FORMS: A Theory of Human Spatial Cognition Implemented in the ACT-R Cognitive Architecture

Franklin P. Tamborello and Hongbin Wang

School of Biomedical Informatics

University of Texas Health Science Center at Houston

7000 Fannin, Ste. 600

Houston, Texas, USA

Version 1.0

Last Modified 2010.10.01Table of Contents

1. Overview

The FORMS spatial module for ACT-R 6 is an implementation of the FORMS (Frame Of Reference-based Maps of Salience) theory of human spatial cognition (Wang, Johnson & Zhang, 2001) within the framework of a unified theory of human cognition, ACT-R (Anderson et al., 2004). FORMS stresses two aspects of spatial cognition: space as experienced from multiple frames of reference and selection among those frames of reference (FORs) and locations within those FORs by salience. The module takes as its input visual representations from ACT-R and parses them into symbolic, primitive bearing and distance representations. It also remaps spatial representations (SRs) from the egocentric FOR (EFOR) to be in terms intrinsic to an exogenous anchor object (the intrinsic FOR, or "IFOR"). The spatial module then makes these symbolic representations of spatial information available to the rest of cognition as theorized and instantiated by ACT-R.

Crucial to FORMS' success as a theory of human spatial cognition is the hand-in-hand interaction of the specifically-spatial part, FORMS itself, with the rest of cognition, which is what ACT-R provides. No account of spatial cognition can be complete without both components working together. Using a computational cognitive architecture is very important for reasons that have been enumerated elsewhere (Anderson et al., 2004; Byrne, 2001; Byrne & Kirlik, 2005; Gunzelmann, Anderson & Douglass, 2004; Newell, 1973), but the multi-FOR aspect of spatial reasoning has not heretofore been addressed. Using this approach we can elucidate the ways in which multiple FOR representations interact with each other and with other cognitive mechanisms to produce the emergent spatial behavior. Previous studies of spatial reasoning have addressed the role that FORs play, but not much has been done in accounts that integrate across FORs, a prerequisite for complex spatial reasoning such as might be the case in many tasks ranging from hide-and-go-seek to local and remote vehicle operation and even combat. Building a useful theory of multi-FOR spatial reasoning behavior within the constraints of a cognitive architecture will likely benefit applications in domains that draw upon that kind of reasoning, such as in the tasks just named. Eventually the computational nature of such an account built within an architecture like ACT-R can be useful as high-fidelity cognitive modeling (Gray, Schoelles & Myers, 2004).

2. Theory: Spatial Representational Structure

The spatial module for ACT-R emphasizes the representational role played by multiple FORs in spatial reasoning. The module acts as both as a parsing mechanism, parsing low-level perceptual information into primitive symbolic spatial representations (SRs)—egocentric distance and bearing—and using those primitive symbolic representations to construct more complex SRs, and as extended working memory representation capacity specifically for those SRs. However, because the FORMS spatial module posits multiple SRs made available to a generalized cognitive system that is fundamentally constrained in the amount of information it may process at any given time (the working memory bottleneck; Anderson, et al, 2004), implicit in this account is a notion of selective processing of SRs according to salience. SRs more relevant to the goals and processes of the model get processed while others may be ignored.

2.1 Ontology of FORMS

FORMS has a dual-nature representational structure, depicted in Figure 1. The two natures stem from the two basic types of representations it creates, “spatial objects” (hereafter SOs) and several different types of spatial representations (SRs). Although the emphasis of the module is on location and not object, it is necessary to interface with object representations so that they may be bound with location representations. The module’s representational structure accomplishes this by having a spatial object chunk for every location represented. The locations are represented by SR (SR) chunks, each of which has two references to spatial object (SO) chunks. One SO reference encodes a viewpoint, a representation of the anchor of the SR’s FOR. These FOR anchor SOs may be primitive representations hard coded into the module, such as the observer’s eye, or they may be other objects the perspective of which is being represented. For example, if the observer wants to represent a bearing of “right” relative to another person, the viewpoint SO for this SR would be the other person.

