A Two-level Computational Architecture for Modeling Human Joint Action

Jens Pfau (jens.pfau@cgi.com)
CGI Deutschland Ltd. & Co. KG, Space Business Unit, 64295 Darmstadt, Germany

Liz Sonenberg (l.sonenberg@unimelb.edu.au)
Department of Computing and Information Systems
University of Melbourne, Parkville, VIC 3010 Australia

Yoshihisa Kashima (ykashima@unimelb.edu.au)
School of Psychological Sciences
University of Melbourne, Parkville, VIC 3010 Australia

Abstract

We propose a computational architecture of human joint action that accounts for interactions between higher- and lower-level coordination processes. A proof-of-concept implementation of the architecture is used to model the social Simon task, a well known experimental task that reveals an interplay between higher- and lower-level processes. We show that our model is able to generate results aligned with human performance data for four task configurations. This work contributes to an understanding of mechanisms involved in joint actions.

Keywords: Joint Action; Computational Cognitive Model.

Introduction

Coordination during joint actions typically requires collaborative planning as well as fine-grained alignment of movements. Two musicians performing together have to agree on the pieces they are going to perform and on the parts each of them is going to play. During their performance, however, they synchronize their movements to a level of detail that goes well beyond the specification of this overall plan.

Philosophers have long analyzed higher-level collaborative planning processes based on propositional attitudes such as (joint) intentions, plans, goals, and beliefs (e.g. Bratman, 1992; Tuomela, 2000).

Notions of shared intention have also underpinned investigations by psychologists (e.g. Pacherie, 2011), though empirical psychological research has put much emphasis on the role of lower-level mechanisms of coordination in joint action such as direct perception-action links (e.g. Wolpert et al., 2003; Haazebroek et al., 2011) and it has been noted that integrative perspectives in psychology that represent the interplay between higher- and lower-level coordination processes are less well studied (Knoblich et al., 2011). Psychological studies suggest that coactors develop shared task representations, i.e. they tend to represent their partners' part of the joint action even if this is not required for successful performance (Sebanz et al., 2005), but there remains debate about the nature and detail of what is shared (Knoblich et al., 2011).

Complementing these philosophical and psychological perspectives, a large body of computational work formalizes models of joint action and collaborative planning, (e.g. Grosz & Kraus, 1996; Rao & Georgeff, 1995; Tambe, 1997) as well as models of lower-level coordination processes, (e.g. Hurley, 2008; Wolpert et al., 2003). Although these latter models often seek to explain the emergence of higher-level coordination from lower-level processes, they are not as powerful in representing collaborative plans as are the former models based on higher-level propositional attitudes.

Computational approaches to cognitive modelling can play many roles as discussed by Sun (2009). In particular, computational cognitive models of joint action can inform applied research on human-robot interaction (e.g. Haazebroek et al., 2011; Vesper, 2013, p. 146), a motivation we share.

Multilayer computational cognitive models have been studied for some time (see Thagard, 2012), and examples are mentioned in related work below. We build on such work to pose a specific set of building blocks that can account for observed phenomena in joint action. We describe a computational architecture that includes interactions between collaborative planning based on propositional attitudes and lower-level coordination processes. In gathering evidence in support of the proposed architecture, we draw on theoretical and empirical work on human joint action. We provide a proof-of-concept implementation and simulate a particular experimental task—the *Simon task* (Simon & Rudell, 1967)—that, together with its social variant (Sebanz et al., 2005), has revealed interactions between higher-level planning and lower-level coordination processes. We demonstrate that our model can account for empirical results obtained from different conditions of the Simon task.

Next we describe our architecture. We then describe its (partial) implementation as a model for the social Simon task, present our analysis and a short comparison with related work, and briefly conclude the paper.

Architecture

Figure 1 shows an overview of our architecture, which is composed of two levels. At the lower *perception-action level* perceptual input is received from the environment and mental action representations are translated into muscular movements. Shared representations for perception and action based on common coding theory (Prinz, 1997) support the engagement in joint actions. At the upper *intentional level* practical reasoning operates on higher-level mental attitudes. The distinction between these levels is not crisp and only adopted to guide the discussion and not to make strict distinctions.

