding 0), with the constraint that the result was between 1 and 9. Each arithmetic operation was initiated by a sign-number combination such as "+2" displayed in red or blue in one of the frames. The color indicated which list to use, and the frame indicated which digit to use from that list. Participants applied the displayed operation to the digit and typed the result on the number keyboard. They were not asked to update the memory list by the result. Participants were instructed to work as quickly and accurately as possible. Immediately after each response, the next operation was displayed.

The color and location of each operation was determined at random, with an equal probability of repeating or switching the list, and an equal probability of selecting each of the three frames. Thus, half the transitions between trials involved list switches, and half involved list repetitions. One third of all trials used the same frame as before; in case of a list repetition, this implies repeated access to the same digit. Except for the no-overlap block, repeated access to the same digit could also occur in conjunction with a list switch. Digit repetitions in combination with list switches occurred on 1 out of 18 trials, because this happened only when a trial used the repeated digit (with probability 1/3), and the following trial used the same digit in the other list (with probability 1/6).

3.2. Results

Statistical analyses focus on response times (RTs). With one exception noted below, analogous analyses with error rates (ERs) yielded no significant effects. RTs from errors and from responses following an error were removed. Outliers in the remaining data were determined by first eliminating

Author's personal copy

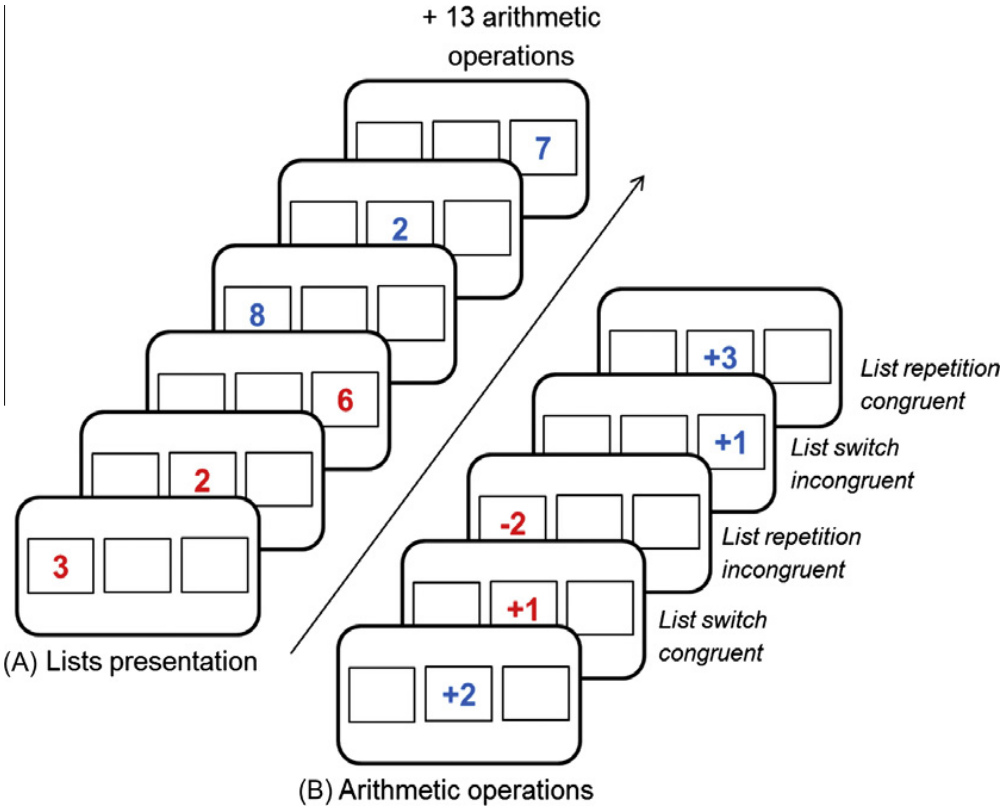

Fig. 3. Example of the flow of events in a run of trials in the same position block. Panel A shows the encoding of the lists part, and panel B shows some of the subsequent trials. The labels indicate whether a trial required access to the same or a different list in relation to trial $n-1$ (list repetition vs. switch, respectively), and whether the position cued required access to a congruent or incongruent item.

RTs > 15 s, and then eliminating RTs exceeding the within-subject mean, computed for each condition of list repetition and position repetition, by more than three within-subject standard deviations (1.8% of RTs were removed as outliers). The remaining RTs were averaged within each cell of the design formed by three independent variables, list repetition (vs. switch), position repetition (vs. switch), and digit repetition (vs. access to a different digit). This was done separately for each block. Depending on the list composition of the block, some design cells were necessarily empty (e.g., there could not be a digit repetition accompanied by a list switch, or by a position switch, in the no-overlap block). The mean RTs are presented in Table 3.

3.2.1. List switch cost

We first assessed the cost of list switching in an ANOVA with List Composition (no overlap, overlap in different position, and overlap in same position) and List Repetition as independent variables. For this analysis, all trials repeating the same position as on the preceding trial were excluded, analogous to the common practice in task-switching studies to exclude stimulus repetitions (e.g., Hübner, Kluwe, Luna-Rodriguez, & Peters, 2004). We also excluded trials requiring retrieval of the same digit as the preceding trial because, as we will show below, digit repetitions had a large effect on RTs that would confound the assessment of list-repetition effects. As predicted, there was a substantial cost of switching the list, $F(1, 22) = 74.0$, partial $\eta^2 = .77$, $p < .001$, which did not interact with list composition, $F < 2.3$. RTs on list repetition trials were on average 2.16 s (SE = .07), whereas list switch trials took on average 2.39 s (SE = .08). The main effect of list composition was significant, $F(1, 22) = 4.7$, partial $\eta^2 = .18$, $p = .01$, mainly because responses were faster in the block with a repeated digit in the same list position. This is due to the fact that this list composition is the only one that included congruent trials.

Author's personal copy

Table 3
Mean response times in seconds (standard deviations in parentheses) by list repetition, position repetition, and digit repetition conditions; Experiment 1.

List composition *(Congruency)*	Digit repetition condition	List Repetition		List Switch	
		Position Repetition	Position Switch	Position Repetition	Position Switch
No overlap	Digit repetition	1.73 (0.38)			
(all incongruent)	Digit switch		*2.17 (0.36)*	2.37 (0.39)	*2.42 (0.37)*
Different position	Digit repetition	**1.64 (0.40)**			**2.80 (0.70)**
(all incongruent)	Digit switch		***2.20 (0.38)***	2.33 (0.38)	***2.49 (0.44)***
Same position	Digit repetition	**1.69 (0.48)**			
(incongruent)	Digit switch		***2.26 (0.40)***	2.26 (0.50)	***2.46 (0.49)***
Same position	Digit repetition	**1.51 (0.56)**		**1.74 (0.75)**	
(congruent)	Digit switch		***1.74 (0.56)***		***1.82 (0.60)***

Note: Data used to assess digit-repetition effects and congruency effects (Fig. 4) are printed in bold. Data used to assess the list repetition effect are printed in italics.

3.2.2. Congruency effect

For an assessment of the effect of congruency we used RTs from the block in which a digit was repeated in the same position of both lists, as the example in panel A of Fig. 2. In this example, all trials probing the third list position were congruent trials, because the third position is bound to the same digit (i.e., 3) in both lists. All other trials are incongruent, because the other list positions are bound to different digits in each list.

To assess the effects of congruency, we ran an ANOVA comparing congruent and incongruent trials in three types of transitions: (1) list repetition and digit repetition; (2) list repetition with a digit switch; and (3) list switch with a digit switch. The relevant data points are depicted in the top panel of Fig. 4. There was a general benefit of congruency on RTs, $F(1, 22) = 36.9$, partial $\eta^2 = .74$, $p < .001$, which was modulated by an interaction with transition type, $F(2, 44) = 26.6$, partial $\eta^2 = .55$, $p < .001$. The interaction reflected the fact that the congruency benefit was relatively small (though still significant, $t = 2.3$, $df = 22$, $p = .02$) when list and digit were repeated, and larger in the two conditions involving a digit switch.

The main effect of congruency was also found in an analogous ANOVA with error rates, $F(1, 22) = 11.4$, partial $\eta^2 = .34$, $p = .003$. ER was 4.9% (SE = 0.6) for incongruent, and 2.4% (SE = 0.7) for congruent trials.

3.2.3. Item repetition effects

Our third analysis focuses on the comparison of digit repetition and digit switch, in conjunction with repeating or switching the list. A comparison of these four conditions unaffected by congruency is afforded by the list composition in which the repeated digit appeared in different positions in the two lists, such as in panel B of Fig. 2. With this composition, all trials are incongruent. A digit repetition with list repetition implied a position repetition, whereas a digit repetition together with a list switch implied a position switch.

The bottom panel of Fig. 4 plots the relevant four data points. An ANOVA revealed a weak and marginal main effect of digit repetition, $F(1, 22) = 4.3$, $p = .050$, partial $\eta^2 = .16$, together with a main effect of list repetition, $F(1, 23) = 103.6$, $p < .001$, partial $\eta^2 = .83$. As predicted, the digit-repetition effect interacted with list repetition, $F(1, 23) = 34.1$, $p < .001$, partial $\eta^2 = .61$. When the list repeated, digit repetitions were faster than digit switches, $t(22) = 7.75$, $p < .001$. When the list switched, digit repetitions were slower than digit switches, $t(22) = 3.0$, $p = .006$.

A further analysis investigated the effect of digit repetition in list-repetition and list-switch trials for the lists repeating a digit in the *same* position (panel A in Fig. 2), focusing on congruent trials only (incongruent trials had to be excluded because the combination of list switch and digit repetition was necessarily congruent in this list composition). In contrast to the preceding analysis of incongruent trials, among the congruent trials digit repetitions were beneficial in conjunction with both list repetitions and list switches (unfilled data points in the top panel of Fig. 4). The ANOVA showed significant

Author's personal copy

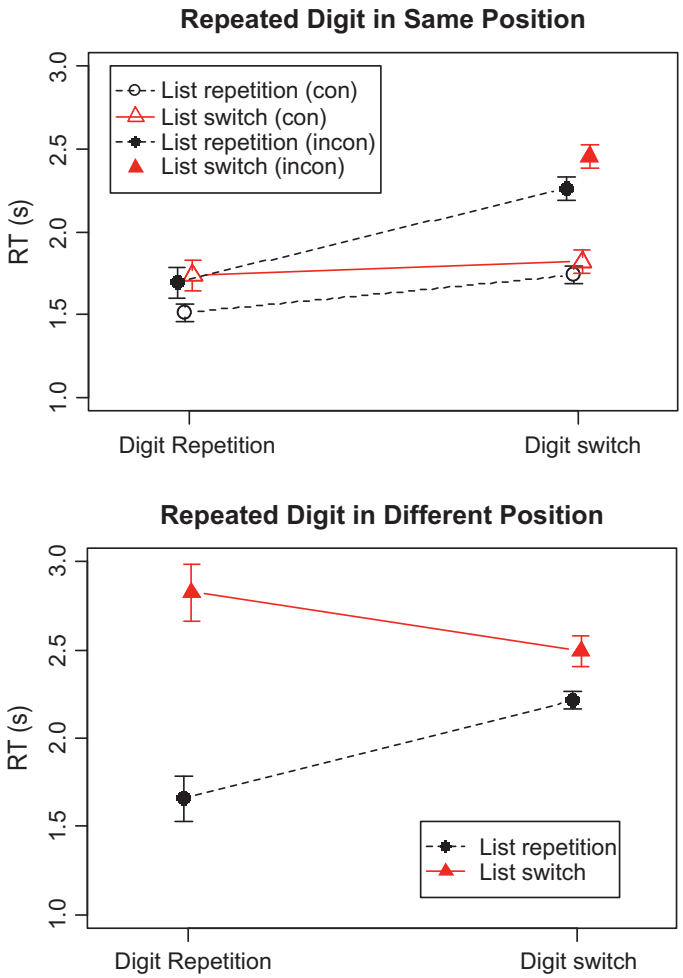

Fig. 4. Mean response times from Experiment 1. Top: Effects of congruency, item repetition, and list repetition in lists with an identical digit in the same list position. Bottom: Item-repetition effects for list repetition and list-switch trials (all incongruent) in lists with an identical digit in different positions of the two lists. Error bars are 95% confidence intervals for within-subject comparisons.

beneficial effects of list repetition, $F(1, 22) = 13.9$, partial $\eta^2 = .38$, $p = .001$, and of digit repetition, $F(1, 22) = 12.0$, partial $\eta^2 = .35$, $p = .002$. The interaction missed the significance level, $F(1, 22) = 3.1$, partial $\eta^2 = .12$, $p = .10$.

3.3. Discussion

Experiment 1 showed three effects in a list-switching paradigm that are analogous to well-established effects in task-switching paradigms. First, switching to a new list takes time. This list-switch cost replicates a previous finding from our lab (Souza et al., 2012), and also meshes well with the observation of a time cost for switching between lists of tasks in a study on sequential behavior (Schneider & Logan, 2007). Second, both error rates and RTs were reduced on congruent trials. This finding is analogous to the congruency effect in task-switching studies (Rogers & Monsell, 1995). In the task-switching paradigm, responses are faster to stimuli that are linked to the same response in both tasks. Here we show that access to information in declarative WM is faster for positions linked to the same items in both lists. Third, we confirmed the predicted interaction between digit repetition and list repetition: In list repetition trials, repeated access to the same digit was beneficial, but in list switch trials, we obtained a digit-repetition cost. This finding is analogous to the observation of response-repetition benefits in sequential choice tasks, which turn into response-repetition costs after a task switch (Kleinsorge, 1999; Rogers & Monsell, 1995).

Author's personal copy

A further finding, not anticipated, was that congruency eliminated the interaction of list repetition and item repetition: With congruent trials, there were only beneficial main effects of repeating the list and of repeating the digit. We searched for a corresponding effect in the task-switching literature. Yehene and Meiran (2007) report two experiments manipulating the critical variables.[3] They found the usual interaction of response repetition with task repetition, and this pattern was not modulated by congruency. Thus, the absence of the list-repetition by item-repetition interaction in congruent trials observed here might be an instance where the analogy to findings from the task-switching paradigm does not hold. We return to this issue in Experiment 3, and present an explanation for the apparent discrepancy using our computational model introduced below.

4. Experiment 2

The second experiment replicates the first, with an additional manipulation of preparation time. We varied the interval between presentation of the list cue and the presentation of the arithmetic operation in one of the frames. With a long preparation interval, participants can use the list cue to retrieve the correct list from activated LTM into the direct-access region. To the extent that people use the preparation interval for this purpose, the list-switch cost should be reduced at the longer interval. A corresponding reduction of task-switch costs with longer preparation intervals has often, though not always, been observed in task-switch studies (Kiesel et al., 2010; Vandierendonck et al., 2010). In a previous study, we found that the list-switch cost was reduced, but not eliminated, with an increased preparation interval (Souza et al., 2012). Our first aim of Experiment 2 was to replicate this effect.

In addition, we wanted to investigate how a longer preparation interval affects the list congruency effect and the pattern of item repetition effects observed in Experiment 1. Again we draw on corresponding manipulations in the task-switch literature to derive hypotheses by analogy. The most robust data come again from the large study by Yehene and Meiran (2007). They found that response repetition benefits (after task repetition) and response repetition costs (after task switching) were substantially reduced at the longer preparation interval. In contrast, the task-congruency effect was undiminished by longer preparation time. The latter result was replicated by Yamaguchi and Proctor (2011). Our second aim for Experiment 2 therefore was to test whether the interaction of item repetition and list repetition is reduced with a longer preparation interval. Our third aim was to test the prediction that the list-congruency effect is not affected by the preparation interval.

4.1. Method

Participants were 18 students from the University of Zurich who took part in exchange for 15 CHF per session or course credit. The second experiment was identical to the first, with the following exceptions: The list to be used for a given trial was cued not only by the color of the arithmetic operation but also by the color of the frames. This enabled us to temporally separate onset of the cue (i.e., the frames turning blue or red) and onset of the operation. Participants were tested in two sessions, one with a cue-target interval (CTI) of zero (as in Experiment 1), and the other with a CTI of 1 s. The order of CTI conditions was counterbalanced across participants.

4.2. Results

Accuracy was high for all participants ($M = 0.97$, $SD = .02$). With one exception, analyses of ERs revealed no significant effects. Response times were treated as in Experiment 1 (2.1% of RTs were excluded as outliers). The mean RTs for the design cells obtained by crossing list repetition, position repetition, and digit repetition are presented in Table 4.

[3] Here and in the remainder of this article we regard the result of Yehene and Meiran (2007) as benchmark data from the standard task-switch paradigm, because this study involved an unusually large sample size ($N = 98$), jointly manipulated the most frequently investigated independent variables, and reported converging results from two different versions of the task-switch paradigm.

Author's personal copy

Table 4
Mean response times (standard deviations in parentheses) by list repetition, position repetition, and digit repetition conditions separately for each CTI level in Experiment 2.

List composition *(Congruency)*	Digit repetition condition	List Repetition		List Switch	
		Position Repetition	Position Switch	Position Repetition	Position Switch
CTI = 0 s					
No overlap	Digit repetition	1.68 (0.49)			
(all incongruent)	Digit switch		*2.04 (0.43)*	2.14 (0.43)	*2.19 (0.48)*
Different position	Digit repetition	**1.66 (0.51)**			**2.44 (0.71)**
(all incongruent)	Digit switch		*2.00 (0.36)*	2.07 (0.34)	***2.24 (0.47)***
Same position	Digit repetition	**1.63 (0.48)**			
(incongruent)	Digit switch		*2.06 (0.42)*	2.03 (0.29)	***2.32 (0.57)***
Same position	Digit repetition	**1.54 (0.57)**		**1.67 (0.57)**	
(congruent)	Digit switch		***1.66 (0.51)***		***1.72 (0.53)***
CTI = 1.0 s					
No overlap	Digit repetition	1.51 (0.40)			
(all incongruent)	Digit switch		*1.59 (0.45)*	1.55 (0.39)	***1.58 (0.45)***
Different position	Digit repetition	**1.42 (0.32)**			**1.85 (0.56)**
(all incongruent)	Digit switch		***1.60 (0.44)***	1.54 (0.34)	***1.63 (0.39)***
Same position	Digit repetition	**1.42 (0.27)**			
(incongruent)	Digit switch		***1.62 (0.43)***	1.56 (0.41)	***1.68 (0.56)***
Same position	Digit repetition	**1.34 (0.45)**		**1.40 (0.32)**	
(congruent)	Digit switch		***1.38 (0.50)***		***1.33(0.38)***

Note: Data used to assess congruency effects (Fig. 5) and item-repetition effects (Fig. 6) are printed in bold. Data used to assess the list repetition effect are printed in italics.

4.2.1. List switch cost and preparation interval

As with Experiment 1, we analyzed the effect of list repetition vs. switch across all three list compositions, excluding trials with position repetition or digit repetition. The ANOVA with List Composition, List Repetition, and CTI as independent variables showed a substantial cost of switching to a new list, $F(1, 17) = 23.0$, partial $\eta^2 = .58$, $p < .001$, which was reduced with the longer CTI, $F(1, 17) = 31.9$, partial $\eta^2 = .65$, $p < .001$, for the interaction. There was also a main effect of CTI, $F(1, 17) = 37.3$, partial $\eta^2 = .69$, $p < .001$. List composition had no main effect and entered into no interaction (all $F < 2$). In a follow-up analysis for the long CTI only, the list-switch cost was no longer significant, $F(1, 17) = 0.44$, partial $\eta^2 = .03$, $p = .52$.

An analogous ANOVA for ERs revealed only a significant main effect of list composition. Error rates were 3.0% (SE = 0.4) for lists without repeated digit, 3.6% (SE = 0.6) for lists with digits repeated in different positions, and 2.2% (SE = 0.4) for lists with digit repetitions in the same position.

4.2.2. Congruency effect and preparation interval

For the analysis of the congruency effect we turn to the lists with a shared digit in the same position (cf. Fig. 2A). Congruent trials were those that probed the position with the shared digit; incongruent trials were trials probing a position occupied by different digits in each list. The relevant data are plotted in Fig. 5. We again focused on the three transition types in Fig. 5 that enabled a comparison of congruent and incongruent trials (i.e., list repetition with digit repetition, list repetition with digit switch, and list switch with digit switch). We included these three transition types in an ANOVA with Congruency and CTI as further independent variables. There was a main effect of congruency, $F(1, 17) = 9.7$, partial $\eta^2 = .37$, $p = .006$, which was only slightly diminished at the longer CTI, as reflected in the marginal interaction, $F(1, 17) = 4.7$, partial $\eta^2 = .22$, $p = .05$. The congruency advantage was 360 ms at the short CTI, and 225 ms at the long CTI; the latter was still significant in a follow-up analysis, $F(1, 17) = 5.0$, partial $\eta^2 = .23$, $p = .04$.