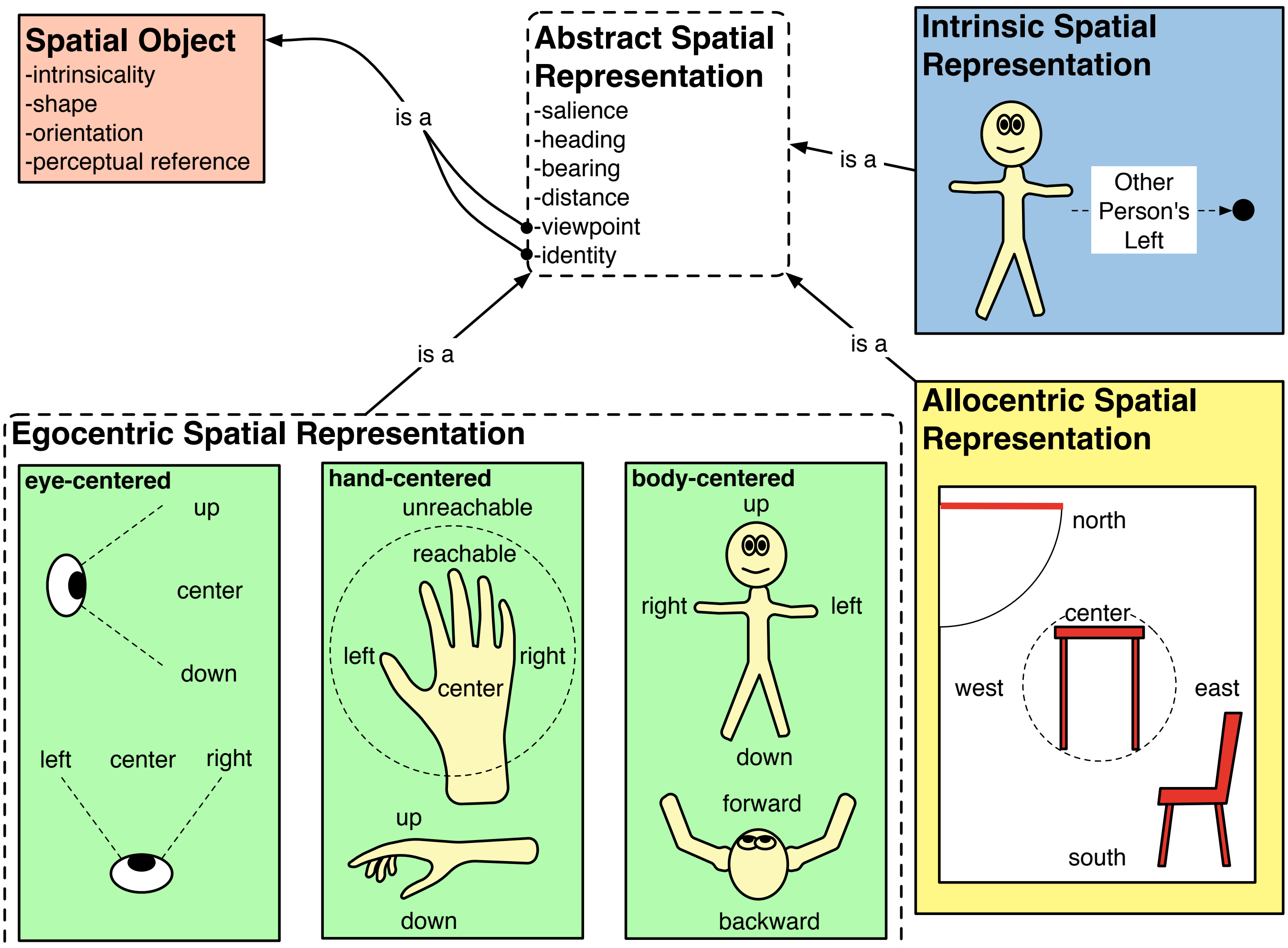

Figure 1. An ontology for spatial cognitive representation.

2.1.1. Spatial Object (SO)

In FORMS’ ontology spatial objects may have the following properties:

Intrinsicality (true or false): Intrinsicality means that the SO provides sufficient spatial cues for a frame of reference to be established with a perspective based on this object. The object therefore has a polar coordinate pole located within itself, with the axis lying along the SO’s orientation. We assume this spatial affordance is a property of the object that becomes available with object recognition, a cognitive process that is beyond the scope of this particular spatial cognition project.

Shape: a description of the spatial object's physical configuration, such as "rectangle" or "arrow-shape". This may have implications for spatial reasoning affordances, such as if this SO possesses intrinsicality or if this SO is a navigable place.

Orientation: spatial orientation of the object as it appears to the observer within the egocentric FOR, as it is recognized by visual encoding

Perceptual-Reference: The percepts used to construct this spatial object. This provides a way for linking back from the spatial representations to the perceptual representations used to form them.

2.1.2. Spatial Representation (SR)

2.1.2.1. Abstract Spatial Representation Chunk Types

The abstract SR is a generic way to represent a position or direction in space. It is not actually instantiated as representations but instead it is a generic way to refer to any SR by serving as a super-type to the three SR types used by FORMS to represent and reason about space. The abstract SR's properties usually mean different things for each SR type because of their differing FORs. These are the properties of the abstract SR chunk-type and, in a general sense, what they refer to:

Salience: This refers to a relative activation of the SR from top-down & bottom-up sources. This is a dynamic quantity, changing as SRs are updated to reflect environmental changes and changes in attention and goals.

Heading: orientation of the SR's polar axis relative to its viewpoint FOR's polar axis

Bearing: location of the SR relative to the viewpoint FOR's pole

Distance: distance of this SR from its viewpoint

Viewpoint: a SO, the object having the origin or axis upon which the SR's FOR is based—for example, the observer's eye or an imagined location

Identity: the spatial object the location of which is represented by this SR

2.1.2.2. Eye-Centered SR

The eye-centered SR is a subtype of the abstract SR. Here the default viewpoint is the observer's eye, a primitive SO in FORMS. FORMS constructs all of this type of SR to be in the egocentric FOR and specifically relative to the eye.