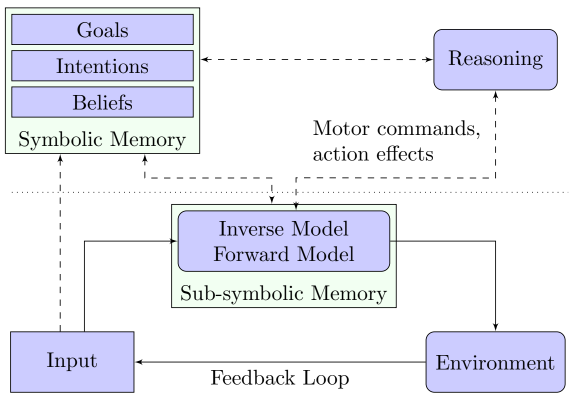

Figure 1: The levels of the architecture. Rectangles denote representations, rectangles with rounded corners denote processes. Solid lines show the feedback loop at the perception-action level. Dashed lines represent information flow.

Two Memory Systems

We employ a dual-process model of a fast, automatic, and subconscious mode of processing and a slow, controlled, and conscious mode of processing. Following common practice (Smith & DeCoster, 2000), we assume that the first mode of processing operates on a *sub-symbolic memory* and the second mode of processing operates on a *symbolic memory*. Symbolic memory and its processing mode enable the processes of the intentional level. Sub-symbolic memory and its processing mode enable the processes of the perception-action level. Via interactions between these two memories, intentional and perception-action level and thereby higher-level and lower-level coordination processes interact.

Building on connectionist models (e.g. Rumelhart, McClelland, & PDP Research Group, 1986), sub-symbolic memory is composed of so called *features*, each of which has a certain *activation level* at any point in time. Activation spreads between features via inhibitory and excitatory connections. Hence, this memory encodes associations between features, and processing exercises these associations by propagating activation. The current activation pattern constitutes a kind of working memory. Learning adjusts connections between features based on how often they are activated simultaneously.

Symbolic memory consists of representations in a language that allows for symbolic reasoning. Processing on this memory amounts to logical inference.

Symbols in symbolic memory are represented by sets of features in sub-symbolic memory. Activating a set of features representing a symbol affects the truth value of that symbol in symbolic memory. However, only those symbols whose features are sufficiently activated are available for reasoning. This constitutes another kind of working memory and allows perceptual context, which is encoded by activated features, to influence which information is accessible for higher-level reasoning. Inferring a formula by symbolic reasoning causes corresponding features in sub-symbolic memory to be activated, which can lead to further activation of features. Mappings between symbols and features are subject to learning.

Actions

An *action* is represented by a *motor command* that produces a movement and by its expected perceptual *action effects*. Both motor commands and action effects are represented by feature sets in sub-symbolic memory. In symbolic memory, these feature sets are represented by symbols. This enables action representations to be shared between the intentional and perception-action levels. In sub-symbolic memory, different action representations can have overlapping features. Likewise, representations of action effects and perceptual input can have overlapping features, as postulated by Prinz (1997). Because the features of the motor command and of its effects are activated at the same time frequently, there is a bi-directional association between motor command and effects in sub-symbolic memory. Consequently, an activation of features associated with the effects of an action also activates the associated motor command and vice versa. This allows for a translation between action effects and motor commands and the planning of actions in terms of their effects. Furthermore, the same action representation can be activated multiple times, yielding an increased activation level; and multiple action representations can be activated at the same time.

Perception-Action Level

We adopt a control system perspective to perception and action for the perception-action level (Hurley, 2008; Wolpert et al., 2003). Two types of internal models are distinguished: *Inverse models* determine the motor command required to cause particular effects. *Forward models* predict the effects of motor commands. Inverse and forward models are implemented by associations between features in sub-symbolic memory.

An appropriate composition of inverse and forward models enables the actor to deal with basic motor control without the involvement of any higher-level cognition. We consider the set-up depicted in Figure 2. We assume that input received from the environment causes an activation of features in sub-symbolic memory.

