In D. Reitter & F. E. Ritter (Eds.), Proceedings of the 14th International Conference on Cognitive Modeling (ICCM 2016). University Park, PA: Penn State.

ACT-R 3D: A 3D Simulation Environment for Python ACT-R

Sterling Somers (sterling@sterlingsomers.com)
Institute of Cognitive Science, 1125 Colonel By Drive
Ottawa, ON K1S 5B6 Canada

Abstract

In this paper I describe an implementation of a time-synchronous middleware for Python ACT-R and the open-source robotics simulator, MORSE (Echeverria et al., 2012; Echeverria, Lassabe, Degroote, & Lemaignan, 2011), an updated vision system, and an updated motor system, which I collectively call ACT-R 3D. A new vision system and a crude body-model robot was added to the MORSE system to facilitate modelling of affordance-based research on aperture passage (walking through apertures and rotating shoulders as needed). Initial experimental results of shoulder rotation are presented as a proof of concept

Keywords: ACT-R; 3D; affordances; motor control; cognitive modeling;

Introduction

ACT-R is typically used to model psychology experiments in the lab. Relatively little work has been done modelling complex behavior and ACT-R natively has very limited architectural components for motor control. The aim of the project described here was to provide the capacity to model ACT-R models in dynamic, 3D environments. ACT-R 3D amalgamates the robotics simulator, MORSE (Echeverria et al., 2012, 2011), with the Python implementation of ACT-R in a time-synchronous manner. Along with ACT-R 3D is introduced a novel vision system for computer vision in 3D environments, and a motor control system to control a humanoid software robot. As a proof of concept, a model of aperture-passage affordance research is presented. The aperture-passage research has participants walk through apertures (doorways) of various sizes and then measures their degree of shoulder rotation to speculate about cognitive processing while performing these tasks (Higuchi, Seya, & Imanaka, 2012; Stefanucci & Geuss, 2010; Wagman & Malek, 2007; Warren & Whang, 1987).

Background

Although ACT-R has mainly been used for modelling human behavior in psychology experiments, there has been some attempts to model human performance in complex tasks such as driving (Salvucci, Monk, & Trafton, 2009), wayfinding (Trafton & Harrison, 2011), or piloting aircraft (Somers & West, 2013), to name a few. The present work is most similar to ACT-R Embodied (ACT-R/E) which uses ACT-R to control a robot in real-time (Trafton et al., 2012). The aim of the present work was to develop a time-synchronous ACT-R model with a tightly-controlled motor module, sufficient for performing motor control for affordance-based research. In particular, the author used ACT-R 3D to model aperture-passage work by Warren and Whang (Warren & Whang, 1987), where they found that participants rotate relative to the ratio between the width of the aperture and their frontal body width (i.e. shoulder width).

ACT-R/E (Trafton & Harrison, 2011) uses ACT-R as a robot controller and uses the visual system, SECS (Harrison & Schunn, 2003). SECS has three main systems: *visual*, *manipulative*, and *configural.* The visual system uses fiducial and face trackers to provide object identification for video camera images. The manipulative system represents objects as 3D geons (stored in a database) as well as positions and orientation information. The manipulative system also supports spatial transformations, such as rotations in a manner similar to that proposed by Shepard and Metzler (Shepard & Metzler, 1971), supporting motor planning.

ACT-R/E also extends the basic ACT-R motor system. Although somewhat vague in description, Trafton and Harrison (2011) suggests that the motor system maintains real-time limb representations and restricts movements based on muscle groups. It is also suggested by Trafton et al. (Trafton, Harrison, Fransen, & Bugajska, 2009) and by Harrison and Trafton (2010) that motor control is handled external to the ACT-R architecture once ACT-R selects a motor command. For example, given the description by Harrison and Trafton (Harrison & Trafton, 2010), once a representation in the manipulative module is associated with the objects semantic representation, the central production system in ACT-R issues an appropriate grasping command to the robot controller, which carries out the grasp. ACT-R/E has been used in a variety of models. Harrison and Trafton (2010) used ACT-R/E to model response times of grasp actions, Trafton and Harrison (2011) used it to model gaze-following and level-one perspective taking, and SECS (the spatial representation system used in ACT-R/E) was used to model an egocentric navigation task (Harrison & Schunn, 2003).