The congruency effect also interacted with the kind of transition, $F(2, 34) = 13.56$, partial $\eta^2 = .44$, $p < .001$. Separate analyses for the three transition types showed that the congruency effect was relatively small (83 ms) and non-significant when lists and digits repeated ($F < 1.1$). The congruency

Author's personal copy

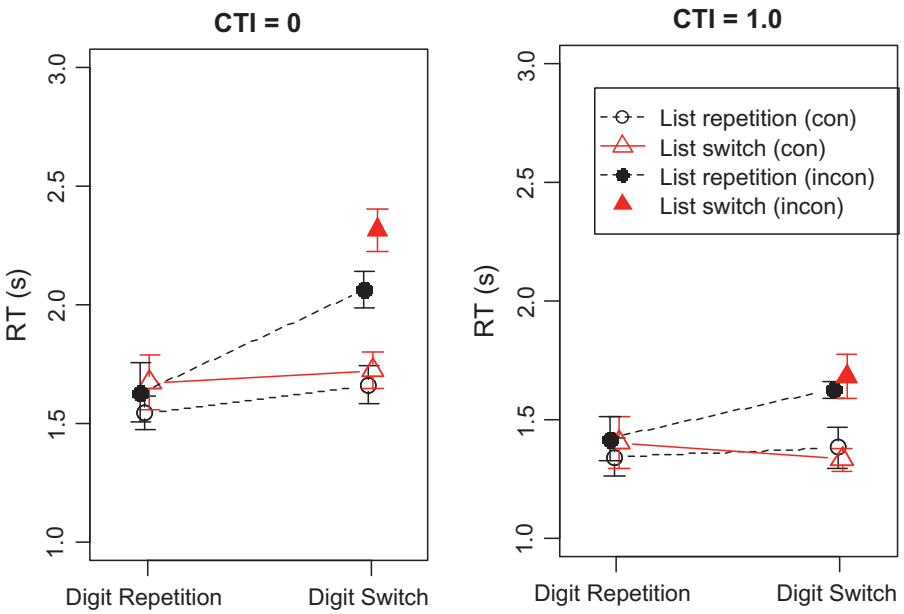

Fig. 5. Mean response times from Experiment 2: Effects of congruency, item-repetition, and list repetition (lists with an identical digit in same list position), for two levels of cue-target interval (CTI). Error bars are 95% confidence intervals for within-subject comparisons.

effect was larger in trials involving a digit switch, combined with a list repetition (congruency effect = 323 ms, $F(1, 17) = 8.2$, partial $\eta^2 = .33$, $p = .01$), and with a list switch (472 ms, $F(1, 17) = 17.4$, partial $\eta^2 = .51$, $p < .001$). Both these effects were reduced with the longer CTI, as reflected in the interaction of congruency with CTI, $F(1, 17) = 5.8$, partial $\eta^2 = .26$, $p = .03$ for digit switch with list repetition, and $F(1, 17) = 5.8$, partial $\eta^2 = .25$, $p = .03$ for digit switch with list switch. To summarize, congruency had a large beneficial effect on trials that involved a switch to a different digit (and a different position). This congruency benefit was diminished but far from eliminated at the long CTI.

4.2.3. Item repetition effects and preparation interval

Fig. 6 shows the critical interaction of list repetition with digit repetition for the list composition with a repeated digit in different positions (as in Fig. 2B), in which all trials were incongruent. The main effect of digit repetition was not significant ($F < 1$), but digit repetition interacted with list repetition, $F(1, 17) = 35.2$, partial $\eta^2 = .67$, $p < .001$. As predicted, there was a digit-repetition benefit in list repetition trials, which turned into a digit-repetition cost in list switch trials. This pattern appeared unaffected by CTI, as indicated by the non-significant interaction of list repetition, digit repetition, and CTI ($F = 1.1$). This analysis also yielded significant effects of list repetition, CTI, and their interaction, reflecting the decline of the list-switch cost with longer CTI reported in the preceding section.

An ANOVA on congruent trials (list composition as in Fig. 2A) with List Repetition, Digit Repetition, and CTI as variables showed that the effects of list repetition and of digit repetition (and their interaction) largely disappeared (see the unfilled data points in Fig. 5). The only significant effect was that RTs were faster with the longer CTI, $F(1, 17) = 19.9$, partial $\eta^2 = .54$, $p < .001$.

4.3. Discussion

Experiment 2 replicated and extended the main result of the first experiment. First, we again observed a list-switch cost. With a preparation interval of one second, this cost disappeared. This finding contrasts with the observation of a residual list-switch cost after even longer preparation intervals in our previous study (Souza et al., 2012) and in the related study by Schneider and Logan (2007). There are a few methodological differences between the present study and those precedents, but none of them could easily explain the difference with regard to residual list-switch costs.

Author's personal copy

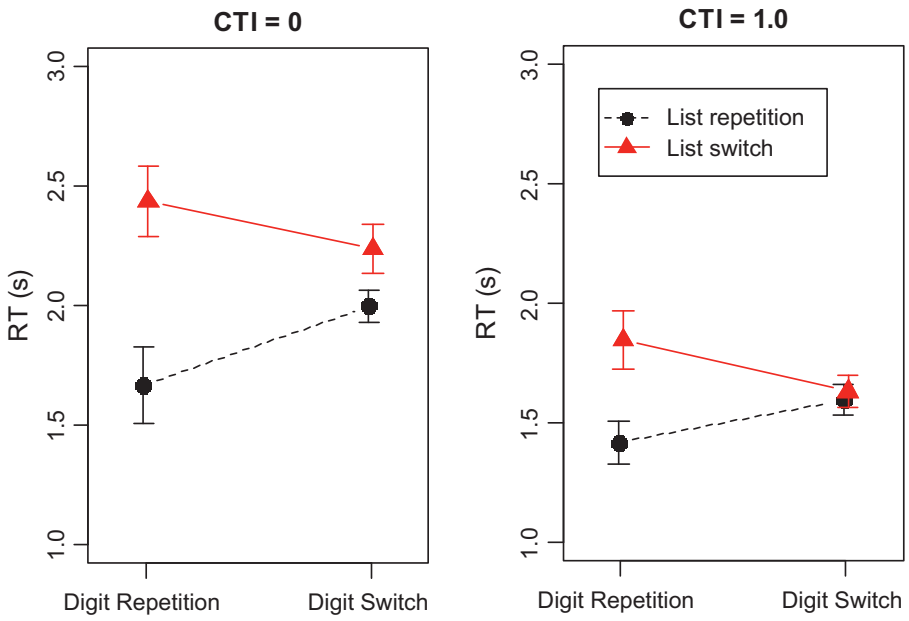

Fig. 6. Mean response times from Experiment 2: Item-repetition effects for list-repetition and list-switch trials (lists with an identical digit in different list positions), for two levels of cue-target interval (CTI). Error bars are 95% confidence intervals for within-subject comparisons.

Second, we replicated the list congruency effect, and in addition showed that it hardly declined with increasing preparation interval. This finding is analogous to the corresponding result in previous task-switching studies: With increasing preparation interval the effect of task congruency was only slightly reduced (Yamaguchi & Proctor, 2011) or not reduced at all (Yehene & Meiran, 2007). In the context of task switching, Yamaguchi and Proctor (2011) have argued from this result that the task-congruency effect arises from primed stimulus–response associations in LTM, which are unaffected by preparation time. They reasoned that a preparation interval could be used to remove the currently irrelevant task set from WM, such that any congruency effect arising from residual representations of the currently not used task in WM should be eliminated or strongly reduced with a longer CTI. Yamaguchi and Proctor (2011) therefore concluded that the task-congruency effect reflects primarily stimulus–response associations in LTM. Our modeling work presented below, however, will show that this conclusion would be premature.

Third, we also replicated the pattern of item-repetition effects found in Experiment 1: The digit-repetition benefit observed after list repetition turned into a digit-repetition cost after a list switch. This interaction was undiminished by increasing preparation time. The latter observation might constitute another violation of the proposed analogy between effects in list-switching and in task-switching paradigms: Yehene and Meiran (2007) found that the interaction of task repetition and response repetition declined substantially at the longer preparation interval. Before we turn to the theoretical implications of our findings, we present a task-switching experiment that closely mirrors the design of the list-switching experiments above. This experiment will enable a comparison of effects in analogous experiments probing declarative and procedural WM with comparable design features.

5. Experiment 3

The results so far largely support the proposed analogy between effects in declarative and in procedural WM. However, we also noted two exceptions to this analogy: In contrast to published task-switch studies, we found that the interaction of list repetition and item repetition was much diminished by congruency, and we found no reduction of that interaction with preparation time. These exceptions could hint at a limit of the analogy between declarative and procedural WM. Alternatively, they could arise from a number of differences between our list-switch paradigm and the standard task-switch paradigm. For instance, most task-switch experiments, including those of Yehene and

Author's personal copy

Meiran (2007), use two stimulus categories mapped onto two responses in each task; in contrast, we used three positions linked to three digits in each list. Also, in many task-switch experiments, including Yehene and Meiran, both tasks use the same response keys; in contrast, in the present experiment, the set of "mental responses" (i.e., the digits selected for processing) overlapped only partially between the two lists. The overlap of response sets between two tasks has a substantial influence on several effects in task-switch studies (Mayr, 2001). These or other incidental differences between the paradigms could be responsible for the deviations from analogy noted above. To test this possibility, we created a task-switch paradigm that parallels as closely as possible the list-switch paradigm of Experiments 1 and 2.

Participants alternated between two three-choice tasks. Three symbols served as stimuli for both tasks; they correspond to the three positions in the list-switch paradigm. Each symbol was mapped to a different response button; the response alternatives correspond to the digits in the list-switch paradigm. Thus, each task set consisted of three stimulus–response mappings. A red and a blue cross served as task cues. The red and blue tasks had one response in common. In one task composition (response repetition to same stimulus, see top left panel of Fig. 7), this common response was mapped to the same stimulus for both tasks – for example, in both tasks the '§' requires the same response (press button 1) – thus creating a congruent stimulus–response mapping. In the other task composition (response repetition to different stimuli, see top right panel of Fig. 7), the common response was mapped to different stimuli – e.g., in the red task, button 4 had to be pressed in response to #, and in the blue task, button 4 was to be pressed in response to §. In this composition all stimuli were incongruent (i.e., they required different responses in the two tasks), so that response-repetition effects could be studied without confounds with congruency.

5.1. Method

5.1.1. Participants

Twenty-four students from the University of Zurich took part in return for either partial course credit or a payment of 20 CHF.

5.1.2. Apparatus and stimuli

The experiment was administered on an IBM-compatible personal computer with a 15-inch flat touch-screen monitor (Elo ET 1915 L) using a 1280 by 1024 resolution. The screen was located centrally on a desk in front of the participants with the surface subtending an angle of about 56°. A small pillow was placed in front of the screen on which participants could rest their wrist.

The screen was divided into two sections by a horizontal line separating the bottom third from the top two thirds. The top section was used for stimulus presentation, while the bottom section was used to present the response buttons. The response buttons comprised five gray circles, numbered 1–5 from left to right. These circles were presented equally spaced along a semicircular arc. In the center of this arc, a black circle was shown (the home button), and participants were required to rest the index finger there until the target stimulus was presented; then participants should press one of the gray circles.

The stimuli comprised three symbols (#, §, and @). Participants responded to them according to one of two tasks, which were defined by different stimulus–response (S–R) mappings. Which task was to be used in the current trial was signaled by a cue – a red cross or a blue cross – preceding the target stimulus. Both the cue and the stimulus were displayed centrally in the top section. At the beginning of a block of trials, the S–R mappings of the red and of the blue task were shown on the screen for self-paced encoding. The display with the S–R mappings comprised a square frame (either in red or blue), in which the respective cue (also in red or blue) was presented centralized in the top, along with three rows, each with a symbol alongside the number of the response mapped to it (see Fig. 7, top panels).

5.1.3. Procedure and design

The experiment was performed in one 75-min session. Each trial consisted of the following sequence of events. To start a trial, the participant placed the index finger on the home button. When the response on the home button was detected, the task cue was presented for 300 ms (short CTI)

Author's personal copy

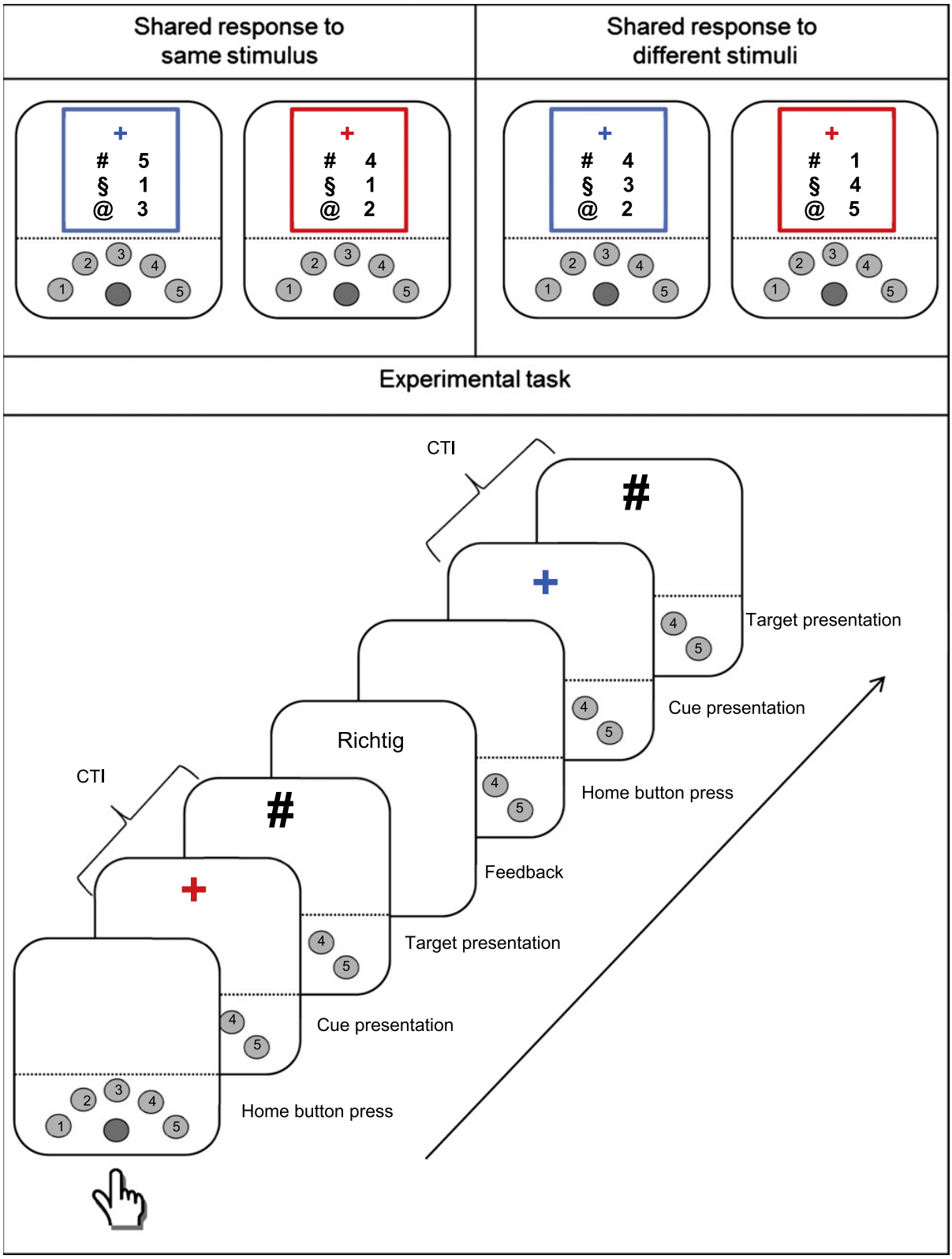

Fig. 7. Illustration of the experimental conditions of Experiment 3 (top panels) and schematic flow of events during the choice reaction task (bottom panel).

or 1000 ms (long CTI), after which the cue was removed and the stimulus was displayed. The stimulus remained on the screen until the participant touched one of the five response alternatives. After responding, feedback (the German words for correct or wrong) was provided for 500 ms, and then a

Author's personal copy

new trial was started when the participant pressed the home button. A short beep indicated when any touch-response was registered. Responses outside the black and gray circles were not recorded.

Participants worked with two task compositions in different blocks in counterbalanced order. In the *same-stimulus* task composition (left upper panel in Fig. 7), the shared response was assigned to the same stimulus in both tasks. The other two stimuli were mapped to unique responses not shared between the tasks. For instance, the responses for the three stimuli (in the order #, §, and @) could be buttons [1 3 4] for the red task and buttons [2 5 4] for the blue task. This task composition corresponds to the *same-position* list composition in Fig. 2A. In the *different-stimulus* task composition (right upper panel in Fig. 7), the shared response was assigned to different stimuli. For instance, the three stimuli could be mapped to the buttons [1 3 4] in the red task and to buttons [2 5 3] in the blue task. This task composition corresponds to the *different-position* list composition in Fig. 2B. The stimulus–response mappings used in each task composition were created randomly for each participant. They stayed constant within each block.

Each block began with 80 practice trials. During the first 40 trials of the practice phase, participants practiced the red and the blue task alone, alternating between tasks after every 10 trials. During the remaining 40 practice trials switches between the red and blue cues occurred randomly with a .5 probability. Each stimulus had a one-third probability of occurrence. The first five trials in a run used one level of CTI (short or long), and the remaining five trials used the other level of CTI (long or short); the order of CTIs was determined at random for each run.

The test trials of each block, following the practice phase, were divided into six runs of 70 trials each. The test trials were as described for the second half of the practice phase – cue repetitions or switches occurred with a .5 probability, and each target stimulus occurred with equal probability. Each run used one CTI in the first 35 trials and the other CTI in the second 35 trials; the order of CTIs was randomly determined for each participant. At the beginning of each run, participants were reminded of the S–R mappings of the red and the blue task, and between runs they were encouraged to take a short break.

5.2. Results

Accuracy was again high ($M = .96$, $SD = .04$). RTs for incorrect responses, or following incorrect responses, were removed from the analyses. Outliers, defined by the same criteria as in the preceding experiments (2.3% of RTs), were also removed. Table 5 shows the means for the design cells obtained by crossing task repetition, stimulus repetition, and response repetition.

5.2.1. Task-switch cost and preparation interval

We first ran a conventional analysis of the task repetition effect as a function of preparation interval (CTI). As is usually done in the task-switching literature, we excluded stimulus-repetition trials. We also excluded response-repetition trials to avoid an artificial inflation of the task-switch cost by the interaction of task repetition with response repetition (see below). An ANOVA with Task Composition (shared response to same stimulus vs. different stimuli), Task Repetition, and CTI revealed a significant cost of switching to a new task, $F(1, 23) = 13.4$, partial $\eta^2 = .37$, $p = .001$. Mean RTs for task repetitions were 1.71 s (SE = .05), compared to 1.79 s (SE = .06) for task switches. The interaction of task repetition with CTI just failed the criterion for significance, $F(1, 23) = 3.5$, partial $\eta^2 = .13$, $p = .08$. It was qualified by the three-way interaction including task composition, $F(1, 23) = 5.6$, partial $\eta^2 = .20$, $p = .03$. The task-switch cost declined with a longer CTI when the repeated response was mapped to different stimuli in the two tasks, but not when it was mapped to the same stimulus. Responses were generally faster when the shared response was mapped to the same stimulus, $F(1, 23) = 14.1$, partial $\eta^2 = .38$, $p = .001$, for the main effect of task composition. This main effect probably arose because there were congruent trials only in the task composition with the shared response mapped to the same stimulus. The main effect of CTI was not significant ($F < 1$).

A follow-up analysis for only the long CTI showed that the 53 ms task-switch cost was still significant, $F(1, 23) = 5.8$, partial $\eta^2 = .20$, $p = .02$, in agreement with the common finding of residual task-switch costs after long preparation intervals (Meiran, Chorev, & Sapir, 2000).

Author's personal copy

Table 5
Mean response times (standard deviations in parentheses) by task repetition, stimulus repetition, and response repetition conditions separately for each CTI level in Experiment 3.

S–R composition (*Congruency*)	Response repetition condition	Task repetition		Task switch	
		Stimulus Repetition	Stimulus Switch	Stimulus Repetition	Stimulus Switch
CTI = 0.3 s					
Diff. stimulus	Resp. repetition	**1.48 (0.22)**			**2.12 (0.58)**
(all incongruent)	Resp. switch		***1.77 (0.33)***	1.73 (0.25)	***1.89 (0.38)***
Same Stimulus	Resp. repetition	**1.45 (0.16)**			
(incongruent)	Resp. switch		***1.77 (0.29)***	1.71 (0.33)	***1.86 (0.38)***
Same Stimulus	Resp. repetition	**1.34 (0.12)**		**1.40 (0.14)**	
(congruent)	Resp. switch		***1.42 (0.13)***		***1.44 (0.13)***
CTI = 1.0 s					
Diff. stimulus	Resp. repetition	**1.53 (0.22)**			**1.95 (0.66)**
(all incongruent)	Resp. switch		***1.79 (0.32)***	1.76 (0.32)	***1.83 (0.36)***
Same stimulus	Resp. repetition	**1.51 (0.26)**			
(incongruent)	Resp. switch		***1.74 (0.30)***	1.69 (0.47)	***1.83(0.38)***
Same stimulus	Resp. repetition	**1.36 (0.11)**		**1.45 (0.17)**	
(congruent)	Resp. switch		***1.46 (0.12)***		***1.48 (0.14)***

Note: Means used to assess congruency effects (Fig. 8) and response-repetition effects (Fig. 9) are printed in bold. Data used to assess the task-repetition effect are printed in italics.

Analogous analyses with error rates only revealed a main effect of task repetition: More errors were committed after a task switch (5.2%, SE = 1.0) than after a task repetition (3.4%, SE = 0.8).

5.2.2. Congruency effect

We analyzed the effect of congruency in the task composition where the shared response was mapped to the same stimulus in both tasks (left upper panel of Fig. 7). In line with the preceding experiments, we analyzed the effect of congruency in the three transition types that enable a comparison of congruent and incongruent trials: task repetitions with response repetitions, task repetitions with response switches, and task switches with response switches. The relevant RT means are plotted in Fig. 8; the layout of this figure is analogous to Fig. 5, which shows the corresponding list-switch results from Experiment 2. As expected, RTs were substantially faster in congruent than in incongruent trials, $F(1, 23) = 42.4$, partial $\eta^2 = .65$, $p < .001$. The congruency effect interacted with type of transition, $F(2, 46) = 22.7$, partial $\eta^2 = .50$, $p < .001$. Compared to the other two transitions, the congruency effect was relatively small when the task and the stimulus repeated (129 ms, compared to 318 ms for task repetition with stimulus switch, and 382 ms for task switch with stimulus switch), but the reduced congruency benefit was still significant, $F(1, 23) = 25.0$, partial $\eta^2 = .52$, $p < .001$. The main effect of transition type, $F(2, 46) = 42.4$, partial $\eta^2 = .65$, $p < .001$, reflects the observation that RTs were fastest for the condition in which both task and stimulus were repeated, slower when the stimulus switched, and slowest when both task and stimulus switched.

The effect of congruency was not significantly modulated by CTI, $F(1, 23) = 2.5$, partial $\eta^2 = .10$, $p = .13$. This finding replicates the absence of a Congruency × CTI interaction in Yehene and Meiran (2007).