Salience: same as generic definition

Heading: nil or the object's intrinsic-reference direction relative to the eye's intrinsic reference direction, "up" in two dimensional reasoning

Bearing: the position of the object's SR (this SR) relative to the viewpoint's orientation

Distance: distance to the location

Viewpoint: the eye (atomic)

Identity: the SO represented by this SR

2.1.2.3. Allocentric SR Chunk Type

Salience: same as generic definition

Heading: nil or the directional offset between the represented object's recognized intrinsic orientation and the viewpoint's heading

Bearing: nil or some direction from the viewpoint's intrinsic reference

Distance: distance to the location

Viewpoint: eye, hand, body, or an imagined SO (such as a top-down-north-up view of a city map)

Identity: the spatial object represented by this SR

2.1.2.4. Intrinsic SR Chunk Type

Salience: same as generic definition

Heading: the identified SO's intrinsic orientation relative to the observer or relative to another reference, such as a cardinal allocentric direction (e.g., north)

Bearing: the represented object's bearing from the IFOR anchor

Distance: distance to the IFOR anchor

Viewpoint: the IFOR anchor's spatial object

Identity: the spatial object represented by this SR

3. End-User Reference: Walkthrough of Spatial Module Functioning

This document assumes of the reader some familiarity with the ACT-R cognitive architecture, such as its nature as a production system and its use of chunks to represent knowledge. Readers are referred to Anderson et al (2004) for discussion of the ACT-R theory and to the ACT-R Reference Manual (Bothell, 2010) for details about the implementation of ACT-R theory as a software package.

3.1. Loading and Running

The FORMS module comes in one Common Lisp file, "spatial.lisp" (version 1.0). Place this file inside your ACT-R installation's "Modules" directory prior to loading ACT-R to have it automatically load with the rest of your ACT-R installation (Bothell, 2010), or load the file yourself between loading ACT-R and loading your model. Once the FORMS module is loaded it works like any other ACT-R cognitive module; it has parameters that can be got and set with ACT-R's sgp command and it has buffers with which it can communicate with other modules as a model runs.

Use the module's spatial-object buffer to make requests of the module to generate SRs with values passed in the request taken from visual or visual-location chunks. The module will create a spatial-object chunk from the request and then automatically create eye-centered EFOR and screen-based AFOR representations, as in this example production:

```
(p got-non-intrinsic-visual-make-spatial
  =visual>
     isa visual-object
    value =val
  ?spatial-object>
     buffer empty
  =goal>
     isa spatial-task
```

```
      state attending-location
  ==>
    +spatial-object>
       isa spatial-object
      shape =val
      perceptual-reference =visual
    =goal>
       state making-spatial-representations)
```

The resultant chunks are then available for harvesting from the spatial-object, eye-central, allocentral, and intrinsic buffers. It is worth noting that the module implements an all-at-once kind of functionality for both chunk creation and clearing. This means that whenever one buffer is requested the module automatically creates chunks for all of the buffers that it can. It may not always be able to create intrinsic-spatial-representation chunks, more on that later. Similarly, whenever one of the FORMS module's buffers is cleared, all of the others are also cleared.

3.2. Implementation of the FORMS Ontology in ACT-R 6

Currently implemented in the FORMS module are symbolic representations of eye-centered egocentric, allocentric, and intrinsic FORs. These representations are each implemented as chunk types which the module generates from visual perceptual information encoded by ACT-R's vision module and passed to the spatial module in requests made to it. The module uses a parsing mechanism that converts ACT-R's visual screen coordinates into relativistic, regionally-encoded bearings such as "up" or "right." The module then creates a SR chunk encoding the symbolic bearing of the object relative to each FOR's intrinsic orientation, such as the eye.

The FORMS module has a buffer for each type of representation which makes that representation's chunk available to the rest of the ACT-R model.

SRs have some salience based on the interactions of ACT-R's modules, with the model's current state dictating which SRs get processed. Together, the contents of all of ACT-R's buffers account for all of the information available to procedural memory, the pattern-matching action selection mechanism of cognition. That means that the collective contents of all of ACT-R's buffers in effect comprise working memory (Anderson, 2007), and so the contents of the spatial module make those SRs salient in the sense that they are the ones that are currently available to cognition. Other cognitive processes, such as declarative memory retrieval, can make SRs stored in long-term memory salient by the means of declarative memory retrieval, discussed elsewhere (Anderson, 2007; Anderson et al., 2004; Bothell, 2010).

3.3. Spatial Chunk-Types

There are two chunks of type spatial object which serve as representational primitives in the module: "eye-so" and "screen-so." These are normal SO chunks except that the module includes their definitions and loads them during the execution of its reset function. Eye-SO provides a reference bearing, "up," which makes it possible to remap egocentric SRs to the IFOR. Screen-SO performs the same function for the allocentric FOR.

```
eye-so
    isa spatial-object
    intrinsicality t
    shape nil
    orientation up

screen-so
    isa spatial-object
    intrinsicality t
    shape rectangle
    orientation up
```

As visual object recognition is beyond the scope of the FORMS module effort, we abstracted the process by which people recognize that a given object has or does not have properties necessary for the creation of intrinsic FORs, namely, a definitive spatial orientation. We did this by creating a subtype of ACT-R's visual-object chunk-type to include the intrinsicality value that the spatial-object chunk-type uses. We assume that intrinsicality recognition is a process of visual object recognition. And so if the model has an intrinsic-visual-object chunk available to it then it can pass the value of the intrinsicality slot of the intrinsic-visual-object chunk to the spatial-object buffer, as in:

```
(p got-intrinsic-visual-make-spatial
  =visual>
     isa intrinsic-visual-object
    value =val
    intrinsicality =int
     orientation =ori
  ?spatial-object>
     buffer empty
  =goal>
     isa spatial-task
    state attending-location
==>
  +spatial-object>
     isa spatial-object
    shape =val
    intrinsicality =int
     orientation =ori
    perceptual-reference =visual
  =goal>
     state making-spatial-representations)
```