The present work is a novel implementation comprised of an updated version of Python ACT-R (Python 3), Mobile OpenRobots Simulation Engine (MORSE) (Echeverria et al., 2012, 2011), and a custom middleware responsible for time synchrony and communication between ACT-R and MORSE. The remainder of this paper will describe the system, and a proof-of-concept experimental task (as well as accompanying initial results).

System

The following section describes MORSE and ACT-R 3D.

MORSE

MORSE simulator is a robotics simulator based in Blender[1] (a free, open-source 3D creation studio). MORSE comes with a number of standard sensors and actuators that are pairable to a number of robotic bases. Custom environments, robots, and sensors can be developed in Blender and incorporated into simulations. MORSE is written in Python 3 and the Python library supports full control of robots through Python script. Communication between the robot, sensor, actuators; and the control program is handled through sockets. Models built for ACT-R 3D function as a simulation control script, albeit complex ones. At the time of development, MORSE was at version 1.2. The current version (version 1.4) has a number of advances but should be largely supported by ACT-R 3D.

ACT-R 3D

Python ACT-R Python ACT-R is a re-implementation of ACT-R for the Python programming language (Stewart & West, 2005). Because of the constraints of scripting with MORSE, Python ACT-R was updated to Python 3 for this project. The overall structure and design is largely the same as originally described.

Middleware The middleware between MORSE and ACT-R is designed primarily to support time-synchrony between the ACT-R simulation loop and MORSE. The design of the time-synchrony is inspired by Somers and West (2013), who, as part of a larger project, created a middleware between a popular flight simulator and Python ACT-R. A second major component of the middleware is that it facilitates communication between the ACT-R and MORSE.

In the current implementation, the middleware is designed to run both MORSE and ACT-R at 100 simulated-*Hz,* with no constraints towards real-time simulation. Although a redundant real-time mode is available, in general this project differs from previous work such as ACT-R/E (Trafton & Harrison, 2011) which control robots in real-time. The reasoning behind this is that timing is critical for prediction. Since timing is undoubtably one of the biggest behavioural measures, it is important that timing is as accurate as possible. To avoid any time delays for processing of complex information, that might occur in a computer vision system, or simply even communication time delays, MORSE/ACT-R middleware works in a *tick-tock* fashion. First the ACT-R system *ticks* 10 *ms* of simulation time, sends information get and set requests to the middleware, then, in turn, runs a corresponding 10 *ms tock* in MORSE. At the end of a tick-tock cycle, 10 *ms* of simulation time has elapsed on both simulators. Importantly, any amount of real-time could have elapsed during the tick-tock cycle. The aim is not to have an efficient robot controller but to facilitate prediction of behavioural measures.

Geometric Camera A custom camera class, Geometric Camera, was developed, for MORSE, for the purposes of this project. The intention behind the camera is to provide a single, structured, retinotopic description of the scene from the perspective of the agent.

The camera outputs a dictionary description of the entire scene as visible within its field of view. The dictionary is primarily organized by screen-coordinate *y*-values (where *y* is the vertical plane). Each *y*-value is organized by object label for each object visible to the camera. These labels are not semantically informative; they simply specify different 3D objects in the scene. To each label is associated the object's extension in *x*-values (denoting the object's edges at that *y-value*), and an approximate egocentric distance to the edge. The accuracy of the camera can be passed as parameters, though there can be significant speed/accuracy trade-offs. The overall output is, therefore, a dictionary that describes the egocentric edges of every object visible to the camera. That information is stored privately on the ACT-R side and only accessible through the vision module. Modelers are free to develop their own vision modules to access the visual information to suit their needs.

Vision Module The vision module developed for this project assumes an input from the Geometric Camera (described above). The control of timing, error rates, etc., happen entirely on the ACT-R side.

The agent can only access visual information through requests. However, because the data is largely unstructured, methods for detecting 'features' relevant to the project were developed. A simple *obstacle detector* provides the location of any visible obstacles. Because the task is to walk through a doorway-like aperture an *openings detector* was also developed. The openings detector finds features like doors, holes in walls, or openings between multiple objects. The output of the *openings* detector is not semantically laden. That is, the agent will not be aware of whether the opening is as large as a doorway or as small as a nail hole. The agent model must determine a means of attending to appropriate features. One of the main goals of modelling the aperture-passage task, is that the agent is not semantically informed about its environment. Absolutely no semantic labels are used in the vision system.