An analysis of ERs confirmed the interaction of congruency with transition type already seen in the RT data: Less than 1% errors were made on congruent trials regardless of transition type. On incongruent trials, error rate was 0.1% (SE = 0.1) for task repetitions with stimulus and response repetitions; ERs were higher for transitions to another stimulus after a task repetition (3.9%, SE = 0.9) and after a task switch (5.6%, SE = 1.1). The ANOVA reflected this pattern by main effects of congruency, $F(1, 23) = 20.7$, partial $\eta^2 = .47$, $p < .001$, of transition type, $F(2, 46) = 21.4$, partial $\eta^2 = .48$, $p < .001$, and their interaction, $F(2, 46) = 19.2$, partial $\eta^2 = .45$, $p < .001$. No effect involving CTI reached significance.

Author's personal copy

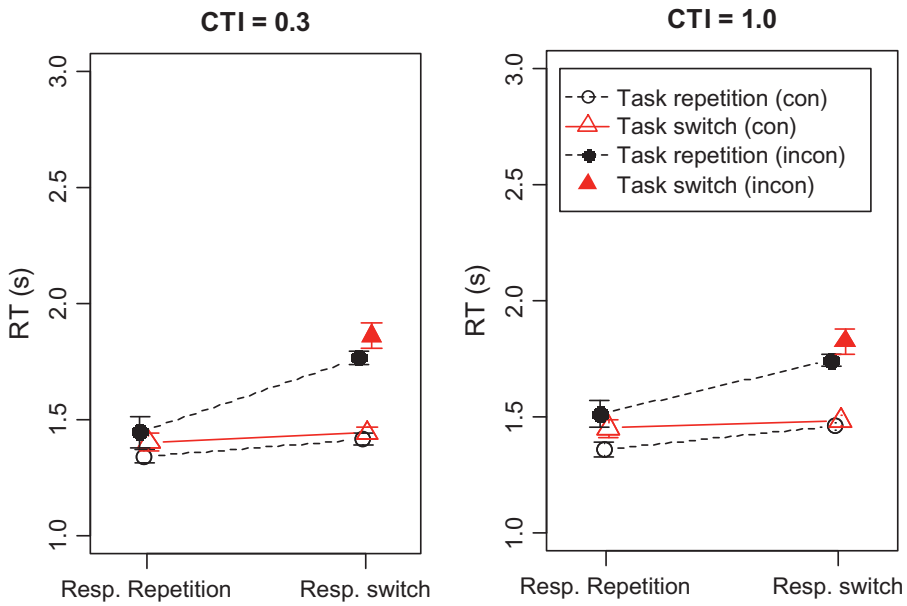

Fig. 8. Mean response times from Experiment 3: Effects of congruency, response-repetition, and task repetition, in the condition where the same response was given to the same stimulus in both tasks, for two levels of cue-target interval (CTI). Error bars are 95% confidence intervals for within-subject comparisons.

5.2.3. Response repetition effects

The third set of analyses concerns the effects of response repetition and their modulation by task switching. We first analyze RTs from the different-stimulus task composition (right upper panel in Fig. 7); this analysis avoids confounds with congruency because all stimuli are incongruent. After that we analyze congruent trials from the same-stimulus task composition (left upper panel in Fig. 7).

5.2.3.1. Incongruent trials. The critical interaction of task repetition with response repetition is shown in Fig. 9 for the two levels of CTI (analogous to Fig. 6 for the list-repetition by item-repetition interaction in Experiment 2). An ANOVA confirmed the interaction of task repetition with list repetition, $F(1, 23) = 22.5$, partial $\eta^2 = .50$, $p < .001$. As predicted, response repetition resulted in a benefit when the task was repeated, but in a cost when the task was switched. This pattern was not affected by the preparation interval, as shown by the non-significant interaction of task repetition, response repetition, and CTI ($F = 1.3$). The absence of this three-way interaction deviates from Yehene and Meiran (2007), but is in full agreement with our corresponding finding in Experiment 2.

In addition, the main effect of task repetition was significant, $F(1, 23) = 25.3$, partial $\eta^2 = .52$, $p < .001$, as was the small main effect of response repetition, $F(1, 23) = 4.9$, partial $\eta^2 = .18$, $p = .04$. Consistent with the analysis in the preceding section, the main effect of CTI was not significant, $F(1, 23) = 3.7$, partial $\eta^2 = .14$, $p = .07$. The task-switch cost, however, declined with a longer CTI, $F(1, 23) = 6.5$, partial $\eta^2 = .22$, $p = .02$.

An analysis of error rates yielded some additional information. Response repetition was generally beneficial for error rates (1.6%, SE = 1.0 for response repetitions, vs. 5.3%, SE = 1.4, for response changes), $F(1, 23) = 28.7$, partial $\eta^2 = .56$, $p < .001$. Task repetitions were also beneficial (2.1%, SE = 0.7 vs. 4.8%, SE = 1.6), $F(1, 23) = 7.1$, partial $\eta^2 = .23$, $p = .01$. In deviation from the RT results, these two variables did not interact, $F < 1$, such that there were response-repetition benefits also after a task switch. CTI had no main effect on error rates and entered into no interaction.

5.2.3.2. Congruent trials. A comparison of Figs. 8 and 9 shows that the interaction of task repetition and response repetition observed among the incongruent trials was much reduced among the congruent trials. For congruent trials, the dominant effects were the two (beneficial) main effects of task repetition, $F(1, 23) = 10.2$, partial $\eta^2 = .31$, $p = .004$, and response repetition, $F(1, 23) = 13.7$, partial $\eta^2 = .37$,

Author's personal copy

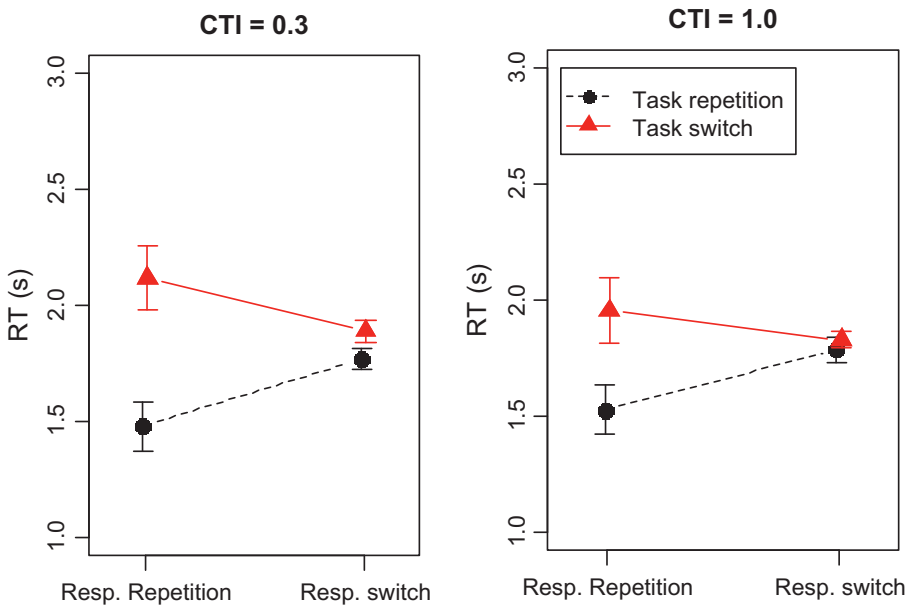

Fig. 9. Mean response times from Experiment 3. Effects of response repetition for task repetition and task-switch trials in the condition where the same response was given to different stimuli in the two tasks, for the two levels of cue-target interval (CTI).

$p = .001$; their interaction was just significant, $F(1, 23) = 4.6$, partial $\eta^2 = .17$, $p = .04$. The finding that task congruency dampens the interaction of task repetition and response repetition contrasts with the absence of such dampening in Yehene and Meiran (2007). It corresponds well with the finding in Experiments 1 and 2, where list congruency eliminated the interaction of list repetition and item repetition.

5.3. Discussion

The purpose of Experiment 3 was to determine whether deviations from the analogy between list-switch and task-switch experiments reflect differences in ancillary features of the experimental paradigms or substantive differences between declarative and procedural WM. To that end we designed a task-switch experiment matching as closely as possible the list-switch paradigm used in our preceding two experiments. The results closely mirrored those obtained with the list-switch paradigm, in particular those of Experiment 2.

First, in all our experiments, set switching incurred a cost. Responses were slower when they involved a switch to a new list (Experiments 1 and 2) or to a new task (Experiment 3). One difference between Experiments 2 and 3 was that list-switch cost disappeared entirely after a long CTI in Experiment 2. In contrast, a residual task-switch cost after the long CTI was observed in Experiment 3. In our previous study, however, we found a residual switch cost also with list switching (Souza et al., 2012). Further research is needed to establish whether the residual list-switch cost is a robust phenomenon.

Second, there was a benefit of task congruency that was not reduced by a longer preparation time. This mirrors the list-congruency benefit in the previous two experiments, which was only slightly diminished with preparation time.

Third, we replicated the well-established interaction of task repetition with response repetition in RTs, corresponding to the interaction of list repetition and item repetition in Experiments 1 and 2. Different from the RT data, we observed no response-repetition cost in the ERs. Our finding of response-repetition benefits after a task switch in ERs is untypical in light of previous studies (Hübner & Druey, 2006; Mayr & Kliegl, 2003; Schuch & Koch, 2004), but it is not without precedent: At least two previous studies found response-repetition costs in RTs but response-repetition benefits in ERs after task switches (Brown, Reynolds, & Braver, 2007; Meiran & Kessler, 2008). Because Experiments 1 and 2 had ERs too small to carry discernible effects, we limit comparisons between Experiment 3 and the preceding experiments to the RT results.

Author's personal copy

Important in the present context are the two results speaking to the apparent exceptions from the pattern of analogous effects in list-switch and task-switch paradigms. First, the interaction of task repetition with response repetition was strongly reduced by congruency. Second, this interaction was not affected by preparation time. In both cases, these results of Experiment 3 deviated from those of Yeheene and Meiran (2007), and corresponded to those of our Experiment 2. We conclude that the analogy holds when closely parallel task-switch and list-switch paradigms are compared, and the two apparent exceptions arose from differences in features of the experiments, rather than from differences in the processing mechanisms of declarative and procedural WM.

6. Theoretical implications of Experiments 1–3

The main hypothesis guiding our investigation was that declarative and procedural WM operate in analogous ways. We obtained evidence for this analogy from three phenomena: (1) In analogy to the task-switch cost, we observed a list-switch cost; in both paradigms, these set-switch costs declined with longer preparation time. (2) In analogy to the task-congruency effect in the task-switch paradigm, we found a list-congruency effect in the new list-switch paradigm. Both congruency effects were largely unaffected by preparation time. (3) In task-switch studies, repeating a response is beneficial when the task remains the same across two successive trials, but costly when the task switches. We obtained the analogous pattern in declarative WM: Repeated access to the same item yielded a benefit when the list remained the same, but incurred a cost when the list was switched. Together, these findings support the proposed analogy between benchmark effects in task-switching and corresponding effects in list-switching. These findings suggest that analogous mechanisms and processes are engaged in selecting task sets and responses in procedural WM, and in selecting memory sets and items in declarative WM. We next consider implications of our findings for characterizing these mechanisms.

6.1. Set switch costs and congruency effects

Two major explanations of the task-switch cost have for a long time competed in the literature: According to one view the additional time needed for task-switch trials reflects the time for reconfiguration of the task settings (Rogers & Monsell, 1995). The alternative view emphasized the role of proactive interference from the preceding task that slows down response selection after a task switch (Allport, Styles, & Hsieh, 1994). The more recent literature is converging towards an integration (e.g., Vandierendonck et al., 2010), recognizing that previously used task sets create proactive interference, and that reconfiguration of task settings is needed to overcome this interference.

Within our framework, switching to a new task requires updating of the task representation in the bridge (procedural WM), and switching to a new memory set requires updating the content of the direct-access region (declarative WM). This updating process involves removal of the old set, cue-based retrieval of the new set from LTM, and configuring the new set by forming temporary bindings between stimulus and response categories (in the bridge), or between context cues and items (in the direct-access region). Thus, the updating process can be described as reconfiguration of a mental set, and at the same time it serves to reduce proactive interference from the previously used set.

The effect of proactive interference from the not-selected set is reflected in congruency effects. Congruency effects can arise from two sources. First, they could reflect residual traces of bindings belonging to the old set in the central WM components (i.e., the bridge and the direct-access region). To the degree that removal of the old set during updating is incomplete, outdated stimulus–response bindings or context-content bindings linger on and contribute to proactive interference (on incongruent trials) or facilitation (on congruent trials). Evidence for lingering effects of outdated bindings comes from a study of WM updating (Oberauer & Vockenberg, 2009): Participants find it easier to establish a new position-item binding in declarative WM when the new binding is congruent with a binding maintained previously, even when that binding had been outdated in the meantime.

The second potential source of congruency effects is learning of stimulus–response associations and context-item associations in LTM. Research on task congruency offers evidence that proactive

Author's personal copy

interference in procedural WM arises at least in part from LTM. For instance, the task-congruency effect depends on the frequency with which an interfering (incongruent) S–R association has been used over the course of the experiment (Kiesel, Wendt, & Peters, 2007). In contrast, the congruency effect is unaffected by concurrent load on procedural WM (Kessler & Meiran, 2010). When the S–R mappings for one of the two tasks are reversed by a new instruction after an initial testing phase, then the subset of stimuli never presented for the reversed task continue to generate a congruency effect according to their pre-reversal mapping (Wendt & Kiesel, 2008). After the new instruction the stimulus–response bindings formed in the bridge should be re-mapped. Therefore, the congruency effect reflecting the outdated mapping must reflect the influence of long-term learning of stimulus–response associations. Consistent with this interpretation, a task-congruency effect cannot be generated by instruction alone but requires actually carrying out the instructed task (Waszak, Wenke, & Brass, 2008).

Our finding that list-congruency and task-congruency effects are hardly reduced with a longer preparation interval is consistent with the view that these congruency effects reflect primarily interference from LTM (Yamaguchi & Proctor, 2011). Yet, as we will show through the model-based investigation reported in the next section, this finding can also arise from residual bindings of the currently not relevant list or task in the central components of WM. It is reasonable to assume that switching to a new list or task requires removing the old list or task to some degree, but removal is incomplete. The representation of the currently not relevant list or task might be diminished in strength until the degree of interference from it falls below a tolerable level, regardless of the length of the preparation interval. The size of the congruency effect would then be determined by the tolerable level of interference and not by the preparation interval.

To conclude, there are good reasons to assume that both associations in LTM and residual bindings in the central components of WM contribute to list congruency and task congruency effects. Below we will explore which of these sources are needed in the context of our model to explain congruency effects in different experimental settings.

6.2. Item repetition and response repetition effects

Repeating a response across trials typically has a beneficial effect on RTs and ERs when the task is repeated, but a cost when the task is switched. We demonstrated the analogous pattern for item repetitions in declarative WM: Item repetition is beneficial when the list remains the same, but incurs a cost when the list is switched.

The two strongest contenders to date for explaining the pattern of response-repetition effects in task switching have both been sketched first by Rogers and Monsell (1995). The association-based explanation (Fig. 10, left panel) builds on the assumption that a task set consists of associations between stimulus categories and responses, the strength of which is modulated by response selection. For instance, an experiment could involve switching between parity judgments and magnitude judgments on digits. Execution of a response to the digit 7 in the parity task strengthens the association of that response with the stimulus category that led to the response (“odd”), and at the same time weakens the association of that response with the stimulus category of the alternative task (“large”). Thus, when the same response is to be given again within the same task (e.g., a parity judgment on 9), it is facilitated, but when the same response must be given within the other task (e.g., a magnitude judgment on 9) it is slowed. This explanation was endorsed by Schuch and Koch (2004), and Meiran (2000) has incorporated a similar idea in his model of task switching. Altmann (2011) has recently presented a more elaborate version of an association-based explanation.

In contrast, the priming-inhibition account (Fig. 10, right panel) explains the pattern of response repetition effects as the net effect of two opposing mechanisms (Hübner & Druey, 2006). One is the inhibition or suppression of responses after they have been executed. Response suppression is commonly assumed in models of sequential action (Li, Lindenberger, Rünger, & Frensch, 2000; MacKay, 1987) and of serial recall (Henson, 1998a) as a mechanism to prevent perseverative errors. The opposing mechanism, according to this model, is repetition priming of the stimulus or stimulus category (e.g., “odd”). Studies of response-repetition effects have mostly used categorization tasks, with one-to-one mappings between stimulus categories and responses. As a consequence, in cases of task repetition, response repetition is inevitably accompanied by a repetition of the relevant stimulus

Author's personal copy

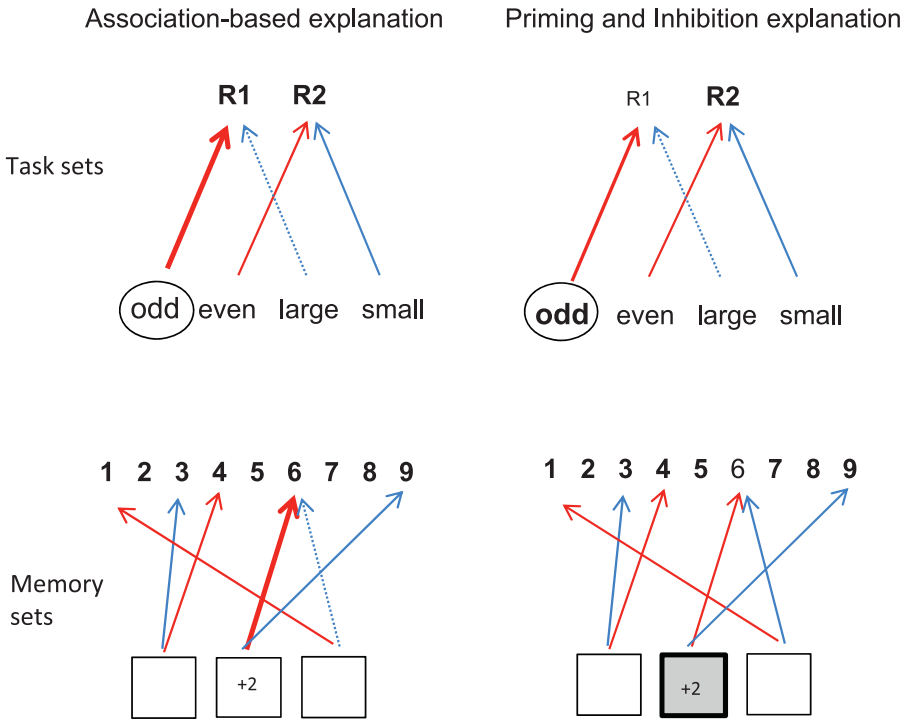

Fig. 10. Top: Illustration of two explanations for the interaction of task repetition and response repetition, using switching between parity judgments (red task) and magnitude judgments (blue task) on digits as example. Each panel shows the hypothetical state of task representations after a trial where a digit was classified as odd according to the parity task, and response 1 (R1) was given. According to the association-based explanation (left panel), the association between stimulus category "odd" and R1 is strengthened, and at the same time, the association between the stimulus category "large" and R1 is weakened. If the next trial again involves parity judgment, repeating R1 (in response to another odd digit) will be facilitated, whereas after a task switch, repeating R1 (in response to a large digit) is made difficult, relative to giving the alternative response R2. According to the priming-inhibition account (right panel), R1 is suppressed, and at the same time, the stimulus category "odd" is strengthened. When the task is repeated, repeating R1 implies repeated use of the primed category "odd". The priming effect is stronger than the inhibition of R1, leading to a net response-repetition benefit. When the task is switched, repeating R1 implies use of the category "large", which is not primed. As a consequence, the inhibition of R1 renders response repetition more difficult than response alternation. Bottom: Application of the two explanations to the list-switching paradigm. Each panel shows the state of list representations after digit 6 has been retrieved from the middle position of the red list. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

category. If priming of the stimulus category has a stronger effect than response suppression, a net benefit of response repetition results in the case of task repetitions. After a task switch, however, a response repetition is no longer accompanied by a repetition of the stimulus category, because stimuli are now categorized in a different way (e.g., as "large"). Thus, there is no stimulus-category priming to counteract the effect of response suppression, and the latter becomes manifest in a response-repetition cost. The same happens when stimuli assigned to the same response do not form a category (Campbell & Proctor, 1993; Smith, 1968).

How could the two explanations of response repetition effects be applied to the item-repetition effects we observed in declarative WM? The association-based explanation could be applied as follows: When a digit is selected from a list, the association of that digit with its position in the respective list is strengthened, and at the same time, the association of that digit to other positions is weakened. Accessing the same item again in a list-repetition trial is facilitated because the position-item association has been strengthened. In contrast, when after a list switch, the same digit is cued by a different position, access is slowed because the association of this digit to the new position has been weakened (see Fig. 10, bottom left panel).

The priming-inhibition explanation could also be applied to declarative WM. Extending the common assumption that list items are suppressed after output in a serial-recall task (Henson, 1998a), we can assume suppression of items in declarative WM after they have been used (e.g., as input for

Author's personal copy

an arithmetic operation). In list-repetition trials, suppression would be over-compensated by priming of the position to which the item is bound, resulting in a net repetition benefit. In list-switch trials, when the same item is accessed in a different position, the effect of item suppression becomes manifest as a repetition cost (see Fig. 10, bottom right panel). Thus, the priming-inhibition account and the associative account make the same predictions for our experiments. In fact, the difference between these two accounts is subtle. In our modeling efforts we incorporated elements of both approaches. As we will show, neither of these elements is dispensable for a satisfactory account of the data.

Our finding of an item-repetition benefit when the list is repeated, together with an item-repetition cost when the list is switched, provides a challenge for current theorizing about item-repetition effects in declarative WM. The item-repetition benefit reflects the well-established object-switch cost or item-switch cost in single-list experiments (Garavan, 1998; Oberauer, 2003; Verhaeghen & Basak, 2005). The dominant explanation for this finding has been that the last used item achieves a status of privileged accessibility that carries over to the next trial. The present finding of an item-repetition cost when the list is switched contradicts this idea. Therefore, the item-repetition benefit (or item-switch cost) cannot be explained by assuming that the item itself remains in a privileged state after completion of a trial. Rather, an explanation along the lines sketched above for response-repetition effects is called for. We will offer such an explanation in the context of our model.