Such a production may result in SR chunks being placed such as:

```
EYE-CENTERED-SPATIAL-REPRESENTATION2-0
 ISA EYE-CENTERED-SPATIAL-REPRESENTATION
  SALIENCE NIL
  HEADING NIL
  BEARING LEFT
  DISTANCE 15.0
  VIEWPOINT EYE-SO
  IDENTITY SPATIAL-OBJECT2-0

ALLOCENTRIC-SPATIAL-REPRESENTATION0
 ISA ALLOCENTRIC-SPATIAL-REPRESENTATION
  SALIENCE NIL
  HEADING DOWN
  BEARING CENTER
  DISTANCE 15.0
  VIEWPOINT SCREEN-SO-0
  IDENTITY SPATIAL-OBJECT0-0

INTRINSIC-SPATIAL-REPRESENTATION1-0
 ISA INTRINSIC-SPATIAL-REPRESENTATION
  SALIENCE NIL
  HEADING NIL
  BEARING RIGHT
  DISTANCE 15.0
  VIEWPOINT SPATIAL-OBJECT0-0-0
  IDENTITY SPATIAL-OBJECT2-0
```

Where spatial-object0-0-0 is:

```
SPATIAL-OBJECT0-0-0
 ISA SPATIAL-OBJECT
  INTRINSICALITY T
  SHAPE ARROW
  ORIENTATION DOWN
  PERCEPTUAL-REFERENCE ARROW0-0
```

And spatial-object2-0 is:

```
SPATIAL-OBJECT2-0
 ISA SPATIAL-OBJECT
  INTRINSICALITY NIL
  SHAPE SQUARE
  ORIENTATION NIL
  PERCEPTUAL-REFERENCE SQUARE0-0
```

Where “square0-0” is a chunk of chunk-type “square,” a subtype of the visual-object chunk-type and polygon-feature is a subtype of the visual-location chunk-type.

3.4. Creation of Intrinsic Spatial Representation Chunks

The SO and SR types each get instantiated in the module as a chunk-type, and the module places each of these chunk-types in its own buffer. This means that the module can simultaneously represent one location in each of the FORs. When the module receives a request to create a spatial-object and the value passed for the intrinsicality slot is t, then in addition to the creation of eye-centered-spatial-representation and allocentric-spatial-representation chunks the model will automatically attempt creation of an intrinsic-spatial-representation chunk, to be placed in its intrinsic buffer. It will also copy the intrinsic FOR-supporting spatial-object chunk to ACT-R’s imaginal buffer. This is to keep available a reference to the spatial-object chunk that will become the intrinsic-SR chunk’s viewpoint and also that viewpoint’s orientation relative to the observer (the model). The orientation is important for remapping the perceived bearing to the IFOR target object to be in terms of the IFOR anchor. For example, in the task environment depicted in Figure 2 the red arrow is oriented down relative to the observer’s egocentric FOR so that in the red arrow’s intrinsic FOR, the red triangle would be to the red arrow’s left. But since the red triangle is to the observer’s right, the red triangle’s “right” bearing must be remapped to be in terms of red arrow’s orientation in order to generate that intrinsic spatial relationship.

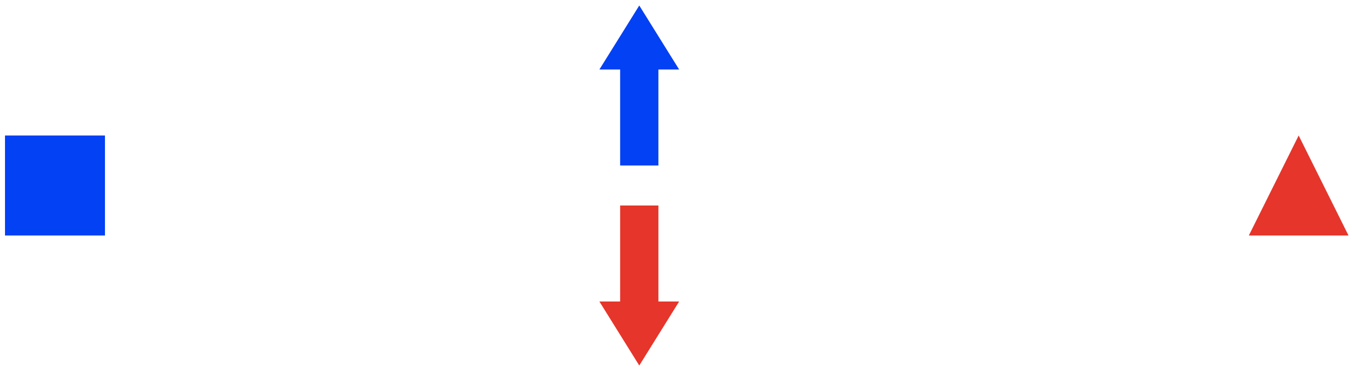

Figure 2. A simple spatial task environment with two potential intrinsic IFOR-supporting objects (arrows), each with two potential IFOR relationships (each arrow with square or triangle).