The vision module is based on the SOS Vision System (West & Emond, 2002, 2005) in Python ACT-R. Although it is far more complex in the terms of the type of data it deals with, fundamentally the information requests work the same. Requests to the vision system are parameterized in order to filter information. For example, when given a request for an *opening*, chunks that describe the minimum size of the opening are used as parameters for the request. If multiple features match the request (e.g. there are multiple openings), the returned chunk is selected based on a weighted random choice, weighted by a *salience* factor, as described by West and Emond, (2002).

[1] https://www.blender.org/

Motor Module The motor module is one of the features of ACT-R 3D that sets it apart from ACT-R/E (Trafton & Harrison, 2011). While perhaps similar at a functional level (controls limbs, restricts movement), the implementation is novel.

The motor module maintains a hierarchical, symbolic and numerical representation of the body parts (currently only the ones being modelled), that is synchronized with the 3D body-model (robot). For each body part, there are representations of their degrees of freedom. At present the degrees of freedom are represented as max/min values on an axis of rotation. The motor module maintains an internal representation of the maximum and minimum rotations performed by an agent. As the agent moves its body, the maximum and minimum achieved rotations are updated and stored in declarative memory. At a functional level, this memory of body postures can be considered an implementation of body-schema that are theorized to be used in motor planning (Coslett, Buxbaum, & Schwoebel, 2008; J Schwoebel, Coslett, & Buxbaum, 2001; John Schwoebel & Coslett, 2005).

The motor module in ACT-R 3D also includes functionality to provide proprioceptive feedback to estimate 3D body dimensions in a given posture. Evidence for this capability comes from a number of sources (Carello & Turvey, 2004; Stefanucci & Geuss, 2010; Wagman & Taylor, 2005; Warren & Whang, 1987), though anecdotally this ability is intuitively obvious: one does not try to fit their body in a 1 cm^2 hole. We can, at the very least, make crude judgments of body volume, and the above cited suggest the volumetric representations are fairly accurate. How exactly the volumetric representations are achieved are currently beyond the scope of motor module and is implemented, quite simply, with a measure of the agent's bounding box. The bounding box values are associated and stored with the body schema at the time of storage into declarative memory.

Evaluation

ACT-R 3D can be evaluated on its ability to model behavior and make useful predictions of that behavior. This section outlines some preliminary work modelling aperture-passage affordance research.

Aperture-Passage Affordance

The term 'affordance' was first used as a noun by Gibson when he presented an approach to Psychology, Ecological Psychology (1986). An affordance can be thought of as a property (or a set of properties) of the environment[2] that an agent uses in order to determine what actions are available. Action, of course, is a broad category and there is empirical research in support of affordances in a large number of domains including grasping (Tucker & Ellis, 1998), reaching (Carello, Grosofsky, Reichel, Solomon, & Turvey, 1989), stair-climbing (Warren, 1984), a number of sports abilities (see Fajen, Riley, & Turvey, 2009), and relevant here, aperture passage (Warren & Whang, 1987). One of the central concepts in affordance research is Gibson's notion of *direct perception*.

Direct perception is the claim that our actions are not mediated by strong, internal, sensory-based, semantically-laden representations of the environment. Instead, direct perception holds that actions are presented to us in the environment when they are in agreement with our action capabilities. For example, a cup is graspable because its shape and size agrees with the action capabilities of our hands. Realizing an action is a perceptual process and actions are said to be directly perceived.

A common theme in empirical ecological psychology research is to identify a body-scaled unit that is used by the perceptual system to perceive an affordance directly. Researchers identify *pi*-numbers (π) that relate some dimension of the environment (E) with some dimension of the body (A), as a ratio:

$$\pi = E / A.$$

The present work is a first attempt at modelling the classic affordance-passage work by Warren and Whang (1987). Warren and Whang performed a series of experiments aimed to show that aperture passage is directly perceived. In the experiment relevant here, they had participants walk through apertures of different widths, at different speeds (*fast* or *slow*). Participants were grouped according to size: *small* or *large*. Unsurprisingly, smaller agents rotated their shoulders less than large agents when passing through the same aperture. However, when expressed as an aperture-width to shoulder-width ratio (*A/S*), the group differences were eliminated, suggesting that the absolute degree of shoulder rotation is modulated by the *A/S* ratio. Warren and Whang were able to establish a *critical ratio* (1.3) at which, regardless of size, participants would change from maintaining a forward posture, to a posture that included shoulder rotation.