7. A computational model of selection of representations in WM

Experiments and theorizing in the fields of WM and action control have reached a level of complexity that challenges the capacity of unaided human reasoning. This is where computational modeling becomes indispensable as a theorizing tool (Farrell & Lewandowsky, 2010; Lewandowsky & Farrell, 2011). For instance, in the discussion of potential mechanisms for explaining our results we have invoked several processes that, on their own, would plausibly generate the effects they are meant to explain. However, it is impossible to determine without a computational implementation how these mechanisms behave when they are all put together. Therefore, we developed a simple connectionist model that implements the theoretical assumptions we made above. The primary purpose of this modeling work is to check whether the theoretical explanation we offered so far is coherent and feasible, and to generate new predictions. The model presented below is not meant to be a comprehensive model of declarative and procedural WM, although we believe that it could provide the starting point for a future more comprehensive model.

We describe the model initially as a model for declarative WM, applied to the list-switch paradigm. Later we show how the same architecture can also be used as a model of procedural WM, to be applied to the task-switch paradigm. Here we present the model informally; the full set of equations can be found in Appendix A. The MATLAB code can be downloaded from the first authors' home page: http://www.psychologie.uzh.ch/fachrichtungen/allgpsy/Team/Oberauer_en.html. The model architecture is shown in Fig. 11. The model consists of two modules, one for the selection of individual representations (i.e., memory items or responses) from a set, and one for the selection of sets (i.e., memory sets or task sets).

7.1. The item-selection module

7.1.1. Architecture

The module for selection of items (on the right of Fig. 11) is built on well-established principles of models of serial recall in the WM literature (Burgess & Hitch, 1999; Henson, 1998b; Lewandowsky & Farrell, 2008). It consists of two fully interconnected layers of units, the *input layer* and the *output layer*, together with a candidate layer that has one-to-one connections with corresponding units in the output layer. Activation patterns in the input layer represent the current list position, and activation patterns in the output layer represent the currently encoded or retrieved list item. The candidate layer is a copy of the output layer, it serves to represent the current memory strength of each item independent of their bindings to list positions. By default, all items eligible for retrieval in the current task context receive activation from the candidate layer, thereby setting them apart from other mem-

Author's personal copy

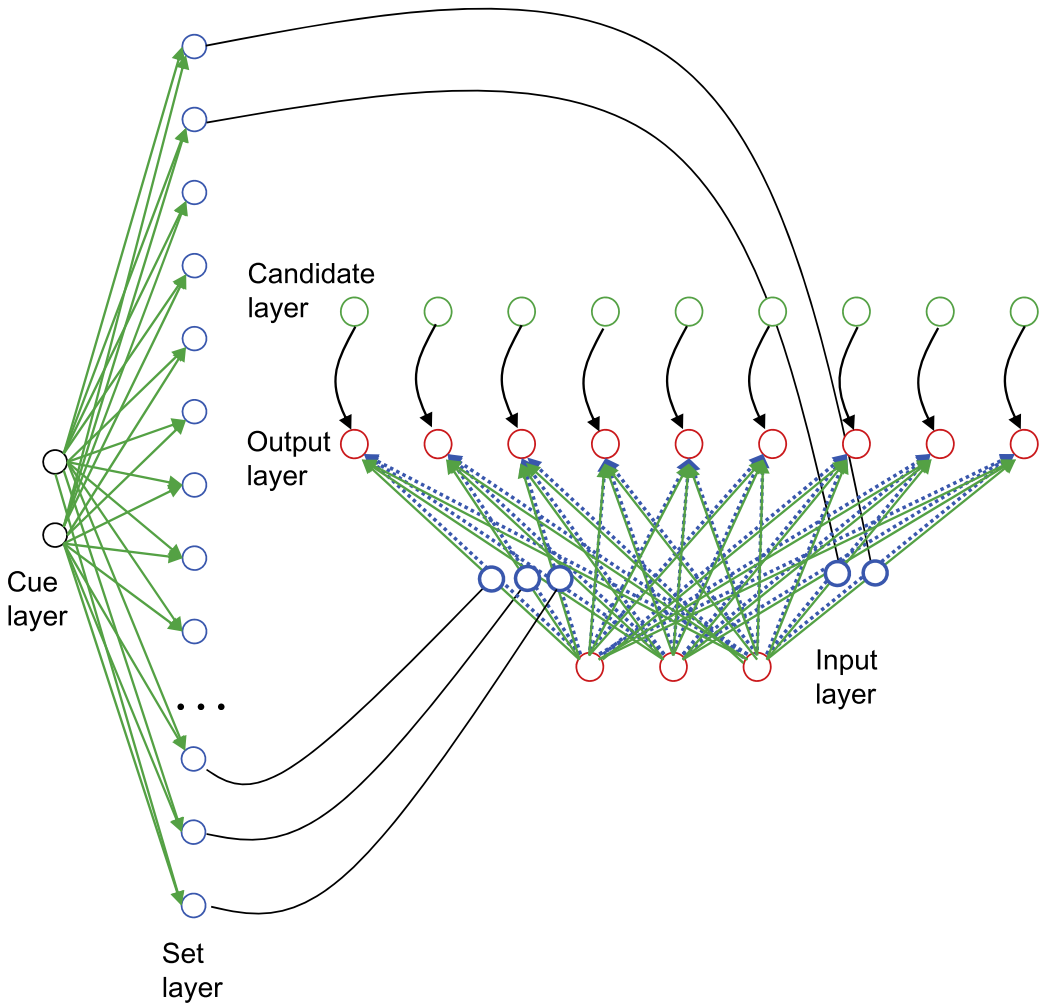

Fig. 11. Architecture of the computational model, as applied to the list-switching paradigm. The model consists of an item-selection module (i.e., the horizontal three-layer structure on the right) and a set-selection module (i.e., the vertical two-layer structure on the left). The item-selection module consists of an input layer for representing list positions, an output layer for representing items, and a candidate layer. Input and output layer are connected by slow-changing associations (green continuous lines) and fast-changing bindings (blue broken lines). The bindings are established by gain-modulating units, a subset of which is represented by small blue circles. Each unit of the output layer is linked by a fixed connection to a corresponding unit in the candidate layer. The set-selection module consists of a cue layer for representing list cues, and a set layer. Each unit in the set layer has a connection to the corresponding gain-modulating binding unit; a sub-set of these connections is shown as lines between set-layer units and binding units. Green components in the figure correspond to the activated part of LTM in the theoretical framework as sketched in Fig. 1; blue components correspond to the region of direct access, and red components to the focus of attention.

ory representations not relevant in the current task context. For instance, when the task involves lists of digits, all digits are activated in the candidate layer, whereas letters are not; this ensures that no letter is accidentally retrieved in place of a digit.

For simplicity we used localist representations for the items (i.e., the digits from 1 to 9), that is, each item is represented by an activation pattern across the output layer in which only one unit is active. Mathematically, an item is a vector of nine values, eight of which are zero and one of which is set to 1. The positions are represented in a similar fashion. Each position is represented by an activation peak in one unit corresponding to this position. However, the units corresponding to neighboring positions also receive some activation; the degree of which falls off exponentially with positional distance. This overlapping scheme reflects the assumption that the distinctiveness of positions is limited; this explains the high prevalence of order errors in serial recall, which tend to consist of confusions with close neighbors (Henson, 1998b). In the application of the model to the present experiments the position overlap plays no explanatory role, so we fixed it to a minimum value to facilitate interpretation of

Author's personal copy

the model's behavior. Larger degrees of overlap did not change the model's prediction for the present experiments.

7.1.2. Encoding

Encoding a new list into WM proceeds by activating each item in turn in the output layer, and the corresponding list position in the input layer. Their association is learned by updating the connection weights in the weight matrix **w** connecting the two layers. Whereas serial-recall models use standard Hebbian learning for this step, we use a learning algorithm known as the *Widrow-Hoff rule* or the *delta rule*, because it is particularly suited for rapid updating. This algorithm starts by generating an expectation for the incoming item by cueing the system with the new item's position. That is, the new position is activated in the input layer, and activation is forwarded to the output layer through the weight matrix. The resulting activation pattern in the output layer is the system's expectation; it reflects what is currently associated to the new position. The algorithm subtracts the expectation from the representation of the actual stimulus, and associates the difference to the position representation by Hebbian learning. Thus, the change of the weight matrix is determined by:

$$\Delta \mathbf{w} = \eta(\mathbf{s} - \mathbf{e})\mathbf{p}^T \tag{1}$$

with **w** for the weight matrix, η for the learning rate, **p** for the position vector currently active in the input layer (transposed), **s** for the item vector of the new stimulus, and **e** for the expectation vector. The delta rule is well suited for updating representations in WM because it establishes new links and removes old links at the same time. This can easily be seen by decomposing Eq. (1) into a part that adds the link from the current position to the new stimulus **s**, and a part that removes links from that position to any old contents, reflected in the expectation **e**:

$$\Delta \mathbf{w} = \eta \mathbf{s} \cdot \mathbf{p}^T - \eta \mathbf{e} \cdot \mathbf{p}^T \tag{1a}$$

In our model, the connections between input and output layer are governed by two weight matrices, one with a fast and one with a slow learning rate (for another model using parallel fast and slow weights see Burgess & Hitch, 1999). The fast learning weight matrix, $\mathbf{w_b}$, (blue dotted lines in Fig. 11) represents bindings in the region of direct access that can be rapidly updated. The slow learning weight matrix, $\mathbf{w_a}$, (green solid lines) represents associations between positions and items in activated LTM. During each learning step both weight matrices are updated by the same learning rule, but using different learning rates, η_b and η_a, respectively.

7.1.3. Retrieval

Once a list is encoded, individual items can be accessed by cueing them with their positions. A cued position activates its representation in the input layer. That activation is forwarded through the combined weight matrix (i.e., the sum of $\mathbf{w_a}$ and $\mathbf{w_b}$) to the output layer. In addition, each unit in the output layer receives activation from the corresponding unit in the candidate layer. The weights between the candidate layer and the output layer are fixed to 1, so that each candidate unit forwards its current level of activation to the corresponding output unit. The resulting activation of each item in the output layer thus reflects the sum of two sources of activation, one from the input layer, the other from the candidate layer. Activation from the input layer, mediated by the combined weight matrix, reflects the strength of cueing each item by the given position. Activation from the candidate layer reflects the current memory strength of each item independent of cueing by a particular position. Loosely speaking, the activation arriving at an output unit j from the candidate layer reflects the system's unconditional probability that item j is the correct item to be selected, and the activation arriving from the input layer through the combined weight matrix reflects the conditional probability that j is to be selected, given the position currently active in the input layer.

To model the duration of retrieval we used the linear ballistic accumulator (LBA) model of Brown and Heathcote (2008). During retrieval, each unit of the output layer serves as a linear accumulator, which starts from zero (plus or minus some noise) and grows with a rate given by the sum of the activations sent to that output unit from the input layer and from the candidate layer. The accumulators race against each other towards a fixed threshold. When the first accumulator surpasses the threshold,

Author's personal copy

the item represented by the winning accumulator is chosen for retrieval. The duration of the retrieval process is the simulated retrieval time.[4]

7.1.4. Transitions between retrievals

Each retrieval event and its resulting response leave traces in the system that carry over into the next trial, and thereby generate the effects of trial-to-trial transitions that we investigated in the experiments above. First, each retrieval is accompanied by updating of the weight matrices $\mathbf{w_a}$ and $\mathbf{w_b}$. When one of the accumulators in the output layer reaches a boundary, all other output items are re-set to zero to obtain an unambiguous representation of the selected item in the output layer. The weight matrices are then updated by the delta rule, strengthening the binding and the association of the retrieved item to the current position. For this step, the learning rates η_a and η_b are reduced by a factor f because learning of items retrieved from memory is assumed to be less strong than learning of stimuli presented perceptually.

Second, the activation of the position representation in the input layer is dampened by multiplying it with a small value γ. As long as γ is not zero, some activation of the position carries over into the next trial. Thus, γ controls the amount of input priming (i.e., priming of an item's retrieval cue, such as its position in our list-switch paradigm, or stimulus priming in task switching).

Finally, after an item has been retrieved and a response has been made, the retrieved item is suppressed. Response suppression is a common mechanism in models of serial recall (Henson, 1998b; Lewandowsky & Farrell, 2008), and there is evidence that it is a necessary feature of these models (Farrell & Lewandowsky, 2004). We implemented response suppression by setting the retrieved item's activation in the candidate layer to zero. This suppression carries over into the following trials, but gradually wears off, because the suppressed item gradually re-gains activation from trial to trial. The rate of re-activation of candidate units after suppression is controlled by parameter s.

7.2. The set-selection module

7.2.1. Architecture

The item-selection module so far constitutes a simple but fully operative model of WM as long as a single list (or a single task set) is involved. However, it is not well suited yet for rapid switching between different lists (or task sets). The function of the set-selection module is to select the relevant memory set and establish it in the item-selection module. The set-selection module operates by the same principles as the item-selection module. It takes as input a representation of the cue indicating which memory set or list is relevant for the current trial (e.g., blue or red for the two memory lists). This input is represented in the *cue layer*, which is connected to the *set layer*. The set layer represents the selected memory set as a pattern of activation across its units. Each unit of the set layer is linked to one connection weight in the fast-learning matrix $\mathbf{w_b}$ of the item-selection module. Thus, the matrix of bindings in the item-selection module is re-represented as a pattern of activations in the set layer.

The rationale of this scheme is derived from the concept of *gain modulation* in computational neuroscience (Brozović, Abbott, & Andersen, 2008; Salinas & Thier, 2000). Gain modulation means that the signal transmitted from a neuron i to a neuron j is modulated by the activation of a third neuron $m(i,j)$. Gain modulation can be modeled by multiplying the activation of i with the activation of $m(i,j)$ to obtain the signal sent from i to j. Multiplicative gain modulation is formally equivalent to the modulation of activation by connection weights in conventional connectionist models, with the activation of $m(i,j)$ playing the role of the connection weight $w(i,j)$. In our model, the activation sent from input unit i to output unit j is modulated by the activation of the binding unit $w_b(i,j)$. The set of binding units constitutes the binding matrix $\mathbf{w_b}$. Thus, the binding matrix is not a matrix of connection weights in the conventional sense. Rather, it is a matrix of activation levels of gain-modulating units that modulates the connections between the input layer and the output layer in the item-selection module. Hence,

[4] We also ran the model with the leaky competing accumulator (LCA) model of Usher and McClelland (2001) instead of the LBA. In the LCA, the accumulators compete with each other through lateral inhibition. In addition, accumulators lose evidence at a constant rate, governed by the leakage parameter. For all simulations reported here, the results with LBA and LCA were virtually equivalent, so we report only the former.

Author's personal copy

updating $\mathbf{w_b}$ by the delta rule is not a form of learning through modifying connection weights, but rather consists of updating the activation levels of the gain-modulating units. For another working-memory model using gain-modulating units to bind items to positions see Botvinick and Watanabe (2007). The novel feature in our model is that each gain-modulating unit in $\mathbf{w_b}$ is connected to one unit of the set layer in the set-selection module. This enables learning an entire list as a chunk in the set-selection module, as follows.

7.2.2. Set learning

When a memory set or list is presented (e.g., at the beginning of each run of the present experiments), the item-selection module encodes it by adjusting its weights as described above. As a result, the position-item bindings of the list are represented in the pattern of activation in $\mathbf{w_b}$. After a complete list has been encoded, the activations of the gain-modulating units in $\mathbf{w_b}$ are read out into the corresponding units in the set layer. We can think of the activation pattern in the set layer as a chunk representing the list. This list chunk in the set layer is now associated to the current list cue (in the cue layer) by the delta learning rule with a slow set-learning rate η_s. In this way, the set-selection module gradually acquires an association between each list and its cue. The cue-to-list associations are reflected in the connection weights of the weight matrix $\mathbf{W}$ of the set-selection module. This process describes how a relational representation in WM, such as a list or a task set, is acquired as a unified chunk in LTM.

7.2.3. Set retrieval and updating of the item-selection module

The learned lists can now be used to carry out runs of trials in a list-switching experiment such as our Experiments 1 and 2. Each trial begins with a list cue, which is encoded in the cue layer. Activation is transmitted through $\mathbf{W}$ to generate an activation pattern in the set layer. We modeled the duration of set retrieval in the same way as the duration of item retrieval: The input into each unit of the set layer is gradually accumulated until the first set-layer unit reaches a threshold. The duration of this process is the set-retrieval time. Once set retrieval has finished, the resulting activation pattern in the set layer is used as input for updating the gain-modulating units in $\mathbf{w_b}$ that control the bindings in the item-selection module. The binding matrix is updated by the delta rule:

$$\Delta \mathbf{w_b} = \eta_c(\mathbf{w_c} - \mathbf{w_b}) \tag{2}$$

Here, $\mathbf{w_c}$ stands for the input from the chunk representation in the set layer (in matrix format). This process describes how chunk representations of memory sets are retrieved from LTM and established in the region of direct access as a matrix of position-item bindings.

The speed of updating the direct-access region is governed by the updating rate η_c. The model goes through several iterations of updating the binding matrix $\mathbf{w_b}$ with the input from the set layer; each iteration takes a constant time of 0.1 s. Updating stops when the absolute difference between $\mathbf{w_b}$ and $\mathbf{w_c}$ falls below a criterion β. Only after the criterion is reached, the representation of the position (i.e., the frame in which the arithmetic operation is presented) in the input layer is allowed to feed activation forward through $\mathbf{w_b}$ and $\mathbf{w_a}$ to retrieve the associated item in the output layer. The number of updating iterations determines the duration of preparation for the upcoming trial. In list-repetition trials, the criterion is reached more rapidly than in list-switch trials, and that difference generates the list-switch cost.

Simulated RTs are composed of three durations, the duration of set retrieval, the updating time (i.e., the number of updating iterations multiplied by the time constant), and the item retrieval time. Set retrieval and set updating can already commence during the preparation interval. Thus, the predicted response time equals

$$RT = \max(0, t_s + t_u - CTI) + t_i \tag{3}$$

with t_s for the set-retrieval time, t_u for the updating time, and t_i for the item-retrieval time.

The model has 12 parameters, which are summarized in Table 6. The table also indicates that five of these parameters were fixed, that is, set to a constant value uninformed by our data, and the other seven parameters were free, that is, we adjusted them by hand during the exploration of the model's behavior, but then kept them constant across all simulations in this article. Because in all our exper-

Author's personal copy

Table 6
Model parameters.

Parameter	Symbol	Full	Reduced 1	Reduced 2	Reduced 3
Association learning rate	η_a	**0**	**0**	**0**	**0**
Binding rate	η_b	1	1	1	1
Set learning rate	η_s	0.07	0.07	0.07	0.07
Set retrieval rate	r	**3**	**3**	**3**	**3**
Set updating rate	η_c	**0.5**	**0.5**	**0.5**	**0.5**
Strengthening factor	f	**0.3**	**0.3**	**0**	**0.8**
Criterion for set updating	β	**0.05**	**0.05**	**0.05**	**0.05**
Input priming	γ	**0.12**	**0.12**	**0.15**	**0**
Activation in candidate layer	α	**0.06**	**0**	**0.06**	**0.03**
Position distinctiveness	D	100	100	100	100
Noise on starting point in LBA	σ	0.03	0.03	0.03	0.03
Reduction of response suppression	s	0.5	0.5	0.5	0.5

Note: Free parameters are printed in bold.

iments accuracy was very high, we chose parameter values that generated perfect accuracy in the simulations, and hence, we only report RT results.

7.3. Relating the model to the three-embedded-components framework

Our model is a computational implementation of the three components of WM that we described in the introduction. The central component of declarative WM, the direct-access region, is modeled by the matrix of bindings in the item-selection module, $\mathbf{w_b}$. When the model is applied to procedural WM, $\mathbf{w_b}$ represents bindings between stimuli (in the input layer) and responses (in the output layer) in the bridge. These bindings are rapidly updated at the beginning of each trial, so that they represent – apart from a small residual – only the memory set (or task set) relevant for the current trial. The bindings in $\mathbf{w_b}$ are reflected in the activation pattern in the set layer of the set-selection module, so one could also point to this activation pattern as the current content of the direct-access region (or the bridge).

The activated LTM is reflected in several features of the model, implying that the various phenomena that have been attributed to activated LTM are unlikely to arise from a single mechanism. The tonic activation of retrieval candidates in the candidate layer reflects one aspect of activated LTM, the priming of items (in declarative WM) and of responses (in procedural WM) that are likely to become relevant in the current context. Another aspect of activated LTM is reflected in the slowly changing associations in the item-selection module, $\mathbf{w_a}$. Their strength represents the cumulative frequency of using (and thereby strengthening) position-item links, or stimulus–response links, across a number of trials. As such, they are obvious candidates for explaining congruency effects, because congruent trials benefit from the fact that the same position-item links (or stimulus–response links), are strengthened across trials with different lists (or tasks). However, one surprising result from our simulations, presented below, is that association learning in $\mathbf{w_a}$ plays only a minor role in explaining congruency effects.

The model also makes explicit that the role of LTM in list switching and task switching goes beyond priming of representations and associations. The whole set-selection module is a model of how chunks are learned and retrieved in LTM. This module serves a role that has also been attributed to activated LTM in our theoretical framework: It maintains a representation of the currently not selected list or task set. As such, it can be used to outsource retention of information that is currently not needed in the region of direct access or the bridge, but that must be brought back later (Lewis-Peacock, Drysdale, Oberauer, & Postle, 2011; Mayr & Kliegl, 2000; Oberauer, 2005).

Least conspicuous in the model description so far is the focus of attention (and its procedural counterpart, the response focus). What to identify as the focus in the model depends on whether the focus is defined by its function or by what has so far been regarded as its empirical signatures. The function of the focus of attention in declarative WM is to single out one item as the input for, or object of, the

Author's personal copy

next cognitive operation. Likewise, the function of the response focus in procedural WM is to single out one response for execution. In the model this function is served by the current pattern of activation in the output layer of the item-selection module, together with the activation pattern in the input layer that serves as a cue to it. Therefore, we regard the current content of the input and the output layer as the content of the model's focus.