When creating an intrinsic-spatial-representation chunk the next thing the model will do is request a visual-location chunk of something that is the same color as the currently-attended item, but not in the same position (so that it does not get the same location). The assumption here is that the model has generated SRs for an object that can support an intrinsic FOR and that when that happens people automatically generate an IFOR representation based on it, if there is one that is salient available. The module uses color congruence as a proxy for bottom-up spatial salience in this process. If the visual-location buffer returns a visual-location chunk matching the request, then the spatial module uses that chunk to create a whole new set of spatial chunks. But this time it also creates an intrinsic-spatial-representation chunk using the just-returned visual-location chunk to describe the location of the IFOR target relative to the IFOR anchor, the spatial-object chunk that had previously been copied to the imaginal buffer. The ISR's bearing is that of the target relative to the anchor, as is the distance. The ISR's viewpoint is the anchor object, represented by the spatial-object chunk held in the imaginal buffer. The identity is a reference to the spatial-object chunk encoding the spatial-object representation of the IFOR target.

4. Demonstration Experiments

Two simple demonstration experiments accompany the FORMS module. They are both based on the arrow, square, and triangle task environment depicted in Figure 2. One is a perception-based experiment wherein the ACT-R model must indicate whether the blue square lies to the left or right according to the EFOR, screen-based AFOR, or the IFOR based on one or the other of the arrows. The retrieval experiment uses the same stimuli but in this case the model studies the display for four seconds. Then the display disappears and the model must respond to the same three questions, but this time from declarative memory.

4.1. Perceptual Experiment

The perceptual experiment is a demonstration of the spatial module emphasizing its capacity to construct primitive bearing representations and then to remap those to be relative to an IFOR anchor object. The stimuli depicted in Figure 2 remain on-screen and the model must respond according to one of two goal chunks the experimenter may hand to the goal-focus function:

1. Where is the blue square relative to the blue arrow?
2. Where is the blue square relative to the red arrow?

These two goals both require that the model find the blue square, construct its primitive bearing representation, then remap the square's bearing to be relative to the anchor object of interest, either the blue or the red arrow.

To run this model make sure the model's call to goal-focus has as its chunk argument find-square. The experiment has a post-event hook function that, when the experiment receives the model's key press, increments the model's goal to find-square-relative-to-blue-arrow and then to find-square-relative-to-red-arrow. Call (do-exp) to run the model on the task. If you only want the model to perform one trial at a time, then comment out the following line near the end of the do-exp function:

```
(add-post-event-hook 'new-instructions)
```

Then give the model's call to goal-focus the appropriate goal chunk argument.

4.2. Retrieval Experiment

The retrieval experiment is another demonstration, this time emphasizing salience for spatial representations based on declarative memory. Since the task is to indicate the square's location from memory, the task is easy if that representation had already been generated, as in the case of the EFOR or the salient IFOR relationship, the blue square's position relative to the blue arrow. But because, for the purposes of this simple example, the blue square relative to the red arrow relationship is not salient it will not have been generated by the model during the study phase. Therefore it will have no such representation in declarative memory available to it. But it can still answer the question by recalling the EFOR representations of the blue square a red arrow separately and then using those it can construct the blue square and red arrow IFOR representation. However, because it has, instead of just one retrieval to perform, two retrievals and an IFOR representation to generate the model takes much longer (approximately 10 seconds) in this condition than in any of the other conditions of the retrieval task (approximately 3 seconds).

Like the perceptual demo task, the retrieval demo task runs with a call to do-exp. Unlike the perceptual task, the retrieval task always begins with the model's goal-focus set to study-stimuli. After the task's study period expires (after four seconds), the experiment calls goal-focus with one of the goal-specific chunks. The function, phase2, sets the retrieval phase goal toward the end of its definition, in a lambda expression. Simple change the lambda expression's goal-focus' chunk argument to change the model's task.

5. Developer Reference: Technical Details

This section is intended to be a reference for developers wishing to modify the FORMS module. It will describe the various Lisp functions that comprise the FORMS module's functioning. Developer-level knowledge of ACT-R is assumed.

In essence, the spatial module receives requests to its spatial object buffer. Spatial requests include perceptual information in the form of a reference to a visual-object or visual-location chunk. Its request interface function calls another function, handle-spatial-object, to begin the process of constructing all of the chunks that the spatial module makes. The module uses a series of handler functions like handle-spatial-object to process the perceptual information referenced in its request to also construct the eye-centered EFOR, AFOR, and when possible, the IFOR chunks. It does this in an all-at-once fashion so that a single request results in the simultaneous creation of multiple spatial chunks, each representing the location within a different FOR. Furthermore, if the location contains an object with an intrinsic orientation, then the FORMS module will automatically request a visual-location for a salient (currently defined simply as another location that is the same color as the currently-attended location) target location. Then it will construct SR chunks for that location, including an IFOR chunk representing the target's location relative to the IFOR's anchor, the previously attended location that had the intrinsic orientation. The resulting chunks are all made available to the rest of ACT-R via the spatial module's buffers, one each for the chunk-types that it creates.