The inverse model translates effects into a motor command given the current input, which specifies the preconditions that the motor command has to satisfy. In basic execution mode, the effects correspond to a *goal state* that the control system is to achieve (point 2 in Figure 2). The execution of the activated motor command acts on the environment, in turn affecting the input to the inverse model. An internal forward model estimates the effects of the current motor command, supporting the inverse model in its control task. When the inverse model fails to control for the error between the input and goal state, control returns to the intentional level to correct for that error.

The inverse model can be used to generate motor commands (3) for different goal states (2). The forward model can then be used to make predictions for the effects (1) of these motor commands (4). This prediction can also be applied to another actor's actions. In line with common coding theory (Prinz, 1997), we assume that the observed effects of others' movements activate corresponding action effect fea-

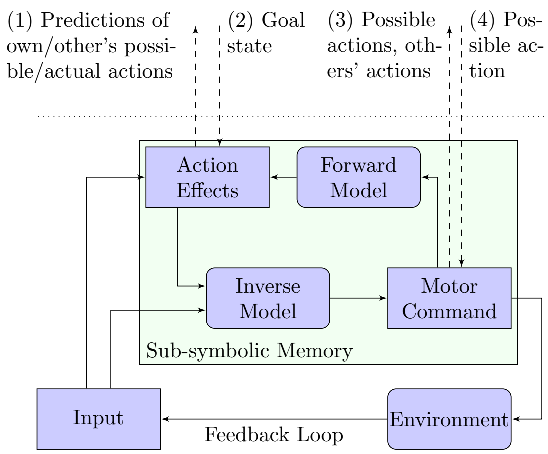

Figure 2: Control and information flow at the perception-action level. Rectangles depict representations, rectangles with rounded corners depict processes. Solid lines represent control flow and dashed lines show the points at which information exchange with the intentional level happens.

tures, causing the respective motor command to be activated via the inverse model. When the extracted motor command (3) is used as an input to the forward model (4), *action simulation* is obtained, which allows to predict the other actor's movement (1) and infer their intentions at a basic level. Given sufficient activation, copying of the other actor's movements results, which facilitates synchronization and imitation. A side-effect is that any input can lead to the activation of action effect features. This represents the interference of observed and planned actions postulated by common coding theory.

Intentional Level

The intentional level implements practical reasoning based on higher-level mental attitudes such as beliefs, goals, and intentions and builds on the perception-action level.

Practical reasoning employs *means-end reasoning* and *intention inference* (using symbolic reasoning and action simulation). Practical reasoning determines the construction of joint intentions and enables, for example, social factors to modulate whether shared task representation are constructed. A *joint intention* is a mental attitude that links the coactors' intentions and practical reasoning to each other's actions and to the overall joint action. Joint intentions drive collaborative planning towards the goal of the joint action (Bratman, 1992). Action representations consist of effects (end) and motor commands (means) as described previously. Goals and intentions are arranged in a hierarchy of alternating levels. A goal corresponding to action effects and an associated intention referring to an action that corresponds to a motor command form the lowest level of this structure. This represents the integration of the perception-action and intentional levels. At higher levels of the hierarchy intentions refer to action plans instead of primitive motor commands.

Goals, intentions, and beliefs are attributed to particular actors, which contrasts with feature activations. If we assume that joint intentions follow the same structure as individual

(a) Stimulus-response compatibility

(b) Stimulus-response incompatibility

Figure 3: Stimulus-response compatibility and incompatibility in the Simon task. In (a), the stimulus appears on the side of the button that is to be pushed. In (b), the stimulus appears on the other side. Note that buttons are not colored in the actual task.

intentions (hierarchies of goals and plans), they integrate well with the practical reasoning of individual actors.

The Simon Task

In the *individual Simon task* a subject pushes one of two buttons (left or right) depending on a non-spatial attribute of a *stimulus* object appearing on a screen. The non-spatial, *task-relevant* stimulus attribute is typically the color of this object. For example, a subject could be instructed to push the left button when the object is green and the right button when the object is red. It turns out that reaction time increases if spatial, *task-irrelevant* attributes of the object are incompatible with spatial features of the expected *response* (push left or right button). The spatial, task-irrelevant attribute is typically the location of the object on the screen. For example, the object can appear on the left or right side of the screen and be either on the same side as the button that is to be pushed or on the other. In the first case, we talk about *stimulus-response compatibility* and about *stimulus-response incompatibility* in the second case (Figure 3). The increase in reaction time in the stimulus-response incompatibility condition is called the *Simon effect* (Simon & Rudell, 1967). The Simon effect is absent when the subject performs only one part of the task.