One major empirical signature of the focus of attention in declarative WM, however, has been the item-repetition benefit, or item-switch cost (Garavan, 1998; Oberauer, 2003). As we have argued above, the item-repetition benefit cannot be explained by persistent activation of the selected item in the focus of attention. Therefore, in our model, activation of the selected item does not persist across trials. The output layer is reset to zero at the end of a trial, and in the candidate layer the retrieved item is even suppressed below baseline. The sources of item (and response) repetition benefits lie elsewhere. One source of the item-repetition benefit after a list repetition is priming of the cue by which the item is accessed (e.g., its position in the present experiments, or the geometric figure in Garavan's counting task): A squashed representation of this cue in the input layer carries over into the next trial. The other source is strengthening of the cue-item binding. Thus, the item-repetition benefit is better characterized as a benefit of repeating the cue and the cue-item conjunction. Analogously, the response-repetition benefit after a task repetition arises from priming of the stimulus (or stimulus category), and strengthening of the stimulus–response binding.

7.4. An application to list switching

Each simulation reported in this article was run with 100 simulated subjects, each completing 50 runs of 20 trials per list composition. We first present a simulation of Experiment 2, using the parameter values for the full model in Table 6. For each subject the testing phase was preceded by a learning phase in which the item-selection module encoded each list, and then the set-selection module learned to associate that list as a chunk to its list cue. Each list was learned 20 times during the learning phase.

Figs. 12 and 13 show the simulated RTs for Experiment 2 in the same format as Figs. 5 and 6, respectively. The model reproduced all the effects observed in the experiment. Both figures show that list-switch trials are slower than list repetitions, and that RTs are overall faster with a longer

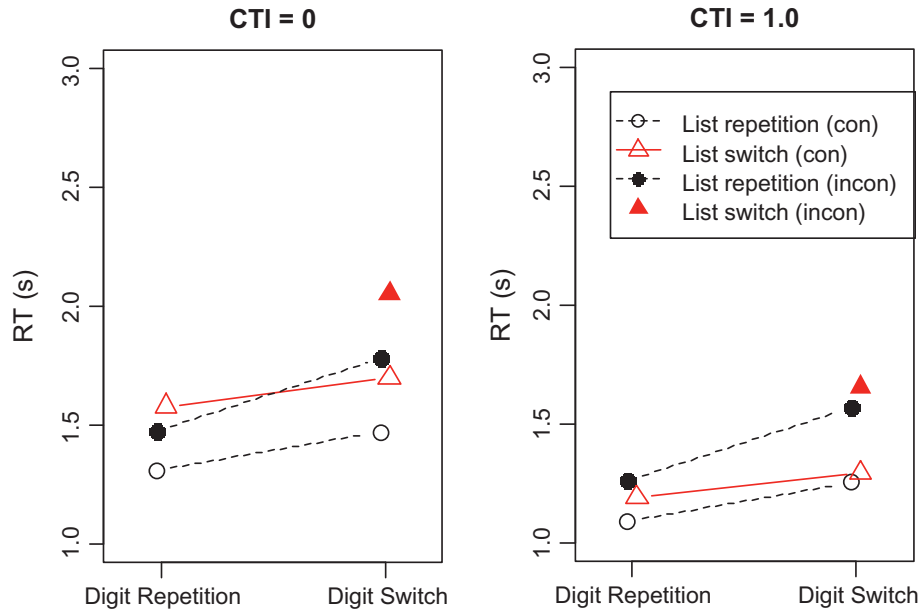

Fig. 12. Simulation results for an experimental design like that of Experiments 2 and 3. Effects of congruency, item repetition, and list repetition in the condition where the same item appears in the same position of the two lists. Labels correspond to the list-switching experiment (see Fig. 5); for comparison with data from the task switching experiment (Fig. 8), exchange "list" by "task" and "item" by "response".

Author's personal copy

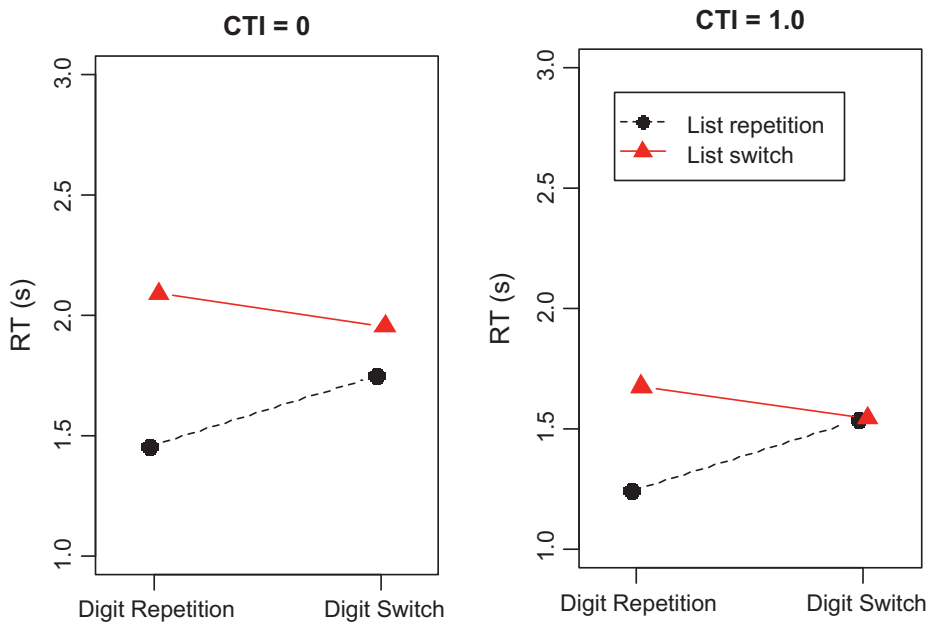

Fig. 13. Simulation results for the experimental design of Experiments 2 and 3. Effects of item repetition and list repetition in the condition where the same item appears in different positions of the two lists. Labels correspond to the list-switching experiment (see Fig. 6); for comparison with data from the task switching experiment (Fig. 9), exchange "list" by "task" and "item" by "response".

preparation interval; the list-switch cost is diminished but not entirely abolished at the long CTI. Fig. 12 shows simulation results for lists in which a digit is repeated in the same position in both lists (cf. Fig. 2A). Incongruent trials, involving positions occupied by different items in both lists, are generally slower than congruent trials involving positions occupied by the same digit in both lists. The congruency effect is relatively large for the two transition types involving a digit switch. Fig. 13 shows the simulation results for list compositions in which a digit is repeated in different positions across lists (cf. Fig. 2B). Repeated access to the same digit has a beneficial effect on list-repetition trials, but incurs a cost on list-switch trials. This effect is not diminished with a longer preparation interval. For the congruent trials in Fig. 12, the interaction of list repetition and digit repetition disappears.

One surprising finding from explorations of the model is that we could set the slow learning rate η_a to zero without impairing the model's ability to reproduce our data. Thus, the slow-changing association matrix $\mathbf{w_a}$ plays no role in accounting for the phenomena of our Experiments 1 and 2. In particular, the model reproduced the effect of list congruency (and of task congruency, as shown below) without long-term association learning. The congruency effect comes about because reconfiguration of the binding matrix in the direct-access region is never complete: It stops when the deviation between the binding matrix $\mathbf{w_b}$ and the output of the set-selection module $\mathbf{w_c}$ falls below the threshold β. At this point, some residual bindings from the previously used set remain in the binding matrix. They converge with the bindings of the currently used set in case of congruent items, but diverge from them, and therefore create interference, in case of incongruent items. The congruency effect arising from residual bindings is not diminished with a longer CTI because the reconfiguration of the binding matrix runs until criterion β is reached, regardless of CTI. The size of the congruency effect is controlled by β, which reflects the degree of interference from the currently irrelevant list that the system is ready to tolerate.

7.5. An application to task switching

When the model is interpreted as a model of procedural WM, the item-selection module becomes the response-selection module. Its input layer represents the stimuli and its output layer represents the responses. The set-selection module encodes task sets as chunks into LTM and retrieves them from LTM according to available task cues. With these role assignments, the model can be applied directly

Author's personal copy

to our Experiment 3. Because the design for Experiment 3 was closely parallel to that of Experiment 2, the model for Experiment 3 is identical to that for Experiment 2. With appropriate re-labeling, Figs. 12 and 13 can therefore directly be compared to the data from Experiment 3, shown in Figs. 8 and 9, respectively.

We noted above that in some regards our findings with both list switching (Experiments 1 and 2) and task switching (Experiment 3) deviated from what is found with the standard task-switching paradigm (e.g., Yehene & Meiran, 2007). In the standard task-switching paradigm, participants switch between two binary-choice tasks. Usually both tasks use the same set of responses, or at least the same response categories (e.g., index finger vs. middle finger of both hands; Druey & Hübner, 2008). Thus, the standard task-switching paradigm differs in several regards from the one we used here, where each task involved three response options, only one of which was shared between the two tasks. We therefore explored whether our model can also reproduce the effects of task repetition, response repetition, task-congruency, and preparation time in a standard task-switching experiment.

We modeled a task-switching paradigm with four stimuli, representing the four possible combinations of two binary feature dimensions (e.g., red and blue squares and triangles). Each task required classifying the stimuli by one dimension (e.g., color or form). Both tasks used the same two response categories (e.g., left vs. right key press). We created stimulus representations analogous to the position representations described above. That is, stimuli were patterns of activation over four units; each stimulus had a peak activation in one unit, and lower levels of activation in the other units, reflecting their similarity to other stimuli (e.g., a red square slightly co-activated the peak units for red triangles and for blue squares). Responses were represented in a localist fashion.

Each task set mapped two of the stimuli to one response, and the other two to the other response, each using one feature dimension as the classification criterion. Thus, the responses mapped to the four stimuli can be described in list format as [1 1 2 2] and [1 2 1 2] for the two tasks, respectively. As a consequence, two stimuli (the first and the fourth) are congruent, and the remaining two are incongruent.

We ran the model with two levels of CTI (0.1 and 1 s), using the same parameter values as for the simulations of list switching reported above. The results, presented in Fig. 14, show the well-established effects from the standard task-switch paradigm: The task-switch cost was substantially diminished at the long CTI, and actually disappeared altogether for the congruent trials. Response repetition

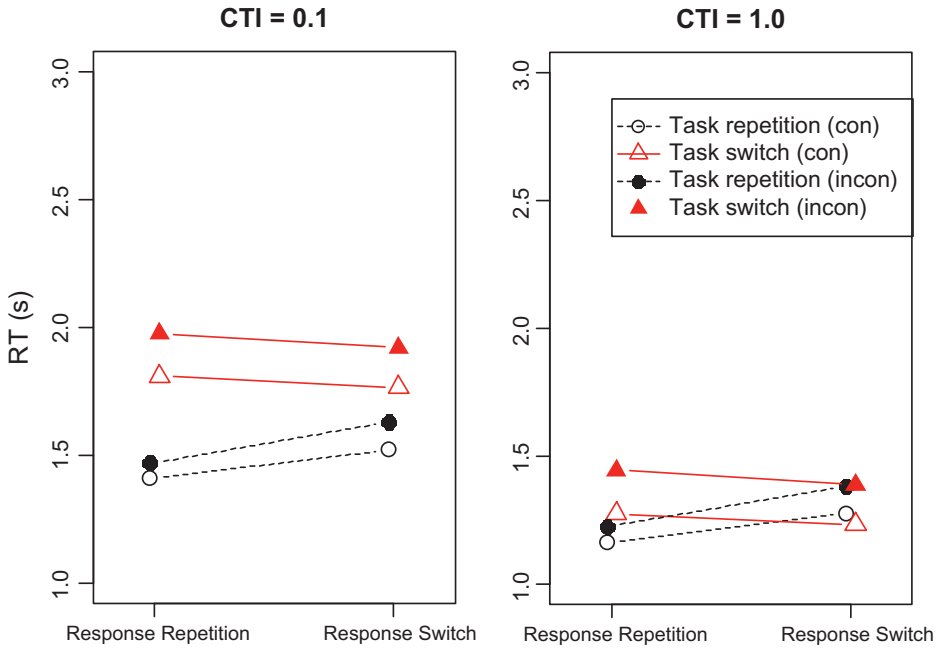

Fig. 14. Simulation results for a standard task-switching experiment: Effects of congruency, response repetition, and task repetition for two levels of CTI.

Author's personal copy

led to faster responses when the task was repeated, but slowed down RTs when the task was switched. In addition, there was a substantial effect of congruency, which was not diminished at the longer CTI.

We were particularly interested in the two interactions where our data from Experiment 3 deviated from those of Yehene and Meiran (2007). In the simulation of the standard task-switching paradigm the interaction of response repetition with task repetition was equally strong for congruent and for incongruent trials. Thus, the present simulation deviates from our simulation of Experiments 2 and 3, but is in complete agreement with Yehene and Meiran, who found no reduction of that interaction for congruent trials. We explain in Appendix B why the model behaves differently in the two simulations.

The interaction of response repetition and task repetition also remained undiminished with a longer CTI. In this regard, the present simulation agrees with the simulation of Experiments 2 and 3, and not with the results of Yehene and Meiran. This deviation from the data might point to a limitation of the model. It is not clear, however, whether the result of Yehene and Meiran generalizes to all variants of the task-switch paradigm. Mayr and Kliegl (2003) observed an interaction of response repetition with task repetition that remained undiminished after a long preparation interval, in agreement with our simulation.

7.6. Testing simplified model versions

In our model, three mechanisms are assumed to jointly produce the interaction of item repetition with list repetition: input priming (i.e., priming of the item's cue, such as its position, in the input layer), and strengthening of bindings during task execution contribute to the item-repetition benefit when the list is repeated; response suppression generates the item-repetition cost when the list is switched. Here we investigate whether each of these mechanisms is necessary by testing three simplified versions of the model, each of which eliminates one mechanism.

In the first simplified version, we set candidate activation parameter α to zero, thereby effectively eliminating response suppression (because there is nothing left to suppress); the set of parameter values can be found in the column *Reduced 1* in Table 6. This model reproduced all the findings of Experiments 1 and 2 with the exception of the item-repetition cost after a list switch. The model also did not produce the response-repetition cost in task-switch trials when applied to the standard task-switch paradigm. We conclude that response suppression is necessary for our model to predict item-repetition and response-repetition costs.

In the second simplified version, we set the weight strengthening parameter f to zero, thus eliminating any strengthening of the binding between the selected item and its retrieval cue (or between the selected response and the stimulus) during task execution; the set of parameter values can be found in the column *Reduced 2* in Table 6. The model was still able to reproduce most of the results of Experiments 1 to 3, with one exception: The simulation generated no interaction between congruency and type of transition. This interaction was clearly present in all three experiments: The congruency effect was reduced for item-repetition trials compared to item-switch trials (see top panel of Fig. 4 for Experiment 1, and Fig. 5 for Experiment 2). Likewise, the congruency effect was reduced for stimulus-repetition trials compared to stimulus-switch trials (see Fig. 8 for Experiment 3). The present simulation result illuminates the reason why the full model successfully produced this interaction: In the full model, the benefits of item repetition and of congruency are under-additive because they arise in part from the same source, strength of the cue-item bindings. Strengthening of bindings by the delta rule is not additive; this rule rather adjusts bindings to minimize prediction error. Bindings that are already strong, such as those for congruent cue-item links, are hardly strengthened further because that would lead to over-prediction of the current item activation in the output layer. This generates the under-additive effect of item-repetition with congruency in the full model. In the reduced model, the item-repetition benefit arises entirely from cue priming, and the congruency effect arises entirely from stronger cue-item bindings for congruent than for incongruent items, and therefore, the effects of these two mechanisms are additive.

The third reduced model eliminated cue priming and stimulus priming by setting γ, the squashing parameter for the input layer, to zero (for parameter values see column *Reduced 3* in Table 6). This reduced model is very successful in reproducing the findings in our experiments. The only point of

Author's personal copy

deviation is that it predicts no effect of congruency for item-repetition trials in list switching, and for response-repetition trials in task switching. In the data, the congruency effect was reduced for item-repetition trials but remained significant in Experiment 1, and likewise, in Experiment 3 the congruency effect remained significant for response-repetition trials. The problem of the third reduced model is the reverse of that of the second reduced model: It produces an under-additive interaction of congruency and item repetition that is too large, because it generates both effects exclusively through the strength of bindings.

We tentatively conclude that the full model gives a better account of the present data than any of the reduced model versions. Therefore, we argue that both cue priming and strengthening of cue-item bindings are jointly responsible for item-repetition benefits when the list is repeated, whereas the item-repetition cost on list-switch trials arises from response suppression. Likewise, the response-repetition benefit on task-repetition trials arises from the joint effects of stimulus priming and strengthening of stimulus–response bindings, and the response-repetition cost on task-switch trials arises from response suppression. We add, however, that the reduced model versions 2 and 3 also fare very well – their failures to account for the present data are too small to rule them out conclusively. In the remainder of this article we test three further predictions from the full model, which will help reducing the number of empirically viable model variants.

8. New model predictions

We used the full model to generate three new predictions, two of which we tested by additional experiments, and one that we tested in the data from all experiments combined.

The first prediction arises from the novel insight that the congruency effect in our experiments can be explained entirely through rapidly-changing bindings, without any contribution from slowly learned associations in LTM. One implication of this discovery is that a congruency effect should arise in a situation in which long-term learning of position-item associations is held constant for all items (This was not the case in Experiment 1 to 3 in which congruent position-item bindings were accessed more often and therefore better learned than incongruent position-item bindings). We tested this prediction in Experiment 4.

The second prediction was motivated by a natural extension of our investigation of item-repetition effects. So far, we only examined list compositions with an item repeated across two different lists. Here we ask what would happen if the same item appeared twice in different positions of the *same* list. Intuitively one might expect an item-repetition cost when the same item is accessed on two successive trials, cued by different positions, because the repeated item would be suppressed, and it would not appear to benefit from cue priming or from strengthening of cue-item bindings. Against this intuition are findings of item-repetition benefits when the same memory item was accessed through different cues. In experiments assigning the same item to two different cues, an item-repetition benefit was found when the same item was accessed on two successive trials, even when cued by a different list position (Oberauer, 2003) or by a different symbol (Gehring, Bryck, Jonides, Albin, & Badre, 2003). These findings contrast with the ones in our Experiments 1 and 2, where we found item-repetition costs in a condition where the same item was accessed in a different list position. The only difference between our experiments and those of Oberauer (2003) and Gehring et al. (2003) is that here we investigated item-repetition effects following a list switch, whereas in the previous studies two instances of the same item were repeated within the same list. We therefore designed Experiment 5 to directly compare item-repetitions in different list positions in different lists (as in Experiments 1 and 2) to item-repetitions in different positions in the same list (as in Oberauer (2003) and Gehring et al. (2003)). When we ran the model on this design, it generated an item-repetition cost for repetitions in different lists, and an item-repetition benefit for repetitions within the same list. Below we explain what causes this somewhat counterintuitive result in the model, and present an experiment testing the prediction.

The third prediction arises directly from the assumption in the model that item-repetition benefits arise in part from cue priming, and response-repetition benefits arise in part from stimulus priming. If this is the case, then we should find an RT benefit for pure repetitions of the position (i.e., list switch,

Author's personal copy

position repetition, and item switch) over trials without position repetition (i.e., list switch, position switch, and item switch). Likewise, we should find a pure stimulus-repetition benefit in task switching. Our experiments included such conditions, and below we present an analysis across all five experiments providing direct evidence for cue priming and stimulus priming.

9. Experiment 4: Short-term list-congruency effect

We designed Experiment 4 to test the prediction that a list-congruency effect should be observed even when long-term learning of position-item associations is held constant. In the preceding experiments the congruent position-item associations were used, and thereby learned, more frequently than the incongruent position-item associations. In Experiment 4 we created a situation in which all position-item associations are used with equal frequency throughout the experiment, so that their strength of association in LTM should be equal. This was accomplished by having participants learn four lists with the following structure:

- List 1: [*A B C*]
- List 2: [*A D E*]
- List 3: [*F B E*]
- List 4: [*F D C*].

Across the four lists, every item occurs exactly twice, both times in the same list position. Because all four lists are learned and used equally often, every position-item association is learned to the same degree. For each run of 13 arithmetic operations, two of the four lists were selected. Within every pair of lists, there is one item that occurs in the same position in both lists. Trials requiring access to this item in this position are regarded as congruent in the given run. For instance, if lists 2 and 4 are selected, access to item D in the middle position constitutes a congruent trial. Note that when list 2 is paired with any other list (e.g., list 1), access to item D in the middle position of list 2 would be an incongruent trial, because the middle position is occupied by a different item in the other list. Thus, congruency is defined by the subset of lists relevant in the current run. Across all runs, every item in every list is equally often regarded as congruent, depending on the pairing of that list with another list. If the congruency benefit arises entirely from the strength of position-item association in LTM acquired gradually over the whole experiment, there should be no congruency effect in this experiment. In contrast, if part of the congruency benefit arises from rapidly changing position-item bindings in those two lists that are relevant in the current run, then we should find a congruency benefit in Experiment 4.

9.1. Method

9.1.1. Participants

Twenty students from University of Zurich took part in return for either partial course credit or a payment of 15 CHF.

9.1.2. Materials and procedure

The materials and procedure were the same as in Experiments 1 and 2, with the following exceptions: Participants learned four lists, composed according to the schema presented above. For each participant, a random set of 6 digits was assigned to the letters of the schema. The four lists were distinguished by four colors (red, blue, green, and orange). Participants first completed 18 practice runs with six arithmetic operations each; the number of operations was reduced in the practice runs to ensure a rapid turnover of different pairs of lists across different runs. Practice was followed by 36 test runs with 13 operations each. Across each set of 6 successive runs, each of the six pairwise combinations of two lists was used once, in a random order.