Model

The aperture-passage agent is inspired by work on steering control in a flight-simulator in ACT-R (Somers & West, 2013) in that it uses the SGOMS (West & Nagy, 2007) modelling framework for modelling complex behavior and both bottom-up and top-down visual modules. The simulation environment reflects the experimental setup in Warren and Whang (1987) and is illustrated in Figure 1.

The agent is illustrated in Figure 2. The agent is a low-fidelity humanoid. The robot consists of a rectangular cuboid base mesh with a single armature that, in turn, consists of: a rectangular cuboid pelvic region, a rectangular cuboid torso, a spherical joint between the pelvic region and the torso (not visible) two rectangular cuboid shoulder, two spherical shoulder joints, two cylindrical upper arm segments, a spherical neck joint, a spherical head, and the Geometric Camera. Figure 2 is a polygon reduced version for quicker graphics processing. Each segment of the armature has full degrees of freedom. Although Blender

[2] Their ontological status varies, depending on the author.

does support inverse kinematic solvers for armature control, there are no kinematic solvers associated with any of the armatures in this project. All joint limb control and position representation is controlled by ACT-R 3D, synchronously, as described above. Collision sensors are placed on the shoulder joint, the shoulders, the upper arm segments, and the torso.

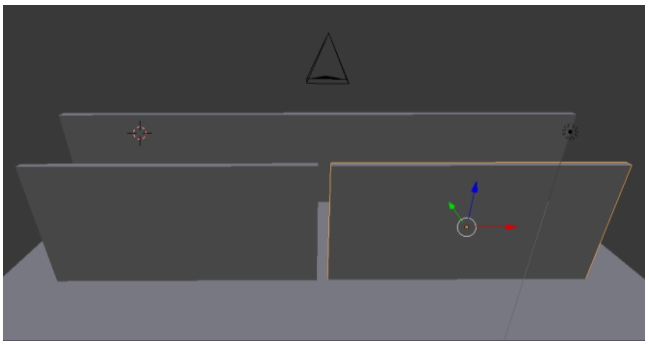

Figure 1: A Snapshot of the modelling environment. Two wall segments create an aperture. The wall segments are moved to create apertures of different sizes.

.

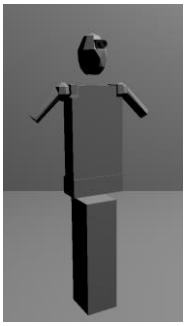

Figure 2: A snapshot of the software robot used in this project. The bottom cuboid represents the agent's legs.

Vision Following the work of Somers and West (2013) a bottom-up vision system was implemented that loops constantly, in this case, looking for obstacles. The bottom-up vision system shares a buffer with the top-down vision system and both also share the main vision module. The buttom-up vision system is implemented using a production system[3] and loops continuously, extracting information from the environment. The bottom-up vision system is constrained by the top-down vision system which is controlled via the central production system. Top-down vision takes priority over the vision module and, as a result, the bottom-up vision system must wait until any top-down requests are complete before it can continue.

[3] Note that Python ACT-R allows for production in modules and this should be thought of as an alternative way of programming modules, not a departure from standard ACT-R.

SGOMS Since an ACT-R implementation of SGOMS has been detailed in (Somers & West, 2013), only a limited description will be provided here. SGOMS --Socio-technical GOMS (Card, Moran, & Newell, 1983)—is a framework for modeling complex tasks and can be implemented in ACT-R. In its ACT-R implementation it uses a hierarchy of buffers that represent the levels of control in a complex task. The hierarchy consists of a planning unit (highest, slowest level of control), the unit task, operators (realized as productions), and methods (compiled productions). The model presented here uses SGOMS to facilitate task interruption.

Design The model can perhaps be best described as going through four phases. The first phase is actually a pre-experiment phase in which the agent stores information about its body size in different postures. Essentially, in this first phase, the agent rotates its shoulders in each direction multiple times, adding the body-schema to declarative memory.