In the *social Simon task*, two subjects carry out the Simon task together, i.e. each subject is responsible for one of the two stimulus-response mappings (task rules). A task-irrelevant spatial attribute referring to the other subject's action leads to an increase in reaction time similar to the one in the individual Simon task (Sebanz et al., 2005). Such an increase in reaction time does not occur if the subjects carry out their parts of the task individually. This suggests that subjects corepresent their coactor's action in the joint task (*action corepresentation*). The representation of the coactor's action can be activated by a compatible stimulus feature, causing an *action conflict* (i.e. a situation where the inappropriate action receives activation which needs to be suppressed to allow the correct action). Consequently a Simon effect is observed.

An increase in reaction time is also observed when a stimulus calls for both subjects to carry out an action at the same time (*task conflict*). The interpretation is that a subject also corepresents the task rule of the partner (*task corepresentation*). If the subject corepresents their coactor's task rule, the associated action is activated when the stimulus triggers that rule's precondition. Like with the action conflict, a task conflict is not observed when the subject performs its part of the

Setup	Conditions			
	No Conflict	Action Conflict	Task Conflict	Action and Task Conflict
Left subject's task	green → left	green → **left**	**right** → left	**left** → **left**
Right subject's task	**red** → **right**	**red** → right	**red** → **right**	**red** → right
Stimulus color/location	red/right	red/left	red/right	red/left

Table 1: The experimental conditions of the social Simon task. By green → left we mean that the subject reacts to a green stimulus by pushing the left button. In the action-conflict condition, there is an action conflict for the right subject. The parts of each task rule activated by the stimulus either because of the relevance to the rule's precondition or an overlap with the features of its action are printed in **bold**.

task individually. Action and task corepresentation can affect task performance in isolation as well as simultaneously. Results show that the reaction time increase due to task conflicts is larger than the one due to action conflicts. If action and task conflict occur simultaneously, reaction time is more than the sum of the reaction times when action and task conflict occur in isolation. Table 1 lists the different experimental conditions and the conflicts they evoke for the right-hand side subject. The no-conflict and action-conflict conditions mirror the same conditions in the individual Simon task.

For a recent review of research involving the Simon task, and its social variant, refer to Dolk et al. (2014).

Model and Analysis

We describe an implementation of those parts sufficient to model the (right-hand side) subject in the individual and social Simon tasks based on our architecture. We compare the Simon effect in our simulations with empirical data.

Sub-symbolic and symbolic memory provide representations of stimulus features (green color, red color, left position, right position) and of the effects and motor commands of the two available actions (push left and push right). Beliefs and goals of agents and the effects of actions are represented with propositional logic. Figure 4 displays the mapping between the elements of both memories and the associations between features in sub-symbolic memory. We assume mapping and associations have been established by some means a priori. To prevent clutter, we do not show here symbols and features referring to stimulus color.

Sub-symbolic memory contains feature sets representing the goal (action effects) of perceiving the left button being pushed (l, u_1, u_2) and the right button being pushed (r, v_1, v_2). Via threshold units, both feature sets forward activation to respective features representing the motor commands achieving these goals (x_1, x_2, x_3 and y_1, y_2, y_3). This represents the inverse model. Motor command features are part of a fully recurrent auto-associative connectionist network in which activation settles into that previously learnt pattern which is closest to the current input (Rumelhart et al., 1986). By prior learning, strong associations were created between the features of the same motor command. Hence, input to both motor commands leads to the respective feature sets competing for activation. The activation pattern requires some time to settle to a stable state. The feature s represents a stimulus, whose activation together with the feature for right (r) or left

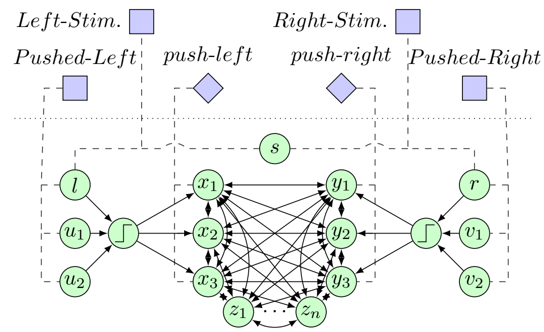

Figure 4: Elements of symbolic and sub-symbolic memory and their relationships. At the intentional level, rectangles denote literals in symbolic memory, diamonds denote actions. At the perception-action level, circles denote features in sub-symbolic memory. Dashed lines show associations between features and symbols, and solid lines with arrows show associations between different features. The threshold symbol indicates that activation through this unit is only propagated if it exceeds a certain threshold.