Discussion of standard ACT-R module interface functions such as the request and reset functions is limited as those are described in the ACT-R documentation (Bothell, 2010). Bothell's (2010) convention for describing commands' syntax will be used:

- items appearing in bold are to be entered verbatim
- items appearing in italics take user-supplied values
- items enclosed in {} are optional
- * indicates that any number of items may be supplied
- + indicates that one or more items may be supplied
- | indicates a choice between options which are enclosed in [square brackets]
- (parentheses) denote that the enclosed items are to be in a list
- a pair of items enclosed in <angle brackets> denote a cons cell with the first the car and the second the cdr
- -> indicates that calling the command on the left of the "arrow" will return the item to the right of the "arrow"
- ::= indicates that the item on the left of that symbol is of the form given by the expression on the right

Many of the FORMS module's functions take a chunk-spec as an argument, since the module takes buffer requests as input. Typically these functions then output a call to *schedule-set-buffer-chunk* with arguments to have the ACT-R scheduler put the chunk that the function just made into the corresponding buffer of the FORMS module. The task environment depicted in Figure 2 will serve as stimuli for the examples that follow.

5.1 handle-spatial-object

Syntax:

handle-spatial-object *chunk-spec* -> [an ACT-R scheduler event]

Description:

Handle-spatial-object is called by the module's request interface function, *request-spatial*, and simply transfers values of the request's chunk-spec to their respective slots in a new spatial-object chunk.

Example:

```
? (handle-spatial-object (define-chunk-spec-fct '(isa spatial-object intrinsicality t shape
arrow orientation up perceptual-reference arrow0-0)))
#S(ACT-R-EVENT :TIME 1.12 :PRIORITY 0 :ACTION SET-BUFFER-CHUNK
        :MODEL CHUNK-SPEC-TEST-MODEL :MP DEFAULT :MODULE SPATIAL
        :DESTINATION NIL :PARAMS (SPATIAL-OBJECT SPATIAL-OBJECT3)
        :DETAILS NIL :OUTPUT LOW :WAIT-CONDITION NIL :DYNAMIC NIL)
```

The spatial-object chunk that results from the above scheduled ACT-R event looks like this:

```
? (pprint-chunks spatial-object0)
SPATIAL-OBJECT0
 ISA SPATIAL-OBJECT
  INTRINSICALITY T
  SHAPE ARROW
  ORIENTATION UP
  PERCEPTUAL-REFERENCE ARROW0-0

(SPATIAL-OBJECT0)
```

5.2 handle-eye-centric

Syntax:

handle-eye-centric *chunk-spec* -> [an ACT-R scheduler event]

Description:

When ACT-R sets the spatial-object buffer chunk it triggers a post-event hook function, post-so-make-eye-sr, which calls handle-eye-centric with the current request's chunk-spec. Since the chunk-spec has a reference to a perceptual chunk (either a visual-object or a visual-location), handle-eye-centric (and its helper functions) use that perceptual chunk to parse ACT-R's visual world into a spatial world instantiated as bearing symbols such as "up." Handle-eye-centric returns a scheduled ACT-R event that puts an eye-centered-spatial-representation chunk into the FORMS module's eye-central buffer.

Example:

```
? (handle-eye-centric (define-chunk-spec-fct '(isa spatial-object intrinsicality t shape ar-
row orientation up perceptual-reference arrow0-0)))
#S(ACT-R-EVENT :TIME 1.12 :PRIORITY 0 :ACTION MODULE-REQUEST
        :MODEL CHUNK-SPEC-TEST-MODEL :MP DEFAULT :MODULE SPATIAL
        :DESTINATION NIL
        :PARAMS (VISUAL-LOCATION
            #S(ACT-R-CHUNK-SPEC :TYPE #S(ACT-R-CHUNK-TYPE :NAME VISUAL-LOCATION
                                :DOCUMENTATION NIL
                                :SUPERTYPES (VISUAL-LOCATION)
                                :SUBTYPES (POLYGON-FEATURE
                                      CHAR-PRIMITIVE
                                      VISUAL-LOCATION)
                                :SLOTS (SCREEN-X
                                    SCREEN-Y
                                    DISTANCE
                                    KIND
                                    COLOR
                                    VALUE
                                    HEIGHT
                                    WIDTH
                                    SIZE)
                                :EXTENDED-SLOTS NIL)
                      :SLOTS (#S(ACT-R-SLOT-SPEC :MODIFIER =
                                 :NAME COLOR
                                 :VALUE RED)
                          #S(ACT-R-SLOT-SPEC :MODIFIER -
                                 :NAME SCREEN-X
                                 :VALUE 380)
                          #S(ACT-R-SLOT-SPEC :MODIFIER -
```

```
                                   :NAME SCREEN-Y
                                   :VALUE 410))))
        :DETAILS "MODULE-REQUEST VISUAL-LOCATION" :OUTPUT MEDIUM
        :WAIT-CONDITION NIL :DYNAMIC NIL)
```

The eye-centered-spatial-representation chunk that results from the above scheduled ACT-R event looks like this:

```
? (pprint-chunks EYE-CENTERED-SPATIAL-REPRESENTATION0)
EYE-CENTERED-SPATIAL-REPRESENTATION0
 ISA EYE-CENTERED-SPATIAL-REPRESENTATION
  SALIENCE NIL
  HEADING UP
  BEARING CENTER
  DISTANCE 15.0
  VIEWPOINT EYE-SO
  IDENTITY SPATIAL-OBJECT0-0

(EYE-CENTERED-SPATIAL-REPRESENTATION0)
```