During the second phase, the agent decides whether it can pass through the aperture at all. This is achieved with the vision system. As described above, the *opening* detector is takes dimensions as parameters and filters openings that are too small to accommodate. There are potentially two steps to this phase. The first step involves judging the passability of the opening relative to the agent's present (at the beginning of the simulation) posture. If that fails, a second step is taken. The agent then attempts to recall a body-schema that meets the constraints of the task (e.g. affords walking, involves rotating the shoulders). If a body-schema is recalled (in this case, a posture with complete rotation either to the left or right should be selected), then the dimensions of the agent in that posture is used as a parameter to the visual system's *openings* detector. If there is a match, that means that at maximum rotation, the agent could pass through the aperture. The agent uses the body-schema as a goal for the motor module at time of passage.

During the third phase the agent simply walks towards the aperture. It is assumed that there is very little top-down vision happening. Instead, the bottom-up vision system is cycling for obstacle detection. This phase continues until the agent detects one or both of the walls on either side of the aperture. The obstacle detection causes a task interruption using the SGOMS framework in a similar manner to that described in (Somers & West, 2013). The obstacle alert ultimately results in the fourth phase, rotation control.

Since no data on the kinematic control of rotation is available in the literature, an instantaneous and constant rotation velocity is assumed. Rotation continues until the vision system detects an opening larger than the agent's frontal width (as the agent rotates its shoulders, its frontal width reduces to a point where it is smaller than the width of the opening, resulting in the opening being detected).

Simulations The experimental conditions (including participant sizes) were created as accurately as possible

based of the description by Warren and Whang (1987). There were 5 agents per group condition (*large, small*) and agent sizes were chosen randomly from a normal distribution around the reported (human) means for each group (40.4 *cm, SD* = 2.0 *cm* for small; and 48.4 *cm*, SD = 0.7 *cm* for large). Agents walked at the average speed per group as reported in Warren and Whang: 1.29 *m/s* and 1.61 *m/s* for the normal and fast conditions (respectively) in the small group; and 1.28 *m/s* and 1.77 *m/s* for the normal and fast conditions (respectively) in the large group. Each software participant did 60 trials in each condition, for each aperture.

Because the original data from Warren and Whang's original paper was not available, only a qualitative visual comparison is presented. A visual comparison between the human data in Warren and Whang (1987) is presented in Figures 3 (normal speed) and Figure 4 (fast speed).

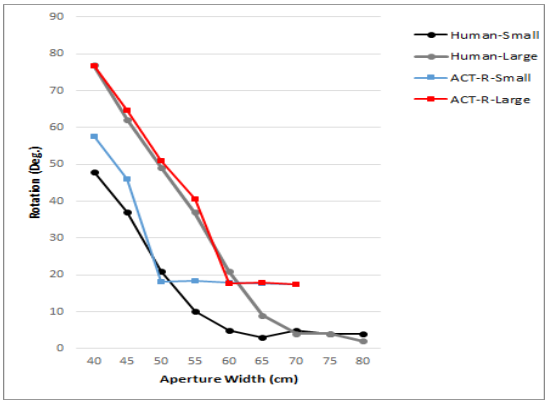

Figure 3: Rotation angle by aperture width for small (blue) and large (red) agents. Gray lines represent human data. Data is from the 'normal' speed condition.

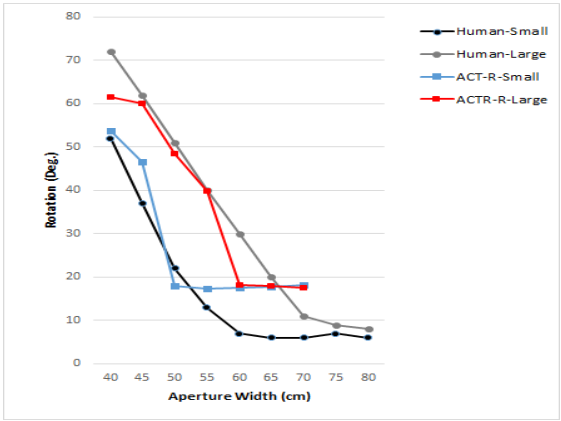

Figure 4: Rotation angle by aperture width for small (blue) and large (red) agents. Gray lines represent human data. Data is from the 'fast' speed condition.