(l) or features representing color (not shown in the figure) represents the perception of a stimulus attribute. Note that the features for right and left (r and l) are both part of a set of features representing action effects and a set of features representing a stimulus attribute. This amounts to the common coding of actions and observations. Note that high-level action representations *push-$left$* and *push-right* are not available a priori but obtained by employing the inverse model.

Features in sub-symbolic memory can receive activation in four ways: (1) when a belief, goal, or intention is created (*attitude representation*); (2) when a stimulus is perceived (*stimulus perception*); (3) when the intentional level employs the inverse model at the perception-action level by providing a goal state to retrieve an appropriate motor command (*action planning*); (4) when the intentional level invokes the execution of a primitive action (*action execution*). Feature activations in sub-symbolic memory due to stimulus perception are translated into corresponding symbols in symbolic memory.

Only together, attitude representation and stimulus perception generate enough activation on the features representing the action effects *Pushed-$Left$* (*Pushed-Right*) so that this activation is propagated to the features representing the motor command *push-left* (*push-right*). In Figure 4, this is represented by threshold units. For example, representing a goal which refers to *Pushed-$Left$* is not sufficient for activation to be propagated to the features of *push-$left$*. Likewise, activating the $left$ feature is not sufficient to activate the features of *push-left*. Thereby the agent is able to represent and plan with goals without executing an action. We assume that activation due to action planning and execution is sufficient to overcome this threshold, which enables action planning based on inverse models and the execution of motor commands. A motor command is executed once sufficient activation is provided and its features settle into a stable pattern.

The intentional level is driven by a modified BDI interpreter that implements practical reasoning based on goals, intentions, and beliefs (Rao & Georgeff, 1995). Means-end reasoning can be based on symbolic reasoning (as is standard) or the inverse model at the perception-action level can be employed to retrieve a motor command for a given goal state. An agent annotates each goal and intention with the actor(s) that is (are) supposed to hold that attitude. A joint intention consists of a goal that has multiple actors and of its subordinate intentions and goals. An agent can plan for coactors but does not act on intentions that it is not the sole actor of.

There is an individual and a social task. In the individual task, there is a no-conflict (*NoC*) and an action-conflict (*AC*) condition. The social task adds a task-conflict (*TC*) and a both-conflicts (*TCAC*) condition. All conditions can be setup so that only one part of the task is represented (no corepresentation) or both parts (corepresentation). Corepresentation is the default but was manipulated by Sebanz et al. (2005) in the individual and by Hommel et al. (2009) in the social task.

A top goal $Goal_A(SimonTask)$ is provided to the agent where A is the set of agents performing the task, e.g. $\{you, me\}$. The agent is equipped with a complex action *SimonTask*, which can be performed to achieve the top goal and itself evokes subgoals according to the experimental condition, e.g. $Goal_{\{you\}}(\textit{Green-Stimulus} \Rightarrow \textit{Pushed-Left})$ and $Goal_{\{me\}}(\textit{Red-Stimulus} \Rightarrow \textit{Pushed-Right})$. The first subgoal means that if the stimulus is green, the left button is to be pushed by the other agent.

In the action-conflict condition, the agent adopts the top goal and means-end reasoning creates an intention to perform the *SimonTask* action with the other subject. This leads to the adoption of the above mentioned subgoals. With corepresentation the subgoal for the other subject is represented. By representing subgoals, activation is added to the corresponding features of symbols (i.e. *Green-Stimulus*, *Red-Stimulus*, *Pushed-Left*, *Pushed-Right*). This is what corresponds to action and task corepresentation according to Sebanz et al. (2005). Both the other subject's task (encoded in the subgoal) and action (via action effects) is corepresented. However, no activation is propagated to motor command features yet.