Author's personal copy

9.2. Results and discussion

Overall accuracy was .90 (*SD* = .10). Response times were treated as in the preceding experiments (2.0% of RTs were eliminated as outliers). Mean RTs in the design cells are given in Table 7. We analyzed mean RTs by an ANOVA with Congruency and Transition Type (list repetition with position and digit repetition, list repetition with position and digit switch, and list switch with position and digit switch) as independent variables. The congruency benefit was significant, $F(1, 19) = 5.4$, partial $\eta^2 = .22$, $p = .03$. Mean RTs were 2.23 s (SE = 0.12) on congruent trials, and 2.40 s (SE = 0.15) on incongruent trials. There was a main effect of transition type, $F(1, 19) = 38.7$, partial $\eta^2 = .67$, $p < .001$, reflecting the same pattern as in the preceding experiments (see Fig. 15): Responses were fastest for repetitions of both list and digit, intermediate for repetition of list with a switch to another digit, and slowest for list switches. Contrary to Experiments 1 and 2, congruency did not interact with transition type ($F < 1$).

The finding of a congruency benefit in the present experiment shows that the congruency effect does not only reflect the degree of long-term learning of position-item associations. In the present experiment, the current task context (i.e., the lists chosen for the current run) determined which position-item link was regarded as congruent. The congruency effect observed here must be attributed to the bindings established in the direct-access region during the current run. As discussed above, in the model the congruency effect arises from the residual bindings of the not-used list that remain in the binding matrix $\mathbf{w_b}$. During each run, the bindings of the two lists used for that run are established in random alternation in the item-selection module by updating the binding matrix. With every list switch, residual bindings of the previous list remain in the binding matrix; these residual bindings cause a congruency benefit. The remaining two lists are not used during the entire run, and any residuals of their bindings from the previous run are quickly washed out by the rapid updating of bindings in the item-selection module.

10. Experiment 5: Item repetition in different positions of the same list

In Experiment 5 we directly compared two kinds of digit repetitions: repeated access to the same digit in a different position in a *different* list, and repeated access to the same digit in a different position of the *same* list. The first case incurred a digit-repetition cost in Experiments 1 and 2; the second case yielded a digit-repetition benefit in Oberauer (2003) and Gehring et al. (2003). Both these effects are reproduced by our model; the predictions are presented in the left panel of Fig. 17.

The opposing predictions for these two situations are surprising. The only thing that differs between the two conditions is that in the first case, where the same item is accessed again in a different list, involves a list switch, whereas the second case involves a list repetition. An analysis of the model behaviour revealed why, when the same list is used again, repeated access to the same digit in a different position leads to a digit-repetition benefit (for a schematic illustration see Fig. 16). The source of this benefit is cue priming (in our case, position priming). Imagine a list with a repeated digit, such as [7 4 7]. On trial $n - 1$, the third position is probed, and the model retrieves the 7 bound to it. After completion of the trial, the binding of 7 to the third position is strengthened, the selected item 7 is suppressed in the candidate layer, and a small proportion of the activation of the third position in

Table 7
Mean response times in seconds (standard deviations in parentheses) by list repetition, position repetition, and digit repetition in Experiment 4.

Congruency	Digit repetition condition	List Repetition		List Switch	
		Position Repetition	Position Switch	Position Repetition	Position Switch
Incongruent	Digit repetition	**2.03 (0.57)**			
	Digit switch		**2.50 (0.72)**	2.44 (0.68)	**2.72 (0.77)**
Congruent	Digit repetition	**1.92 (0.56)**		**2.38 (0.58)**	
	Digit switch		**2.23 (0.58)**		**2.55 (0.65)**

Note: Means plotted in Fig. 15 are printed in bold.

Author's personal copy

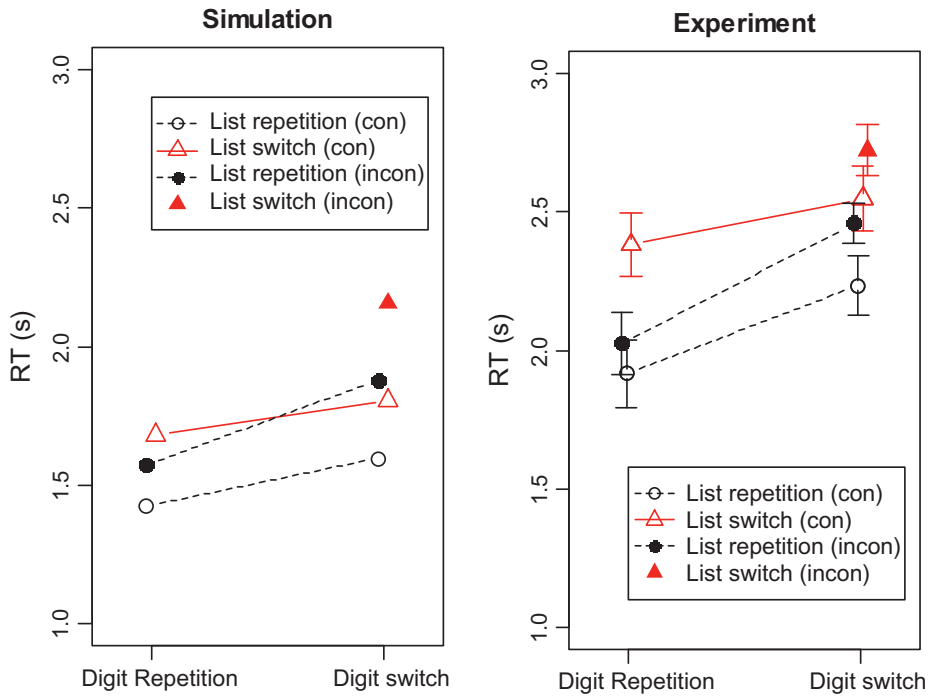

Fig. 15. Simulation results (left) and experimental data (right) for Experiment 4: Effects of congruency, item repetition, and list repetition.

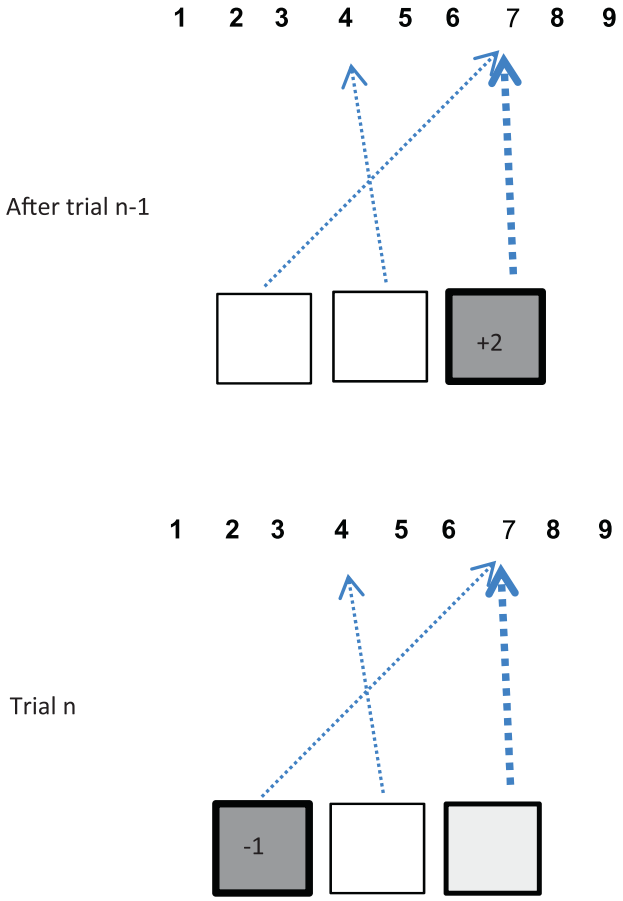

Fig. 16. Illustration of the model predictions for item repetitions in different positions in the same list. On trial $n-1$, the digit "7" was accessed from the third position, leaving the representation of that digit suppressed, the third position primed, and the binding between the third position and "7" strengthened (top panel). On trial n, the same digit is accessed from the first position (bottom panel). Activation of "7" from the first position is supported by residual activation from the third position, transmitted through a strengthened binding. Together, these two sources of activation overcome the suppression of the digit representation.

Author's personal copy

the input layer carries over into trial *n*. On trial *n*, the first position is probed. Item 7 in the output layer now receives activation from the highly active first position, mediated through the binding from the first position to 7, and in addition from the still activated third position, mediated through the (strengthened) binding from the third position to 7. Thus, in comparison to a trial *n* accessing a new digit (e.g., the 4 in the second position), accessing the same digit again in a different position is assisted by the fact that the residual activation of the previous position acts as an additional cue to the correct item. This additional cueing more than compensates the suppression of the item in the candidate layer.

Note that additional cueing from the previous position is helpful only when the list remains the same from trial $n-1$ to trial *n*. If the list is switched, the bindings that mediate between the previous position and the repeated item are no longer in place, and therefore no additional beneficial cueing occurs. Therefore, no item-repetition benefit is observed when the same item is accessed again in a new position after a list switch (as shown in Experiments 1 and 2). To conclude, the predicted item-repetition benefit for different positions within the same list arises from cue priming. We verified this analysis by setting the input priming parameter γ to zero, as in model version *Reduced 3*, upon which the model no longer predicts the item-repetition benefit within a list.

10.1. Method

Participants were 25 students from University of Zurich, who took part for partial course credit or CHF 15. Three further participants with error rates > 30% were excluded from analysis.

The method was identical to that of Experiment 1, with the exception that the no-overlap list composition was replaced by a pair of lists in which a digit repeated within one of the two lists, e.g., [7 4 7], [8 5 2]. The two identical digits were placed in two randomly selected positions for each participant. The other two list compositions were the same as in Experiment 1: repeating a digit in a different position across the two lists, e.g., [7 4 5], [8 5 2], and repeating a digit in the same position in both lists, e.g., [7 4 5], [8 2 5]. The last composition is of no interest in the present context (i.e. it just replicated the congruency effect of Experiments 1 and 2), and we therefore limit our report to the first two compositions.

10.2. Results and discussion

Accuracy was again high ($M = 0.97$, $SD = .02$). RTs from erroneous trials, as well as outliers, defined as in the preceding experiments (2.0% of RTs), were removed. The mean RTs per condition are summarized in Table 8. Fig. 17 plots the means for digit repetitions and digit switches for list-repetition and list-switch trials. The left panel presents the model predictions and the right panel presents the experimental data.

The list-switch condition replicated the findings from Experiments 1 and 2: When the list and the position was switched, repeated access to the same digit incurred a cost of 121 ms. In contrast, when the list was repeated, repeated access to the same digit in a different position of that list was 280 ms

Table 8
Mean response times (standard deviations in parentheses) by list repetition, position repetition, and digit repetition condition in Experiment 5.

List composition	Digit repetition condition	List Repetition		List Switch	
		Position Repetition	Position Switch	Position Repetition	Position Switch
Different position, same list	Digit repetition	1.75 (0.52)	**1.67 (0.53)**		
	Digit switch		**1.95 (0.40)**	2.11 (0.56)	2.18 (0.50)
Different position, different list	Digit repetition	1.73 (0.53)			**2.56 (0.68)**
	Digit switch		2.18 (0.50)	2.35 (0.59)	**2.44 (0.49)**

Note: Means displayed in Fig. 17 are printed in bold.

Author's personal copy

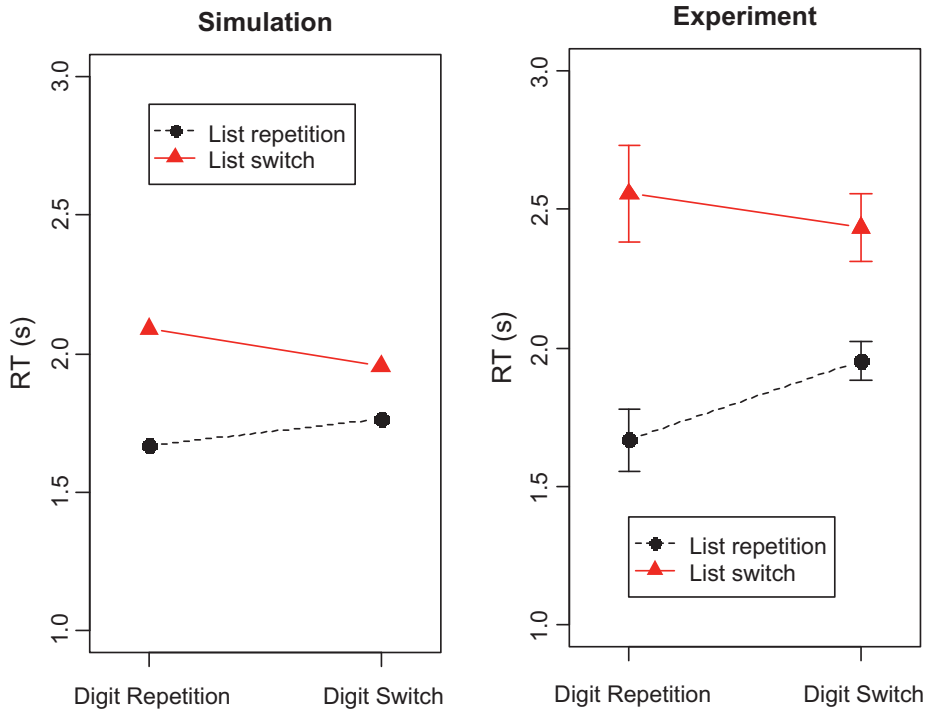

Fig. 17. Simulation results (left) and experimental data (right) for Experiment 5: Effects of digit repetition for all combinations of list repetition and position repetition.

faster than access to a different digit in a different position of the same list. This pattern was reflected in the interaction of list repetition with digit repetition, $F(1, 23) = 15.83$, partial $\eta^2 = .40$, $p = .001$. The new prediction of a digit-repetition benefit in the list-repetition condition was confirmed in a post hoc test, $t(24) = 4.6$, $p < .001$. The only significant effect in ER was the interaction of list repetition and digit repetition, $F(1, 23) = 4.3$, partial $\eta^2 = .15$, $p = .048$, which went in the same direction as that in the RT data.

As predicted by the model simulation, digit repetition in a different position within the same list resulted in a repetition benefit. This prediction arises from the assumption that residual activation of the position representation in one trial carries over into the next trial. The confirmation of this prediction therefore supports the assumption of cue priming (here: priming of the position). One implication of our model analysis is that the finding of an item-repetition effect when the cue is switched (Gehring et al., 2003; Oberauer, 2003) does not rule out that the item-repetition benefit arises purely from priming of the cue, as represented by model version *Reduced 2*. In the next section we provide more diagnostic evidence against this possibility.

11. Repetition priming of item cues and target stimuli

Our third new model prediction arises directly from the assumption that representations in the input layer (i.e., item cues in declarative WM and target stimuli in procedural WM) carry over some of their activation into the next trial: Repeated use of the same position cue by itself should lead to faster RTs compared to a position switch. By analogy, repetitions of the same stimulus in a task-switch experiment should lead to an RT benefit even when nothing else is repeated.

The pure effect of position repetition in the list-switch experiments (1, 2, 4, and 5) in the absence of repetition of any other feature can be evaluated by looking at the list-switch columns in Tables 3, 4, 7 and 8, comparing position repetition to position switches in the digit-switch rows. Analogously, the pure effect of stimulus repetition in the task-switch experiment (Experiment 3) can be found in Table 5 in the task-switch column, response-switch row, stimulus repetition versus stimulus switch. In the simulated data this comparison yielded an advantage of 135 ms for input repetitions (i.e., repetitions

Author's personal copy

of item cues or target stimuli) over input switches; this advantage disappeared in the *Reduced 3* version in which the input priming parameter γ was set to zero.

Across the five experiments there were 16 such differences, all of which were positive, with a mean of 133 ms (*SD* = 78, range 30–290 ms). We interpret this effect as resulting from priming of the item cue (i.e., priming of the position in our list-switch experiments) and priming of the target stimulus (i.e., the symbol in our task-switch experiment). The input-priming effect is consistent with previous research showing a contribution of task-cue priming to task-switching effects (Logan & Bundesen, 2003; Mayr & Kliegl, 2003) and of object-cue priming to object-switch effects (Li et al., 2006). It also converges with a recent demonstration in our lab that repetition of the stimulus category leads to an RT benefit even after a task switch (Druey, 2011).

Priming of the item cue likely contributes to item-repetition benefits in comparisons that confound repetition of the item with repetition of the cue. This confound is usually present when item-repetition effects are evaluated within a single list. It is therefore conceivable that the item-repetition benefit in experiments investigating selection of items in declarative WM (e.g., Oberauer, 2003; Verhaeghen & Basak, 2005) is nothing but an effect of cue priming. Likewise, the response-repetition benefit in experiments on response selection might be nothing but an effect of priming the corresponding stimulus category. As our Experiment 5 and the accompanying simulation has shown, this possibility cannot be ruled out by using different cues to the same memory item, or different stimuli mapped to the same response.

One reason why we do not endorse this simplified view is that our second reduced model, in which we set strengthening of bindings to zero, gave a slightly less satisfying account of the present data than the full model. Another reason is that the pure position-repetition (or stimulus-repetition) effect evaluated above is too small to explain the benefit of repeating an item together with its position (or a stimulus together with its response). For comparison, we evaluated the classical item-repetition benefit in list-repetition trials, and the classical response-repetition benefit in task-repetition trials, across the five experiments. The relevant comparison is found in the data tables in the list-repetition columns, comparing the digit-repetition with position-repetition cell to the digit-change with position-change cell, excluding congruent trials. The equivalent cells in the task-repetition column of Table 5 are stimulus repetition with response repetition, and stimulus switch with response change. The classic repetition benefit (i.e., the joint effect of repeating item and position, or stimulus and response) amounted to 336 ms across the 16 comparisons (*SD* = 142, range 80–570 ms). For all 16 cases, the classic repetition benefit of jointly repeating item and position was larger than the pure position-repetition benefit evaluated above. This result shows that repeating the conjunction of item and position cue (or of stimulus and response) adds a benefit over and above just repeating the position cue (or just repeating the stimulus). The simulated data from the full model agree with this result, showing a classical item-repetition benefit of 299 ms (more than twice as much as the pure cue-repetition benefit). The second reduced model, which eliminates strengthening of bindings and leaves only cue priming in place, generated a classical item-repetition benefit of only 78 ms (even less than the pure cue-repetition benefit).

We conclude that the comparison between the classical item-repetition benefit and the pure cue-repetition benefit provides evidence for our assumption that, after each trial, the binding between the selected item and its cue is strengthened. Hence, the classical item-repetition benefit, first described by Garavan (1998), has two sources, cue priming and strengthening of item-cue bindings. Likewise, the classical response-repetition effect (Bertelson, 1965) arises from priming of the stimulus (or the stimulus category) and strengthening of stimulus–response bindings.

12. Discussion of model

12.1. Basic model assumptions

Our model incorporates two main assumptions about WM that distinguishes it from other models of WM and action control. First, we provide an explicit mechanism for chunking of memory sets and of task sets. This model feature is motivated by converging evidence for a mechanism that forms unified

Author's personal copy

representations of structures. Studies of immediate serial recall, for instance, have shown that the improvement of immediate recall of frequently occurring lists (i.e., the so-called "Hebb effect") arises from the acquisition of unified representations of memory lists that are strengthened through repetition (Burgess & Hitch, 2006). Incidental learning of regularities in sequences governed by artificial grammars has also been modeled most successfully through the assumption that frequently repeated sub-sequences are acquired as chunks (French, Addyman, & Mareschal, 2011; Perruchet & Vinter, 1998). The costs of switching between tasks, and of switching between lists, provide further evidence that tasks and lists are represented in a unified fashion, so that they can and must be exchanged as a whole. Most prior models of task switching assume unified representations of tasks (e.g., Brown et al., 2007; Gilbert & Shallice, 2002; Meiran, 2000; for an exception see Schneider & Logan, 2009); here we present a mechanism for how such unified representations can be acquired.

The work of Dreisbach and colleagues directly demonstrates the unified nature of task sets by comparing a condition in which participants learned a set of eight stimulus–response mappings either as a single, unstructured set or as two classification tasks, each of which covered four S–R mappings (Dreisbach, Goschke, & Haider, 2006, 2007). A task-switch cost was observed only when the S–R mappings were instructed as belonging to different task sets. In our model, an unstructured set of S–R mappings would be learned as a single set of bindings in $\mathbf{w_b}$, such that no updating of the binding matrix is necessary, and hence, no switch cost would emerge. In contrast, when the same set of S–R mappings is instructed as belonging to two tasks, then each sub-set is acquired as a separate set of bindings in $\mathbf{w_b}$, and each set is associated to a task cue in the set-selection module. As a consequence, switching between the two tasks implies an update of the matrix of S–R bindings, which translates into a switch cost.

This leads us to the second core assumption in our model: We developed a mechanism that recodes bindings between activation patterns in one module (i.e., the item-selection or response-selection module) into activation patterns in another module (i.e., the set-selection module). This is made possible by representing bindings by gain-modulating units with varying activation levels, rather than by connections with varying strengths. In this way, the model can translate between two representations of structure: In the item-selection module, structures such as lists and task sets are represented by temporary bindings between elements (e.g., bindings between positions and items, or between stimuli and responses). In the set-selection module, the same structures are represented as elementary representations (i.e., activation patterns in the set layer), which themselves are bound to other elements (i.e., activation patterns in the cue layer). Therefore, we can use the same mechanisms for selecting entire sets and for selecting elements within those sets. We can also use the same principles of learning for acquisition of links between sets and their cues (e.g., between task cues and tasks), and for acquisition of links between elements and their cues within sets (e.g., between stimuli and responses).