5.3 handle-allocentric

Syntax:

handle-allocentric *chunk-spec* -> [an ACT-R scheduler event]

Description:

When ACT-R sets the eye-central buffer chunk it triggers a post-event hook function, post-so-make-allo-sr, which calls handle-allocentric with the current request's chunk-spec. For now, since we assume the modeled subject is sitting at a computer and the visual task environment is the computer screen, this essentially just copies what the spatial module has already done for the eye-central buffer. However it sets the allocentric SR's viewpoint to be the primitive spatial object, screen-SO, in effect making the computer screen the anchor for the allocentric FOR. Handle-allocentric returns a scheduled ACT-R event that puts an allocentric-spatial-representation chunk into the FORMS module's allocentral buffer.

Example:

```
? (handle-allocentric (define-chunk-spec-fct '(isa spatial-object intrinsicality t shape ar-
  row orientation up perceptual-reference arrow0-0)))
#S(ACT-R-EVENT :TIME 1.12 :PRIORITY 0 :ACTION SET-BUFFER-CHUNK
        :MODEL CHUNK-SPEC-TEST-MODEL :MP DEFAULT :MODULE SPATIAL
        :DESTINATION NIL
```

```
        :PARAMS (ALLOCENTRAL ALLOCENTRIC-SPATIAL-REPRESENTATION4)
        :DETAILS NIL :OUTPUT LOW :WAIT-CONDITION NIL :DYNAMIC NIL)
```

The allocentric-spatial-representation chunk that results from the above scheduled ACT-R event looks like this:

```
? (pprint-chunks ALLOCENTRIC-SPATIAL-REPRESENTATION0)
ALLOCENTRIC-SPATIAL-REPRESENTATION0
 ISA ALLOCENTRIC-SPATIAL-REPRESENTATION
  SALIENCE NIL
  HEADING UP
  BEARING CENTER
  DISTANCE 15.0
  VIEWPOINT SCREEN-SO-0
  IDENTITY SPATIAL-OBJECT0-0

(ALLOCENTRIC-SPATIAL-REPRESENTATION0)
```

5.4. Intrinsic Chunks

When the spatial object chunk being generated by the spatial module has an intrinsic orientation and there is another same-color location in the visicon, the module will make an intrinsic-spatial-representation chunk. These chunks are made by an interaction of three major functions, handle-intrinsic, handle-eye-centric, and stuff-intrinsic-buffer. When ACT-R sets the allocentral buffer chunk it triggers a post-event hook, post-so-make-int-sr, which calls handle-intrinsic. Handle-intrinsic tests whether there is an IFOR-supporting spatial-object chunk in the imaginal buffer. If there is not, then it copies the current cognition cycle's spatial-object chunk to the imaginal buffer regardless of its possession of an intrinsic orientation axis.

When the spatial-object is capable of supporting an IFOR then handle-eye-centric will begin the process of creating an IFOR by making a request of the visual-location buffer to find a location of the same color as the current spatial-object—this is a very simplistic assumption about how salience works, but it is sufficient for an initial version of the theory and implementation. Handle-eye-centric does this because it comes earlier in the processing cycle than handle-intrinsic, but after the spatial-object chunk has been placed in the spatial-object buffer. If the vision module finds such a location and therefore sets the visual-location buffer to be the chunk representing it and the imaginal buffer holds a spatial-object chunk that has an intrinsic orientation, then stuff-intrinsic-buffer starts the process of automatically, involuntarily creating an IFOR representation. It does this by scheduling a request of the spatial-object buffer. Remember, the module uses an all-at-once mechanism such that issuing a request to the spatial-object buffer sets in motion, in effect, automatic requests to all the other buffers. The module then makes the respective spatial chunks in all the buffers for the newly-found location. This time when it gets to handle-intrinsic there should be an IFOR-supporting spatial-object chunk in the imaginal buffer—remember, it was put there in the last cycle by handle-intrinsic. In this cycle handle-intrinsic references that chunk in the viewpoint slot of the intrinsic-spatial-representation chunk that it creates. The intrinsic-SR chunk, then, represents the location of the target relative to the viewpoint. It is worth noting that currently the FORMS module generates intrinsic-SR chunks only by this automatic triggering mechanism.

5.4.1. handle-intrinsic

Syntax:

handle-intrinsic *chunk-spec* -> [an ACT-R scheduler event]

Description:

Handle-intrinsic will handle requests for intrinsic-spatial-relationship chunk-types, making one such chunk if there's a valid spatial-object chunk in the imaginal buffer. A valid spatial-object chunk would have been copied to the imaginal buffer earlier if the current spatial object supports an IFOR and this chunk will serve as the viewpoint, in other words the anchor, for the IFOR representation. Handle-intrinsic is called when ACT-R sets the allocentral buffer chunk. That event triggers a post-event hook function, post-so-make-int-sr, which calls handle-intrinsic with the current request's chunk-spec. Handle-eye-centric returns a scheduled ACT-R event that puts the intrinsic-spatial-representation chunk that handle-intrinsic makes into the FORMS module's intrinsic buffer. Handle-intrinsic calls helper functions to remap the bearing slot value of the eye-centered-SR chunk representing the current target according to the viewpoint spatial-object's orientation.