A red stimulus is represented on the left. This leads to an activation of the features s and l and the feature representing red. Now the feature set representing *Pushed-Left* has sufficient input to have activation leak over into the features representing the *push-left* action. Also, corresponding propositions (*Red-Stimulus* and *Left-Stimulus*) are then made true in symbolic memory. Now, the subgoal $Goal_{\{me\}}(\textit{Red-Stimulus} \Rightarrow \textit{Pushed-Right})$ needs to be achieved because *Red-Stimulus* is true and *Pushed-Right* false. Means-end reasoning employs the inverse model to obtain an action that can achieve *Pushed-Right*, which we call *push-right* but which was not explicitly available to the agent before. By using the inverse model, activation is added to the feature sets representing *Pushed-Right* and *push-right*.

An intention is created to execute *push-right*, which further adds to the activation of the features representing that action. Then activation is provided to the *push-right* features to execute that action but only after the recurrent motor command network settles into a stable activation pattern. Perceptual input of pushing the button then increases the activation of the *Pushed-Right* features, so that the proposition is made true and the subgoal of this agent deemed achieved.

The time until motor command features settle to a stable activation pattern is an estimate of the response time and hence of the Simon effect, c.f. Haazebroek et al. (2011). Any activation on the incorrect response (*push-left* for the right-hand side subject) increases this time. In the action-conflict condition, the left-side stimulus adds activation to the feature l, which is shared with the features representing *Pushed-Left* and hence provides further activation to the features of *push-left*. In the task-conflict condition, means-end reasoning for the other subject's task via the inverse model adds onto the activation of the features representing the *push-left* motor command.

We performed a parameter estimation for this model against the empirically observed reaction times in the different conditions of the Simon task. The goal was to maximize the correlation between the *relative* reaction times observed empirically and the ones observed in our model. The term "relative" here refers to the differences of reaction times between the conditions, reflecting that the Simon effect is an increase of reaction time compared to a baseline condition (the no-conflict condition). Note that the reaction time observed in the no-conflict condition is comparable to the reaction time in all conditions without corepresentation (in the individual and social task); and the observed reaction times in the social no-conflict and action-conflict conditions are comparable to those in the individual task. Therefore the four experimental conditions presented here implicitly represent 12 conditions.

Figure 5 shows the relative reaction times obtained empirically and from the parameter set (9 parameters) that maximizes the correlation between empirical and simulated data. By all reasonable means, this match is very close. In fact, we found a large set of parameters that achieve a correlation of 0.95 or more, which we cannot show here due to space constraints. Suitable parameters cover a significant portion of the parameter space, which suggests that the model is not overly sensitive to any of its parameters.

Related Work

MOSAIC (Wolpert et al., 2003) is a computational model of motor control that relies on forward and inverse models similar to our perception-action level. In contrast to our inten-

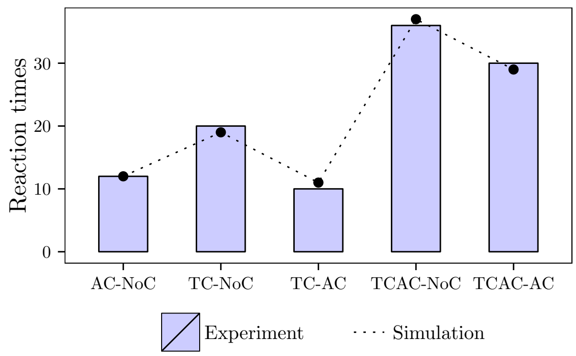

Figure 5: Relative reaction times using the parameter set that maximizes the correlation between empirical and simulated data. Simulated data points are regression-adjusted to the empirical data. Empirical data drawn from Sebanz et al. (2005).

tional level, MOSAIC explains higher-level (collaborative) planning with a hierarchy of control modules.

SCM (Hurley, 2008) is a description of motor control at an intermediary level between neural-level mechanisms and higher-level reasoning. SCM predicts how neural-level mechanisms enable higher-level ones, in particular those for joint action such as imitation and mind-reading. Our motor control models at the perception-action level borrow from SCM.