This architecture differs from existing models of task selection and task switching, in which stimulus–response links, as well as links between task cues and tasks, are not learned and therefore must be coded by hand (e.g., Brown et al., 2007; Gilbert & Shallice, 2002; Meiran, Kessler, & Adi-Japha, 2008). The ability to learn new task sets and new memory sets was borrowed from models of serial-order memory (e.g., Burgess & Hitch, 2006; Henson, 1998b; Lewandowsky & Farrell, 2008), which encode new lists by rapid Hebbian learning of associations between list positions and items. Those models, however, have never addressed the problem of switching between different lists, and therefore, with one exception (Burgess & Hitch, 2006), have not included unified representations of lists that can be associated to list cues. Our systematic juxtaposition of phenomena in declarative and procedural WM has revealed that the research traditions in the two fields have highlighted complementary aspects of the flexibility of our WM system: The rapid learning of new sets has been studied in research on declarative WM, whereas the rapid switching between already learned sets has been investigated in research on procedural WM. Here we propose a connectionist architecture that accomplishes both, thereby offering an explanation of as yet neglected phenomena: how new task sets can be learned from feedback, or implemented from instructions, and how people can switch between different memory sets in declarative WM.

Our model is among the first to offer a computational explication of the relation between WM and LTM in a connectionist framework, following the lead of Burgess and Hitch (2005, 2006). A crucial as-

Author's personal copy

pect of this relation is the translation between structural representations in WM, and unified representations of these structures as chunks in LTM. Structures in WM are encoded into LTM by chunking them, and these chunks are unpacked again when they are retrieved back into WM (Oberauer, 2009). Our mechanism for translating between chunked and unpacked representations of structures can help to explain how we construct hierarchical memory structures such as embedded episodes (Farrell, 2012) and plans (Miller, Galanter, & Pribram, 1960; Schneider & Logan, 2007).

The architecture of our model has some resemblance to the ACT-R architecture (Anderson, 2007): Like our model, ACT-R distinguishes declarative and procedural representations. ACT-R consists of several modules, each with its own buffer that is limited to a single chunk. A chunk is a structure of representational elements (e.g., the fact that 3 + 4 equals 7). As such, the buffers serve a similar function as the central components of WM in our model (i.e., the direct-access region and the bridge). ACT-R has two buffers for holding declarative representations, the *imaginal buffer* holding intermediate steps of manipulation of declarative representations, and the *declarative buffer* holding representations retrieved from declarative LTM. In our model there is no distinction between declarative representations retrieved from LTM and those generated through manipulation, both are held in the region of direct access.

A perhaps more important difference between our model and ACT-R is that the procedural module in ACT-R has no buffer of its own. Thus, the basic unit of procedural memory in ACT-R, the production rule, is not held in a buffer. Rather, procedures are selected on a moment-by-moment basis according to their strength in LTM and their match to contents of other buffers to which they can be applied. Once executed, a production falls back to its status prior to selection. Therefore, the inertia of task sets, as reflected in the task-switch cost, cannot be explained in ACT-R by assuming that the just-executed production still resides in some procedural buffer. Instead, models of task switching built within the ACT-R architecture assume that task sets are declarative chunks, and the task-switch cost arises because the most recent representations in episodic memory are the easiest to retrieve (Altmann & Gray, 2008). In contrast, in our model, a task set is a procedural representation that, when held in the bridge of procedural WM, acts as a "prepared reflex" that automatically executes as soon as it receives a matching input. In ACT-R, it is possible to learn new, highly specific productions through practice, so that it should be possible to acquire productions implementing task sets. Thus, it appears that, after prolonged practice with two (or more) tasks, ACT-R should predict that these tasks are implemented as productions (each matching to the conjunction of a task cue and a target stimulus), and thereby, the switch cost should disappear. The list-switch cost, however, should not disappear with practice, because memory lists remain declarative representations. In contrast, our model predicts that neither task-switch nor list-switch costs disappear with practice. If our predictions from ACT-R can be verified through simulations, the fate of list-switch and task-switch costs after practice would be diagnostic for empirically adjudicating between our model, which posits analogous structures of declarative and procedural WM, and the ACT-R architecture, which does not incorporate this analogy.

At the present state of development, we do not aspire for our model to explain all well-established phenomena in the literature on short-term memory for lists and in the literature on task and response selection. Rather, our aim was to provide a coherent account for the findings in our experiments, and analogous findings in the literature. These findings form three groups: (1) the time costs of switching between sets as a function of preparation interval, (2) the effects of list congruency and task congruency, and (3) effects of repeated selection of the same memory item or the same response. In the introduction, we tentatively interpreted set-switching costs as reflecting the updating of the direct-access region and the bridge, congruency effects as arising from activated LTM, and item-repetition/response-repetition effects as reflecting the shifting of the foci in declarative and procedural WM, respectively. Our modeling results show that these attributions must be partially revised, as will become clear in the following sections.

12.2. List switching and task switching

In the literature on task switching, switch costs have been attributed to two sources: proactive interference from the task set used in the preceding trial (Allport et al., 1994), and the time for

Author's personal copy

reconfiguring the system for the current trial (Rogers & Monsell, 1995). Our model implements both: On each trial the relevant task set is retrieved from the set-selection module into the item-selection (or response-selection) module, reconfiguring the binding matrix, and this reconfiguration takes time. The cost of switching to a new list or task largely reflects this time. Reconfiguration is necessary to overcome proactive interference from the previous trial, and the amount of reconfiguration needed depends on the strength of proactive interference: The more the state of the binding matrix $\mathbf{w_b}$ after tial $n-1$ differs from the state needed to carry out trial n, the longer it takes for reconfiguration to change $\mathbf{w_b}$ until it is in sufficient agreement with the specification of the new set retrieved from LTM, $\mathbf{w_c}$, so that their difference falls below the criterion β.

We assume that the preparation time available at a long CTI is used for reconfiguration. With a sufficiently long preparation interval, the criterion β is reached before the end of the CTI, and as a consequence, list-switch and task-switch costs are eliminated entirely for most kinds of trials. There is one exception, however: item-repetition trials are slowed after a list switch relative to a list repetition (at least for incongruent trials, see Fig. 12). Likewise, in the standard task-switching paradigm response-repetition trials are slowed after a task switch relative to a task repetition (see Fig. 14). When these trials are included in the analysis of set-switch costs (as they usually are), our model predicts a residual switch cost even at long preparation intervals. One reason why we did not obtain a residual list-switch cost in Experiment 2 is that we excluded these trials from analysis. Of course, further residual switch costs will arise to the degree that people fail to use the preparation interval for reconfiguration of the binding matrix (De Jong, 2000). To incorporate the possibility of failure to engage in advance reconfiguration, the model could include a parameter for the probability of initiating reconfiguration of the binding matrix in the preparation interval.

12.3. List congruency and task congruency

One discovery enabled by our computational model was that the congruency effects in our experiments, and in typical task-switch experiments, can be explained by the residual bindings of the currently irrelevant list or task. Theoretical considerations of whether the task-congruency effect arises from representations of the not-selected task in WM or in LTM (Meiran & Kessler, 2008; Yamaguchi & Proctor, 2011) have assumed that longer preparation intervals result in more complete removal of the old task from WM. On this premise, one would assume that with sufficient preparation interval the not-used task set is removed from WM, so that congruency effects should disappear. In our model, congruency effects don't diminish over a preparation interval because removal of the irrelevant task set (or memory set) proceeds until the not-used set has been weakened sufficiently, regardless of the available preparation interval. Therefore, the interaction of congruency with preparation interval is not diagnostic for the source of congruency effects.

Other evidence speaking to the role of LTM in generating congruency effects comes from studies investigating the effect of long-term experience on the size of the congruency effect. For instance, Kiesel et al. (2007) asked participants to switch between an odd–even and a large-small classification of digits. In one of the tasks, only a subset of the digits was presented (e.g., only the digits 2, 3, 8, and 9 were used in the large-small task). When the digits omitted in one task were presented in the other task (e.g., the digits 1 or 4 presented in the odd–even task), they still generated a task-congruency effect, but the congruency effect was larger for digits that were used in both tasks. This latter finding suggests that long-term learning of stimulus–response associations for digits used in both tasks contributed to the congruency effect.

To explore the limits of our model's ability to explain such long-term congruency effects we ran the model on the design of Kiesel et al. (2007). The model generated a congruency effect for stimuli that were used in only one task (e.g., digits 1 and 4). This congruency effect arises because when a task set is implemented in the bridge (i.e., the response-selection module), all its bindings are implemented, including those linking the unused stimuli to their responses. When the task is later removed during a task switch, residuals of these bindings remain, creating a congruency effect. Contrary to the findings of Kiesel et al., however, the simulated congruency effect was only negligibly larger for digits used in both tasks than for digits not used in one of the tasks. Apparently, in the model the congruency effect arises mostly from the implementation of stimulus–response bindings when a task set is retrieved

Author's personal copy

into the bridge, whereas the short-term strengthening of stimulus–response bindings used during task execution does not contribute much to the effect. When we increased the slow learning rate for stimulus–response associations, η_a, from 0 to 0.02, the congruency effect for digits used in both tasks increased relative to the congruency effect for digits used in only one task, bringing the model into better agreement with the data of Kiesel et al. (2007). We conclude that the modulation of the task-congruency effect by frequency of use of stimulus–response mappings observed by Kiesel et al. is evidence for a contribution of long-term association learning to the congruency effect.

Further evidence for a role of long-term learning of individual stimulus–response associations was obtained by Mayr and Bryck (2005), who found that RTs decreased for individual stimulus–response mappings practiced frequently, relative to those practiced with lower frequency. Moreover, Horner and Henson (2009) found that stimulus-priming effects in speeded choice tasks can in large part be attributed to long-term strengthening of stimulus–response associations on several levels of representation of stimuli and responses.

It is for these reasons that we assumed slow-changing association weights $\mathbf{w_a}$, in addition to the fast-changing bindings $\mathbf{w_b}$, although at first glance the two matrices appear redundant. As mentioned above, the $\mathbf{w_a}$ matrix turned out not to be needed to account for the present data, but there is convincing evidence in the literature that long-term learning of stimulus–response associations takes place in procedural WM. We are not aware of comparable evidence for long-term position-item associations in declarative WM – this would be a prediction to be tested in future work.

12.4. Item repetition and response repetition

Our model generates the item-repetition benefit (a.k.a. object-switch cost) through the combination of two mechanisms. One is priming of the retrieval cue bound to the item, and the other is temporary strengthening of the binding between the item and its retrieval cue. The retrieval cue can be the item's spatial position, as in our list-switch paradigm and the paradigm used by Verhaeghen and colleagues (Verhaeghen & Basak, 2005; Verhaeghen & Hoyer, 2007), or some other feature uniquely bound to the item to be used. For instance, in Garavan's (1998) counting task, each count is bound to one kind of geometric figure, such that each figure displayed for counting serves as a retrieval cue to the corresponding count.

The novel finding from our Experiments 1, 2, and 5 is that item repetition incurs a cost when the list is switched. This item-repetition cost is generated in the model by response suppression: At the end of each trial the item used in that trial is suppressed in the output layer. In list-repetition trials, the joint effect of cue priming and strengthened binding overcomes the effect of response suppression, but on list-switch trials, response suppression becomes manifest as an item-repetition cost.

As we have seen in the context of Experiment 5, the item-repetition benefit arises whether or not the cue is repeated, because even when the same item is accessed through two different cues (e.g., different positions) on successive trials, cue priming from trial $n-1$ boosts activation of the correct item on trial n when it is the same item as was selected on trial $n-1$. Only when the list is switched, this benefit is eliminated because new bindings are in place in the item-selection module, so that cue activation carrying over from trial $n-1$ cues a different item in trial n than it did in trial $n-1$. Therefore, after a list switch, the pure effect of response suppression becomes manifest as a cost of repeatedly selecting the same item.

It appears therefore that two mechanisms, cue priming and response suppression, are sufficient to explain the pattern of item-repetition effects. Nevertheless, we also assume strengthening of bindings in the model. The most direct evidence for this assumption comes from the finding that repeating an item in the same position led to a stronger benefit than repeating only the position, as shown in our joint analysis of all experiments.

The same logic holds in procedural WM for response repetition effects as a function of task repetition or task switch. Response-repetition benefits in task-repetition trials arise from the joint action of stimulus priming and of strengthened stimulus–response bindings; their effect is counteracted by response suppression. Task switching obliterates the effect of the first two mechanisms, leaving the pure effect of response suppression.

Author's personal copy

13. General discussion

The two main aims of the present work were to present evidence for analogous mechanisms in declarative and procedural WM, and to develop a computational model that makes these mechanisms and their interplay explicit and allows us to generate new predictions.

We have established a number of analogous phenomena in the list-switch and the task-switch paradigm, designed to measure comparable processes of selection and updating of representations in declarative and procedural WM. Table 9 lists the corresponding phenomena in the two paradigms, and their explanation in our computational model. Note that the task-switch findings in Table 9 reflect well-established findings with the standard task-switch paradigm. Thus, our case for analogous phenomena does not rest on the specific details of our somewhat untypical task-switch paradigm in Experiment 3.

In light of these results, the theoretical framework that we started from (Oberauer, 2009) requires revision on two counts. The first revision concerns the effects of list congruency and task congruency. In the introduction we have attributed them to priming effects arising from activated LTM, bypassing the direct-access region and the bridge. Our model simulations have shown that congruency effects can to a large degree—though not entirely—be explained by residual bindings of the currently not used list or task in the direct-access region or the bridge.

Second, the item-repetition benefit (a.k.a. object-switch cost) does not reflect the time it takes for the focus of attention to switch from one item to another. The original interpretation of the object-switch cost (Garavan, 1998; Oberauer, 2003; Verhaeghen & Basak, 2005) was that an object selected for a processing step stays in the focus and thereby is immediately available for the next processing step if that step requires access to the same object. Our results show instead that a selected object is suppressed after completion of the processing step for which it was selected. This is not to say that there is no focus of attention in declarative WM. In our model, the input and output layers jointly represent the focus of attention: they serve to single out one cue (e.g., one list position) and one item

Table 9
Analogous phenomena in list switching and task switching, and their explanation in the computational model.

List-switching paradigm (declarative WM)	Task-switching paradigm (procedural WM)	Responsible model mechanisms
List-switch cost	Task-switch cost	Time to reconfigure the binding matrix
Faster RTs after longer CTI	Faster RTs after longer CTI	Set retrieval starts during CTI
Reduction of list-switch cost with CTI	Reduction of task-switch cost with CTI	Use of preparation interval to reconfigure binding matrix
List-congruency effect: Faster RTs to positional cues bound to the same item in both lists	Task-congruency effect: Faster RTs to stimulus (categories) bound to the same response in both tasks	Residual bindings of currently irrelevant set, and long-term learning of associations
No reduction of congruency effect with CTI	No reduction of congruency effect with CTI	Strength of residual bindings controlled by criterion parameter β regardless of CTI
For list-repetition trials: Repetition benefit for accessing the same item in the same position	For task-repetition trials: Repetition benefit for responses to the same stimulus[a]	Cue priming/stimulus priming plus strengthening of bindings
For list-repetition trials: Item-repetition benefit for accessing the same item in another list position	For task-repetition trials: Response-repetition benefit for responses to other stimuli of the same task-relevant category	Cue priming/stimulus priming
For list-switch trials: Item-repetition cost for accessing the same item in a different position	For task-switch trials: Response-repetition cost in response to a different stimulus	Response suppression in the candidate layer
Benefit of repeating the position without repeating any other trial feature	Benefit of repeating the stimulus (category) without repeating any other trial feature	Cue priming/stimulus priming

[a] *Note*: In most task-switching experiments, exact repetitions of stimuli in successive trials are excluded by design or excluded from analysis, so that this case is rarely observed.

Author's personal copy

bound to it, selecting that item for the next processing step. But only the cue, not the item itself, is carried over into the next step. This is sufficient to generate an item-repetition benefit as long as the bindings between input and output layer remain unchanged. Once we conceptualize the focus of attention as including not only the item but also the cue, and the cue-item binding (cf. Bialkova & Oberauer, 2010), the object-switch cost can still be interpreted as reflecting the cost of switching the focus of attention from one cue-item conjunction to another.

These revisions have been motivated jointly by experimental results and modeling. Our computational modeling efforts have clarified the theoretical framework and at the same time given new meaning to many of the framework's concepts. One important element of clarification is that the three components of declarative WM, and the corresponding three components of procedural WM, don't map onto three distinct, unitary modules in the model. The three components are not three distinct "stores" among which representations are shifted back and forth. The verbal description of the framework used a spatial metaphor for the three components as regions or containers that "hold" representations and exchange them among each other. The model architecture depicted in Fig. 11 clearly shows that this metaphor is inadequate. There is no single structure holding activated representations in LTM – rather, LTM figures in at least three separate ways in the model, as long-term learning of associations between the input and the output layer, as sustained activation of selection candidates in the candidate layer, and as a mechanism for learning and retrieving sets as chunks. The region of direct access is not a region holding representations of memory items but rather consists of the matrix of bindings between input and output layer, and the corresponding pattern of activation in the set layer (in this respect, the term "bridge" for the corresponding component of procedural WM is more apt, because these bindings form the bridge between stimuli and responses in task sets). The focus of attention is not a single structure either but consists of the currently activated representations in the input and the output layer.

The model obviously needs elaboration to account for the rich pattern of results established by research on declarative WM (for reviews see Jonides et al., 2008; Lewandowsky & Farrell, 2008) and research on procedural WM (for reviews see Kiesel et al., 2010; Koch, Gade, Schuch, & Philipp, 2010; Lien & Proctor, 2002; Vandierendonck et al., 2010). Nevertheless, we hope to have made a start towards a theoretical model of WM that is at the same time broad in its potential scope of application – integrating two strands of research – and explicit and precise about its assumptions.

Acknowledgment

The research reported in this article was supported by a grant from the Swiss National Science Foundation to Miriam Gade, Michel Druey, and Klaus Oberauer.

Appendix A. Formal description of model

A.1. Architecture

The item-selection module consists of two layers, a position layer with m units and an item layer with n units. When the model is applied to the present list-switch experiments, $m = 3$ (for three list positions) and $n = 9$ (for nine digits). When the model is applied to the standard task-switch paradigm, $m = 4$ (for the four combinations of two binary stimulus categories) and $n = 2$ (for two responses shared by both tasks).

The list-selection module consists of two layers, a cue layer with q units (where q equals the number of task cues or list cues), and a set layer with $m \times n$ units.

A.2. Representations

Item representations are vectors of length n with one value of 1 and all other values set to 0. The item is identified by the position of 1 in the vector, which is the position of the unit in the item layer that is activated when that item is encoded. Position representations are vectors of length m with

Author's personal copy

values between 0 and 1. The value j of position vector i is set according to the absolute distance between i and j and the position discrimination parameter D:

$$C_{i,j} = \exp(-D \cdot | i - j |) \tag{A1}$$

Because positional overlap plays no role in modelling the present experiments, and variations in the degree of overlap did not affect model behavior (except for increasing error rate at some point), we fixed D to 100, implying virtually no overlap between positions.

A.3. Pre-training: Encoding into item selection module

Encoding in the item-selection module occurs by two parallel learning processes, one updating the slow-changing weights $\mathbf{w_a}$ and one updating the fast-changing weights $\mathbf{w_b}$. Both use the delta rule for updating of weights. First, the activation in the input layer is updated by adding the position vector $\mathbf{c}$ for the new item j, after squashing the previous activation pattern by the input priming parameter γ:

$$\mathbf{p} = \gamma \mathbf{p} + \mathbf{c}_j \tag{A2}$$

The activation pattern $\mathbf{p}$ in the input layer is next used as retrieval cue to generate an expectation in the output layer according to the current state of the weight matrix. This occurs separately for slow and for fast changing weights:

$$\mathbf{e_a} = \mathbf{w_a} \cdot \mathbf{p} \tag{A3a}$$

$$\mathbf{e_b} = \mathbf{w_b} \cdot \mathbf{p} \tag{A3b}$$

The difference between the actual stimulus $\mathbf{s}$ and the expectation $\mathbf{e_x}$ is associated/bound to the current position vector $\mathbf{p}$ by Hebbian learning, using the learning rate η_a for slow association learning and η_b for fast binding:

$$\Delta \mathbf{w_a} = \eta_a \cdot (\mathbf{s} - \mathbf{e_a}) \cdot \mathbf{p}^T \tag{A4a}$$

$$\Delta \mathbf{w_b} = \eta_b \cdot (\mathbf{s} - \mathbf{e_b}) \cdot \mathbf{p}^T \tag{A4b}$$

A.4. Pre-training: Encoding into set selection module

After a list has been completely encoded into the item-selection module, the weight matrix $\mathbf{W}$ of the set selection module is updated according to the delta rule. First, an expectation $\mathbf{E}$ is generated in the set layer, using the activation pattern $\mathbf{q}$ in the cue layer as retrieval cue to probe the current state of $\mathbf{W}$:

$$\mathbf{E} = \mathbf{W} \cdot \mathbf{q} \tag{A5}$$

Next, the current state of the binding matrix $\mathbf{w_b}$ in the item-selection module is read out as a vector $\mathbf{B}$ of length $m \times n$, which constitutes the activation pattern in the set layer. The difference between this vectorized version of $\mathbf{w_b}$ and the expectation vector $\mathbf{E}$ is associated to the cue representation $\mathbf{q}$ by Hebbian learning, using a slow learning rate for set learning:

$$\Delta \mathbf{W} = \eta_s \cdot (\mathbf{B} - \mathbf{E}) \cdot \mathbf{q}^T \tag{A6}$$

A.5. Main task: Set selection phase

At the beginning of a run of the main task, all units in the input layer, the output layer, the cue layer and the set layer are set to zero, and all units of the candidate layer are set to the candidate activation value α, which reflects the tonic activation value of all items eligible for retrieval.