In lieu of comparative statistics, an ANOVA was ran on the model data to see if the main effects described in Warren and Whang's original experiment were replicated. Just as for the human data, participants rotate more for smaller apertures (main effect of aperture), larger participants rotated more than smaller participants (main effect of group) (ps < 0.01). The effect of speed, however, had the opposite effect relative to the findings in Warren and Whang such that the model rotated less when walking at a faster pace (also main effect, $p < 0.01$). However, as illustrated in Figure 5, the group difference in the speed condition is dominated by a single aperture (40 *cm*).

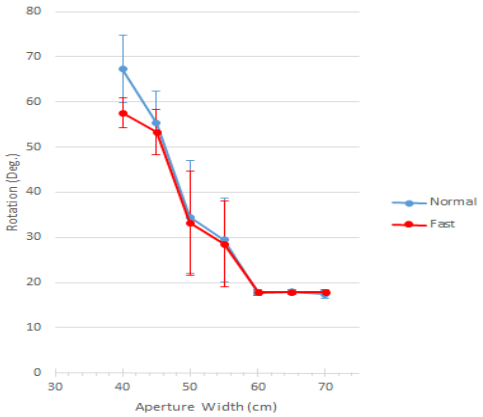

Figure 5: Rotation angle by aperture width for the normal (blue) and fast (red) speed conditions. Error bars represent 95% confidence intervals.

Discussion

ACT-R 3D is designed to support research that involves complex, dynamic environments. Although the model of aperture-passage as presented here has many open questions (some of which have been answered in further analyses, forthcoming), the model, even in its early stages, has offered strong insight to the domain of aperture-passage (though, admittedly it raises more empirical questions than it answers). Forthcoming work will see the model performing similar experiments. Currently it also models aperture passage while carrying an object (Higuchi et al., 2012). Also under development with ACT-R 3D is an embedded driving model, that uses the geometric camera embedded in a simulated car, to drive around a track. Because it uses the Blender simulation engine, a large variety of simulation can potentially be developed in ACT-R 3D for applied cognitive research.

References

Card, S. K., Moran, T. P., & Newell, A. (1983). The keystroke level model for user performance with interactive systems. *Communications of the ACM1*, *23*, 396–410.

Carello, C., Grosofsky, A., Reichel, F. D., Solomon, H. Y., & Turvey, M. T. (1989). Visually Perceiving What is Reachable. *Ecological Psychology*, *1*(1), 27–54. http://doi.org/10.1207/s15326969eco0101_3

Carello, C., & Turvey, M. T. (2004). Physics and Psychology of the Muscle Sense. *Current Directions*

in Psychological Science, *13*(1), 25–28. http://doi.org/10.1111/j.0963-7214.2004.01301007.x

Coslett, H. B., Buxbaum, L. J., & Schwoebel, J. (2008). Accurate reaching after active but not passive movements of the hand: Evidence for forward modeling. *Behavioural Neurology*, *19*(3), 117–125. http://doi.org/10.1155/2008/972542

Echeverria, G., Lassabe, N., Degroote, A., & Lemaignan, S. (2011). Modular open robots simulation engine: MORSE. In *2011 IEEE International Conference on Robotics and Automation* (pp. 46–51). IEEE. http://doi.org/10.1109/ICRA.2011.5980252

Echeverria, G., Lemaignan, S., Lacroix, S., Karg, M., Koch, P., Lesire, C., & Stinckwich, S. (2012). Simulating Complex Robotic Scenarios with MORSE. In *SIMPAR* (pp. 197–208).

Fajen, B. R., Riley, M. A., & Turvey, M. T. (2009). Information, affordances, and the control of action in sport. *International Journal of Sport Psychology*, *40*(1), 79–107.

Gibson, J. J. (1986). *The ecological approach to visual perception*. Hillsdale, NJ: Erlb.

Harrison, A. M., & Schunn, C. (2003). ACT-R/S: Look Ma, no "cognitive map"! In *Proceedings of the Fifth International Conference on Cognitive Modeling* (pp. 129–134). Bamberg, Germany: Universitats-Verlag Bamberg.

Harrison, A. M., & Trafton, J. G. (2010). Cognition for action: an architectural account for "grounded interaction." *Proceedings of the 32nd Annual Conference of the Cognitive Science Society (CogSci 2010)*, 200–205.