HiTEC (Haazebroek et al., 2011) is a cognitive architecture of the interplay between perception and action based on common coding theory. Representations of stimuli and action effects share the same set of features. The architecture has been used to represent the Simon task. Without intentional mechanisms, however, higher-level modulations such as social factors cannot easily be represented.

Conclusion

The cognitive mechanisms underlying joint action are not yet well understood. We describe a computational architecture of human joint action that incorporates an interplay between higher- and lower-level coordination processes and have reproduced results of four conditions of the social Simon task. Our model is consistent with the referential coding account of Dolk et al. (2014), that provides a novel approach to analyzing the Simon effect. While explorations with computational models cannot directly shed light on human cognition, c.f. Sun (2009), our demonstration contributes to analyses of potential building blocks for mechanisms involved in coordination in joint action – whether it be in purely human, or human-robot interaction contexts.

References

Bratman, M. E. (1992). Shared cooperative activity. *The Philosophical Review*, *101*, 327–341.

Dolk, T., Hommel, B., Colzato, L. S., Schütz-Bosbach, S., Prinz, W., & Liepelt, R. (2014). The joint Simon effect: a review and theoretical integration. *Frontiers in Psychology*, *5*.

Grosz, B. J., & Kraus, S. (1996). Collaborative plans for complex group action. *Artificial Intelligence*, *86*(2), 269–357.

Haazebroek, P., van Dantzig, S., & Hommel, B. (2011). A computational model of perception and action for cognitive robotics. *Cognitive Processing*, *12*(4), 355–365.

Hommel, B., Colzato, L. S., & Van Den Wildenberg, W. P. M. (2009). How social are task representations? *Psychological Science*, *20*, 794–798.

Hurley, S. (2008). The shared circuits model (SCM): How control, mirroring, and simulation can enable imitation, deliberation, and mindreading. *Behavioral and Brain Sciences*, *31*(1), 1–22.

Knoblich, G., Butterfill, S., & Sebanz, N. (2011). Psychological research on joint action: Theory and data. In B. Ross (Ed.), *The psychology of learning and motivation* (Vol. 54, pp. 59–101). Burlington, MA: Academic Press.

Pacherie, E. (2011). Framing joint action. *Review of Philosophy and Psychology*, *2*(2), 173-192.

Prinz, W. (1997). Perception and action planning. *European Journal of Cognitive Psychology*, *9*(2), 129–154.

Rao, A. S., & Georgeff, M. P. (1995). BDI agents: From theory to practice. In *Proceedings of the first international conference on multi-agent systems* (pp. 312–319).

Rumelhart, D. E., McClelland, J. L., & PDP Research Group. (1986). *Parallel distributed processing*. MIT Press.

Sebanz, N., Knoblich, G., & Prinz, W. (2005). How two share a task: Corepresenting stimulus–response mappings. *Journal of Experimental Psychology*, *31*(6), 1234–1246.

Simon, J. R., & Rudell, A. P. (1967). Auditory S-R compatibility: The effect of an irrelevant cue on information processing. *Journal of Applied Psychology*, *51*(3), 300–304.

Smith, E. R., & DeCoster, J. (2000). Dual-process models in social and cognitive psychology: Conceptual integration and links to underlying memory systems. *Personality and Social Psychology Review*, *4*(2), 108–131.

Sun, R. (2009, June). Theoretical status of computational cognitive modeling. *Cognitive Systems Research*, *10*(2), 124–140.

Tambe, M. (1997). Towards flexible teamwork. *Journal of Artificial Intelligence Research*, *7*, 83–124.

Thagard, P. (2012). Cognitive architectures. In K. Frankish & W. Ramsay (Eds.), *The Cambridge Handbook of Cognitive Science* (p. 50-70). University Press.

Tuomela, R. (2000). Collective and joint intention. *Mind & Society*, *1*(2), 39–69.

Vesper, C. (2013). *Acting together: Mechanisms of intentional coordination* (Doctoral dissertation, Radboud Universiteit Nijmegen). Retrieved from `http://hdl.handle.net/2066/112295`

Wolpert, D. M., Doya, K., & Kawato, M. (2003). A unifying computational framework for motor control and social interaction. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, *358*(1431), 593–602.