Author's personal copy

For each trial one list is cued as relevant, and after a variable CTI, one position in that list is cued to indicate the relevant item. The list cue is activated in the cue layer and used as a retrieval cue to probe **W**, generating a pattern of inputs $\mathbf{B}'$ into the set layer:

$$\mathbf{B}' = \mathbf{W} \cdot \mathbf{q} \tag{A7}$$

The input vector $\mathbf{B}'$, multiplied with the set-retrieval-rate parameter r, serves as the drift rate for the accumulation of activation in the set layer. Each set-layer unit is initialized at a random value with mean zero and a standard deviation given by the noise parameter σ, and accumulates activation at rate $r\mathbf{B}'$. Once the first unit reaches the fixed boundary of 1, set-retrieval finishes. Because evidence accumulation is linear and ballistic (i.e., the accumulation process itself is not noisy), we can compute the duration of the set-retrieval process simply as:

$$t_s = \min\left[\frac{1-\Gamma}{r\mathbf{B}'}\right] \tag{A8}$$

with Γ as the vector of random start values of the accumulators. The activation pattern accumulated in the set layer until this point is now re-arranged into a matrix of size (m, n), referred to as $\mathbf{w_c}$. This matrix is used as input for updating the binding matrix $\mathbf{w_b}$ in the item-selection layer through the delta rule:

$$\Delta\mathbf{w_b} = \eta_c(\mathbf{w}_c - \mathbf{w}_b) \tag{A9}$$

with η_c as the updating rate. This way, the bindings in the item-selection module are configured to represent the relevant list. This updating of $\mathbf{w_b}$ is repeated until the mean absolute difference between $\mathbf{w_b}$ and $\mathbf{w_c}$ falls below a criterion β. Each updating iteration lasts a constant time step of 0.1 s; thus η_c is the set-updating rate per time step. If the criterion β is reached before the end of the CTI, updating stops and the system waits until the end of the CTI. If the criterion is not yet reached at the end of the CTI, updating continues into the time after the position cue was presented until β is reached. Updating duration t_u is the number of iterations times 0.1 s.

A.6. Main task: Item selection phase

When the position cue is presented, its vector representation is added to the input layer according to Eq. (A2). After the set selection phase has ended, the required item is selected by using the position as retrieval cue to probe both the binding matrix $\mathbf{w_b}$ and the association matrix $\mathbf{w_a}$. The resulting vector is combined additively with the vector of activations in the candidate layer **a**. The sum of these two vectors determines the vector of drift rates **v** for each output unit:

$$\mathbf{v} = \mathbf{a} + \mathbf{w_a} \cdot \mathbf{p} + \mathbf{w_b} \cdot \mathbf{p} \tag{A10}$$

This drift rate determines the rate at which each output unit accumulates evidence over time. Each output unit is initialized at a random value with mean zero and a standard deviation given by the noise parameter σ. The first accumulator reaching the fixed boundary of 1 determines which item is selected. The duration of item retrieval is:

$$t_i = \min\left[\frac{1-\Gamma}{\mathbf{v}}\right] \tag{A11}$$

with Γ as the vector of random start values of the accumulators.

The total response time is calculated as the item-selection duration t_i plus any part of the sum of set-retrieval time t_s and updating time t_u that exceeds the CTI:

$$RT = \max(0, t_s + t_u - CTI) + t_i \tag{A12}$$

After each response, the following events take place: All output units except the one selected for retrieval are reset to zero. Next, the weight matrices $\mathbf{w_a}$ and $\mathbf{w_b}$ are updated by the delta rule, with learning rates (i.e., η_a and η_b) multiplied by a weight-strengthening factor f. This factor determines how much the selected item's association and binding to the current position are strengthened by further learning. Next, all units in the cue layer, the set layer, and the output layer are re-set to zero, and

Author's personal copy

all units in the input layer are squashed by multiplying them with γ. Activation of all items in the candidate layer is increased by s times their difference from the maximum activation value α; in this way, response suppression from previous trials gradually dissipates (i.e., activation grows back towards α). Then the item selected on the current trial is suppressed in the candidate layer by setting its activation to zero. Thus, the strength of response suppression is indirectly controlled by parameter α, which sets the initial activation level of units in the candidate layer and thereby determines by how much this activation is reduced through response suppression.

In sum, what carries over from one trial to the next is (1) strengthened binding and association of the selected item to its position, (2) suppression of the selected item in the candidate layer, and (3) priming of the item cue/stimulus in the input layer because the input layer is not cleared at the end of the trial but only squashed by γ.

Appendix B. The interaction of response repetition with task repetition in Experiment 3 and in the standard task switching paradigm

Our first simulation, presented in Figs. 12 and 13 of this article, applies to the design of the list-switch experiments (in particular Experiment 2) and the analogous design of the task-switch experiment (Experiment 3). When applied to task-switching as in Experiment 3, the model predicts that the interaction of response repetition with task repetition disappears for congruent trials. The data of Experiment 3 confirmed this prediction: Among congruent trials, response repetitions were faster than response changes regardless of task repetition or task switch. In contrast, our simulation of the standard task-switch paradigm showed the response-repetition by task-repetition interaction even among congruent trials, in agreement with findings from the standard task-switch paradigm. Why does the model behave differently when applied to the standard task-switch paradigm than when applied to the design of Experiment 3?

The reason is that the standard task-switch paradigm includes two congruent conditions that don't exist in the design of our Experiment 3:

(1) Task repetition, stimulus switch and response repetition, e.g., repeating the shape task, switching from an (incongruent) green circle to a (congruent) blue circle, leading to the same response (e.g., the right key).
(2) Task switch, stimulus switch and response repetition, e.g., switching from color to shape task, switching from an (incongruent) blue square to a (congruent) blue circle, leading to the same response (e.g., right key).

These conditions differ in the effects of stimulus priming, and thereby generate the response-repetition by task-repetition interaction for congruent trials. In condition (1), residual activation of the previous stimulus (green circle) activates the correct response, so stimulus priming has a beneficial effect. In condition (2), residual activation of the blue square, fed through the new task set, now activates the left key (the previous response was right key, but because the blue square is an incongruent stimulus, it now activates the left key), hence the stimulus-priming effect is detrimental. The stimulus-priming effect in condition (1) speeds up response repetitions in task-repetition trials, and the stimulus-priming effect in condition (2) slows down response repetitions in task-switch trials; together they create the response-repetition by task-repetition interaction. These two conditions are absent in the design of Experiment 3: Within one task, no two stimuli lead to the same response (thereby excluding condition 1), and across two tasks, the same response was only given to the same stimulus (thereby excluding condition 2).

References

Allport, A., Styles, E. A., & Hsieh, S. (1994). Shifting intentional set: Exploring the dynamic control of tasks. In C. Umiltá & M. Moscovitch (Eds.). *Attention & performance* (Vol. XV, pp. 421–452). Cambridge: MIT Press.

Altmann, E. M. (2011). Testing probability matching and episodic retrieval accounts of response repetition effects in task switching. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 37*, 935–951.

Author's personal copy

Altmann, E. M., & Gray, W. D. (2008). An integrated model of cognitive control in task switching. *Psychological Review, 115*(602–639).
Anderson, J. R. (1987). Skill acquisition: Compilation of weak-method problem solutions. *Psychological Review, 94*, 192–210.
Anderson, J. R. (2007). *How can the human mind occur in the physical universe?* Oxford: Oxford University Press.
Berman, M. G., Jonides, J., & Lewis, R. L. (2009). In search of decay in verbal short-term memory. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 35*, 317–333.
Bertelson, P. (1965). Serial choice reaction-time as a function of response versus signal-and-response repetition. *Nature, 206*, 217–218.
Bialkova, S., & Oberauer, K. (2010). Direct access to working memory contents. *Experimental Psychology, 57*(5), 383–389.
Botvinick, M., & Watanabe, T. (2007). From numerosity to ordinal rank: A gain-field model of serial order representation in cortical working memory. *Journal of Neuroscience, 27*, 8636–8642.
Brown, S. D., & Heathcote, A. (2008). The simplest complete model of choice response time: Linear ballistic accumulation. *Cognitive Psychology, 57*, 153–178.
Brown, J. W., Reynolds, J. R., & Braver, T. S. (2007). A computational model of fractionated conflict-control mechanisms in task-switching. *Cognitive Psychology, 55*, 37–85.
Brozović, M., Abbott, L. F., & Andersen, R. A. (2008). Mechanisms of gain modulation at single neuron and network levels. *Journal of Computational Neuroscience, 25*, 158–168.
Burgess, N., & Hitch, G. J. (1999). Memory for serial order: A network model of the phonological loop and its timing. *Psychological Review, 106*, 551–581.
Burgess, N., & Hitch, G. J. (2005). Computational models of working memory: Putting long-term memory into context. *Trends in Cognitive Sciences, 9*, 535–541.
Burgess, N., & Hitch, G. J. (2006). A revised model of short-term memory and long-term learning of verbal sequences. *Journal of Memory and Language, 55*, 627–652.
Campbell, K. C., & Proctor, R. W. (1993). Repetition effects with categorizable stimulus and response sets. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 19*, 1345–1362.
Cohen-Kdoshay, O., & Meiran, N. (2009). The representation of instructions operates like a prepared reflex: Flanker compatibility effects found in first trial following S–R instructions. *Experimental Psychology, 56*, 128–133.
Cowan, N. (1995). *Attention and memory: An integrated framework*. New York: Oxford University Press.
De Jong, R. (2000). An intention–activation account of residual switch costs. In S. Monsell & J. Driver (Eds.), *Control of cognitive processes: Attention and performance XVIII* (pp. 357–376). Cambridge, MA: MIT Press.
Dreisbach, G., Goschke, T., & Haider, H. (2006). Implicit task sets in task switching? *Journal of Experimental Psychology: Learning, Memory, and Cognition, 32*, 1221–1233.
Dreisbach, G., Goschke, T., & Haider, H. (2007). The role of task rules and stimulus–response mappings in the task switching paradigm. *Psychological Research, 71*, 383–392.
Druey, M. (2011). *Stimulus category priming and response category inhibition best explain the stimulus and response category repetition effects in task switching*. Unpublished manuscript. University of Zurich.
Druey, M., & Hübner, R. (2008). Effects of stimulus features and instruction on response coding, selection, and inhibition: Evidence from repetition effects under task switching. *Quarterly Journal of Experimental Psychology, 61*, 1573–1600.
Farrell, S. (2012). Temporal clustering and sequencing in short-term memory and episodic memory. *Psychological Review, 119*, 223–271.
Farrell, S., & Lewandowsky, S. (2004). Modelling transposition latencies: Constraints for theories of serial order memory. *Journal of Memory and Language, 51*, 115–135.
Farrell, S., & Lewandowsky, S. (2010). Computational models as aids to better reasoning in psychology. *Current Directions in Psychological Science, 19*, 329–335.
French, R. M., Addyman, C., & Mareschal, D. (2011). TRACX: A recognition-based connectionist framework for sequence segmentation and chunk extraction. *Psychological Review, 118*, 614–636.
Garavan, H. (1998). Serial attention within working memory. *Memory & Cognition, 26*, 263–276.
Gehring, W. J., Bryck, R. L., Jonides, J., Albin, R. L., & Badre, D. (2003). The mind's eye, looking inward? In search of executive control in internal attention shifting. *Psychophysiology, 40*, 572–585.
Gilbert, S. J., & Shallice, T. (2002). Task switching: A PDP model. *Cognitive Psychology, 44*, 297–337.
Halford, G. S., Cowan, N., & Andrews, G. (2007). Separating cognitive capacity from knowledge: A new hypothesis. *Trends in Cognitive Sciences, 11*, 236–242.
Henson, R. N. A. (1998a). Item repetition in short-term memory: Ranschburg repeated. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 24*, 1162–1181.
Henson, R. N. A. (1998b). Short-term memory for serial order: The start-end model. *Cognitive Psychology, 36*, 73–137.
Hick, W. E. (1952). On the rate of gain of information. *Quarterly Journal of Experimental Psychology, 4*, 11–26.
Hommel, B. (2000). The prepared reflex: Automaticity and control in stimulus–response translation. In S. Monsell & J. Driver (Eds.), *Attention and performance XVIII: Control of cognitive processes* (pp. 247–273). Cambridge, MA: MIT Press.
Horner, A. J., & Henson, R. A. (2009). Bindings between stimuli and multiple response codes dominate long-lag repetition priming in speeded classification tasks. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 35*, 757–779.
Hübner, R., & Druey, M. (2006). Response execution, selection, or activation: What is sufficient for response-related repetition effects under task shifting? *Psychological Research, 70*, 245–261.
Hübner, M., Kluwe, R. H., Luna-Rodriguez, A., & Peters, A. (2004). Response selection difficulty and asymmetrical costs of switching between tasks and stimuli: No evidence for an exogenous component of task-set reconfiguration. *Journal of Experimental Psychology: Human Perception and Performance, 30*, 1043–1063.
Jersild, A. T. (1927). Mental set and shift. *Archives of Psychology, 89*.
Jones, M. N., Kintsch, W., & Mewhort, D. J. K. (2006). High-dimensional semantic space accounts of priming. *Journal of Memory and Language, 55*, 534–552.
Jonides, J., Lewis, R. L., Nee, D. E., Lustig, C. A., Berman, M. G., & Moore, K. S. (2008). The mind and brain of short-term memory. *Annual Review of Psychology, 59*, 193–224.

Author's personal copy

Kessler, Y., & Meiran, N. (2010). The reaction-time task-rule congruency effect is not affected by working memory load: Further support for the activated long-term memory hypothesis. *Psychological Research, 74*, 388–399. http://dx.doi.org/10.1007/s00426-009-0261-z.

Kiesel, A., Steinhauser, M., Wendt, M., Falkenstein, M., Jost, K., Philipp, A. M., et al (2010). Control and interference in task switching – A review. *Psychological Bulletin, 136*, 849–874.

Kiesel, A., Wendt, M., & Peters, A. (2007). Task switching: On the origin of response congruency effects. *Psychological Research, 71*, 117–125.

Kleinsorge, T. (1999). Response repetition benefits and costs. *Acta Psychologica, 103*, 295–310.

Koch, I., Gade, M., Schuch, S., & Philipp, A. M. (2010). The role of inhibition in task switching: A review. *Psychonomic Bulletin & Review, 17*, 1–14.

Lewandowsky, S., & Farrell, S. (2008). Short-term memory: New data and a model. In B. H. Ross (Ed.). *The psychology of learning and motivation* (Vol. 49, pp. 1–48). London, UK: Elsevier.

Lewandowsky, S., & Farrell, S. (2011). *Computational modeling in cognition: Principles and practice*. Thousand Oaks, CA: Sage.

Lewis-Peacock, J. A., Drysdale, A. T., Oberauer, K., & Postle, B. R. (2011). Neural evidence for a distinction between short-term memory and the focus of attention. *Journal of Cognitive Neuroscience, 24*, 61–79.

Li, Z., Bao, M., Chen, X., Zhang, D. R., Han, S., He, S., et al (2006). Attention shift in human verbal working memory: Priming contribution and dynamic brain activation. *Brain Research, 1078*, 132–142.

Li, K. Z. H., Lindenberger, U., Rünger, D., & Frensch, P. A. (2000). The role of inhibition in the regulation of sequential action. *Psychological Science, 11*, 343–347.

Liefooghe, B., Barrouillet, P., Vandierendonck, A., & Camos, V. (2008). Working memory costs of task switching. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 34*, 478–494.

Lien, M.-C., & Proctor, R. W. (2002). Stimulus–response compatibility and psychological refractory period effects: Implications for response selection. *Psychonomic Bulletin & Review, 9*, 212–238.

Logan, G. D. (1978). Attention in character-classification tasks: Evidence for the automaticity of component stages. *Journal of Experimental Psychology: General, 107*, 32–63.

Logan, G. D. (2004). Working memory, task switching, and executive control in the task span procedure. *Journal of Experimental Psychology: General, 133*, 218–236.

Logan, G. D. (2007). What it costs to implement a plan: Plan-level and task-level contributions to switch costs. *Memory & Cognition, 35*, 591–602.

Logan, G. D., & Bundesen, C. (2003). Clever homunculus: Is there an endogenous act of control in the explicit task-cuing procedure? *Journal of Experimental Psychology: Human Perception and Performance, 29*, 575–599.

Logan, G. D., & Gordon, R. D. (2001). Executive control of visual attention in dual-task situations. *Psychological Review, 108*, 393–434.

MacKay, D. G. (1987). *The organization of perception and action*. Berlin: Springer.

Mayr, U. (2001). Age differences in the selection of mental sets: The role of inhibition, stimulus ambiguity, and response-set overlap. *Psychology & Aging, 16*, 96–109.

Mayr, U., & Bryck, R. L. (2005). Sticky rules: Integration between abstract rules and specific actions. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 31*, 337–350.

Mayr, U., & Kliegl, R. (2000). Task-set switching and long-term memory retrieval. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 26*, 1124–1140.

Mayr, U., & Kliegl, R. (2003). Differential effects of cue changes and task changes on task-set selection costs. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 29*, 362–372.

McKone, E. (1998). The decay of short-term implicit memory: Unpacking lag. *Memory & Cognition, 26*, 1173–1186.

Meiran, N. (2000). Modeling cognitive control in task-switching. *Psychological Research, 63*, 234–249.

Meiran, N., Chorev, Z., & Sapir, A. (2000). Component processes in task-switching. *Cognitive Psychology, 41*, 211–253.

Meiran, N., & Cohen-Kdoshay, O. (2012). Working memory load but not multitasking eliminates the prepared reflex: Further evidence from the adapted flanker paradigm. *Acta Psychologica, 139*, 309–313.

Meiran, N., & Kessler, Y. (2008). The task-rule congruency effect in task switching reflects activated long-term memory. *Journal of Experimental Psychology: Human Perception and Performance, 34*, 137–157.

Meiran, N., Kessler, Y., & Adi-Japha, E. (2008). Control by action representation and input selection (CARIS): A theoretical framework for task switching. *Psychological Research, 72*, 473–500.

Miller, G. A., Galanter, E., & Pribram, K. H. (1960). *Plans and the structure of behavior*. New York: Holt, Rinehart & Winston.

Monsell, S. (1978). Recency, immediate recognition memory, and reaction time. *Cognitive Psychology, 10*, 465–501.

Monsell, S. (2003). Task switching. *Trends in Cognitive Sciences, 7*, 134–140.

Oberauer, K. (2001). Removing irrelevant information from working memory. A cognitive aging study with the modified Sternberg task. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 27*, 948–957.

Oberauer, K. (2002). Access to information in working memory: Exploring the focus of attention. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 28*, 411–421.

Oberauer, K. (2003). Selective attention to elements in working memory. *Experimental Psychology, 50*(4), 257–269.

Oberauer, K. (2005). Control of the contents of working memory – A comparison of two paradigms and two age groups. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 31*, 714–728.

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

Oberauer, K., & Göthe, K. (2006). Dual-task effects in working memory: Interference between two processing tasks, between two memory demands, and between storage and processing. *European Journal of Cognitive Psychology, 18*, 493–519.

Oberauer, K., & Vockenberg, K. (2009). Updating of working memory: Lingering bindings. *Quarterly Journal of Experimental Psychology, 62*, 967–987.

Pashler, H. (1994). Dual-task interference in simple tasks: Data and theory. *Psychological Bulletin, 116*, 220–244.

Perruchet, P., & Vinter, A. (1998). PARSER: A model for word segmentation. *Journal of Memory and Language, 39*, 246–263.

Risse, S., & Oberauer, K. (2010). Selection of objects and tasks in working memory. *Quarterly Journal of Experimental Psychology, 63*(4), 784–804.

Author's personal copy

Rogers, R. D., & Monsell, S. (1995). Costs of a predictable switch between simple cognitive tasks. *Journal of Experimental Psychology: General, 124*, 207–231.
Rothermund, K., Wentura, D., & De Houwer, J. (2005). Retrieval of incidental stimulus–response associations as a source of negative priming. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 31*, 482–495.
Salinas, E., & Thier, P. (2000). Gain modulation: A major computational principle of the central nervous system. *Neuron, 27*, 15–21.
Schneider, D. W., & Anderson, J. R. (2011). A memory-based model of Hick's law. *Cognitive Psychology, 62*, 193–222.
Schneider, D. W., & Logan, G. D. (2007). Retrieving information from a hierarchical plan. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 33*, 1076–1091.
Schneider, D. W., & Logan, G. D. (2009). Selecting a response in task switching: Testing a model of compound cue retrieval. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 35*, 122–136.
Schuch, S., & Koch, I. (2004). The costs of changing the representation of action: Response repetition and response–response compatibility in dual tasks. *Journal of Experimental Psychology: Human Perception and Performance, 30*, 566–582.
Smith, M. C. (1968). Repetition effect and short-term memory. *Journal of Experimental Psychology, 77*, 435–439.
Souza, A. S., Oberauer, K., Gade, M., & Druey, M. D. (2012). Processing of representations in declarative and procedural working memory. *Quarterly Journal of Experimental Psychology, 65*, 1006–1033.
Sternberg, S. (1969). Memory scanning: Mental processes revealed by reaction-time experiments. *American Scientist, 57*, 421–457.
Usher, M., & McClelland, J. L. (2001). The time course of perceptual choice: The leaky, competing accumulator model. *Psychological Review, 108*, 550–592.
Vandierendonck, A., Liefooghe, B., & Verbruggen, F. (2010). Task switching: Interplay of reconfiguration and interference control. *Psychological Bulletin, 136*, 601–626.
Verhaeghen, P., & Basak, C. (2005). Ageing and switching of the focus of attention in working memory: Results from a modified N-back task. *Quarterly Journal of Experimental Psychology*.
Verhaeghen, P., & Hoyer, W. J. (2007). Aging, focus switching, and task switching in a continuous calculation task: Evidence toward a new working memory control process. *Aging, Neuropsychology, and Cognition, 14*, 22–39.
Waszak, F., Hommel, B., & Allport, A. (2003). Task-switching and long-term priming: Role of episodic stimulus-task bindings in task-shift costs. *Cognitive Psychology, 46*, 361–413.
Waszak, F., Wenke, D., & Brass, M. (2008). Cross-talk of instructed and applied arbitrary visuomotor mappings. *Acta Psychologica, 127*, 30–35.
Wendt, M., & Kiesel, A. (2008). The impact of stimulus-specific practice and task instructions on response congruency effects between tasks. *Psychological Research, 72*, 425–432.
Woltz, D. J., & Was, C. A. (2007). Available but unattended conceptual information in working memory: Temporarily active semantic content or persistent memory for prior operations? *Journal of Experimental Psychology: Learning, Memory, and Cognition, 33*, 155–168.
Yamaguchi, M., & Proctor, R. W. (2011). Automaticity without extensive training: The role of memory retrieval in implementation of task-defined rules. *Psychonomic Bulletin & Review, 18*, 347–354.
Yehene, E., & Meiran, N. (2007). Is there a general task switching ability? *Acta Psychologica, 126*, 169–195.