Higuchi, T., Seya, Y., & Imanaka, K. (2012). Rule for Scaling Shoulder Rotation Angles while Walking through Apertures. *PLoS ONE*, *7*(10), 1–8. http://doi.org/10.1371/journal.pone.0048123

Salvucci, D. D., Monk, C. A., & Trafton, J. G. (2009). A Process-Model Account of Task Interruption and Resumption: When Does Encoding of the Problem State Occur? In *Proceedings of the Human Factors and Ergonomics Society 53rd Annual Meeting* (pp. 799–803). Santa Monica, CA: Human Factors and Ergonomics Society. Retrieved from http://pro.sagepub.com/content/53/12/799.short

Schwoebel, J., & Coslett, H. B. (2005). Evidence for Multiple, Distinct Representations of the Human Body. *Cognitive Neuroscience*, *17*(4), 543–553.

Schwoebel, J., Coslett, H. B., & Buxbaum, L. J. (2001). Compensatory coding of body part location in autotopagnosia: Evidence for extrinsic egocentric coding. *Cognitive Neuropsychology*, *18*(4), 363–381.

Shepard, R. N., & Metzler, J. (1971). Mental Rotation of Three-Dimensional Objects. *Science*, *171*(3972), 701–703. http://doi.org/10.1126/science.171.3972.701

Somers, S., & West, R. (2013). Steering Control in a Flight Simulator Using ACT-R. In R. L. West & T. C. Stewart (Eds.), *Proceedings of the 12th International Conference on Cognitive MOdelling*. Ottawa: Carleton University.

Stefanucci, J. K., & Geuss, M. N. (2010). Duck! Scaling the height of a horizontal barrier to body height. *Attention, Perception & Psychophysics*, *72*(5), 1338–49. http://doi.org/10.3758/APP.72.5.1338

Stewart, T. C., & West, R. L. (2005). Python ACT-R: A New Implementation and a New Syntax. In *12th Annual ACT-R Workshop*.

Trafton, J. G., & Harrison, A. M. (2011). Embodied Spatial Cognition. *Topics in Cognitive Science*, *3*(4), 686–706. http://doi.org/10.1111/j.1756-8765.2011.01158.x

Trafton, J. G., Harrison, A. M., Fransen, B. R., & Bugajska, M. D. (2009). An embodied model of infant gaze-following. *International Conference of Cognitive Modeling*, (2003).

Trafton, J. G., Hiatt, L. M., Harrison, A. M., Tamborello II, F. P., Khemlani, S. S., & Schultz, A. C. (2012). ACT-R/E: An embodied cognitive architecture for Human-Robot Interaction. *Journal of Human-Robot Interaction*, *1*(1), 78–95.

Tucker, M., & Ellis, R. (1998). On the relations between seen objects and components of potential actions. *Journal of Experimental Psychology. Human Perception and Performance*, *24*(3), 830–46.

Wagman, J. B., & Malek, E. A. (2007). Perception of Whether an Object Can Be Carried Through an Aperture Depends on Anticipated Speed. *Experimental Psychology (formerly "Zeitschrift Für Experimentelle Psychologie")*, *54*(1), 54–61. http://doi.org/10.1027/1618-3169.54.1.54

Wagman, J. B., & Taylor, K. R. (2005). Perceiving Affordances for Aperture Crossing for the Person-Plus-Object System. *Ecological Psychology*, *17*(2), 105–130. http://doi.org/10.1207/s15326969eco1702_3

Warren, W. H. (1984). Perceiving affordances: visual guidance of stair climbing. *Journal of Experimental Psychology. Human Perception and Performance*, *10*(5), 683–703.

Warren, W. H., & Whang, S. (1987). Visual guidance of walking through apertures: body-scaled information for affordances. *Journal of Experimental Psychology. Human Perception and Performance*, *13*(3), 371–83.

West, R. L., & Emond, B. (2002). SOS: A Simple Operating System for modeling HCI with ACT-R. In *Seventh Annual ACT-R Workship*. Pittsburg, PA.

West, R. L., & Emond, B. (2005). The Environment as Theory : An Example Using the ACT-R / SOS Environment. *Proceedings of the Sixth International Conference on Cognitive Modeling*, 398–399.

West, R. L., & Nagy, G. (2007). Using GOMS for Modeling Routine Tasks Within Complex Sociotechnical Systems : Connecting Macrocognitive Models to Microcognition. *Joural of Cognitive Engineering and Decision Making*, *1*(2), 186–211. http://doi.org/10.1518/155534307X232848.