Example:
```
? (handle-intrinsic (define-chunk-spec-fct '(isa spatial-object intrinsicality t shape arrow
orientation up perceptual-reference arrow0-0)))
#S(ACT-R-EVENT :TIME 1.12 :PRIORITY 0 :ACTION SET-BUFFER-CHUNK
        :MODEL CHUNK-SPEC-TEST-MODEL :MP DEFAULT :MODULE SPATIAL
        :DESTINATION NIL
        :PARAMS (IMAGINAL SPATIAL-OBJECT0-1 :REQUESTED NIL) :DETAILS NIL
        :OUTPUT LOW :WAIT-CONDITION NIL :DYNAMIC NIL)
```

The intrinsic-spatial-representation chunk that results from the above scheduled ACT-R event looks like this:

```
? (pprint-chunks INTRINSIC-SPATIAL-REPRESENTATION0)
INTRINSIC-SPATIAL-REPRESENTATION0
 ISA INTRINSIC-SPATIAL-REPRESENTATION
  SALIENCE NIL
  HEADING NIL
  BEARING LEFT
  DISTANCE 15.0
  VIEWPOINT SPATIAL-OBJECT0-0-0
  IDENTITY SPATIAL-OBJECT1-0

(INTRINSIC-SPATIAL-REPRESENTATION0)
```

6. Issues and Future Work

6.1. Theoretical Issues

FORMS is meant to be a stand-alone theory of human spatial competence. It is not dependent upon ACT-R for its ontology. Rather ACT-R, in this case, provides a useful means of getting perceptual information into a spatial representational architecture and then again for making use of those representations once they are generated. FORMS theory could be implemented in other cognitive architectures, as well.

FORMS does not currently implement a bottom-up account of spatial salience for currently-active representations, as might be the case in certain Gestaltist configurations. For example, an array of dots, all grouped closely together and lying along a line might be salient in a spatially-meaningful way while other dots in the environment, scattered seemingly at random, would likely not be (Figure 3). The module also lacks means to create SRs from chunk types other than visual and visual-location, which is clearly an issue as people can clearly do so based on aural or imaginative information. Finally there is not currently implemented method of automatically generating nor updating SRs (but see the discussion on IFOR buffer stuffing). Entangled with the salience issue is the issue of whether or not IFOR representations should really be generated involuntarily and automatically as they currently are. Future research will have to determine automaticity of the IFOR.

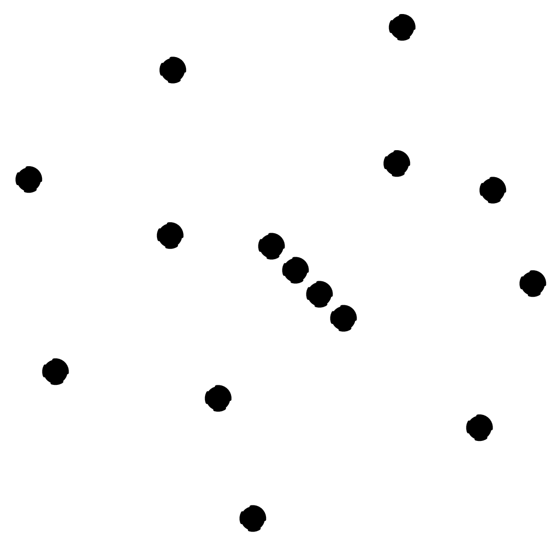

Figure 3. An example of Gestaltist-type spatial salience by proximal and configural grouping: The four dots arranged along a line and near to each other are salient while other dots in the image are not.

A good test of the FORMS module would involve reasoning in a fundamentally spatial task that has large components drawing on situation awareness (creating and maintaining) and integration of spatial & situational information from multiple frames of reference, particularly from intrinsic frames of reference, to synthesize an evolving situation awareness to be used in the task's decision making. Such a problem domain could be found in several industries including remote surgery (e.g., laparoscopy), remotely operated vehicles (e.g., submersibles, aircraft), human-robot interaction, vehicle operation (particularly large and complex vehicles with poor visibility, such as submarines, large surface ships, trains, and commercial aircraft), and air traffic control.

Going forward, aside from the obvious issue of completing the list of FORs humans use (e.g., body- and hand-centered egocentric)other refinements of the module should be evaluated against empirical data and revised accordingly. For instance, the remapping mechanism that synthesizes egocentric primitive SRs into IFOR representations may actually lie within mechanisms of central cognition rather than in a specialized spatial module (Gunzelmann et al., 2004). The spatial parsing mechanism is unevaluated—most literature seems to suggest that such primitive SRs are more or less instantly available to us (Klatzky, 1998), but there may be other subtleties in things like the fuzziness of demarcations between bearing regions or task- or strategy-dependent methods of parsing. Also the parsing mechanism and FORMS' ontology must be extended to account for three dimensions of space, both in how three dimensions are represented and in how those representations support three-dimensional spatial reasoning. FORMS also has no explicit account of time and it is not clear whether it would extend naturally as is to cover dynamic spatial behavior. Finally, as mentioned earlier, a meaningful account of salience remains to be completed.

6.2. Technical Issues

As FORMS is meant to be about human spatial competence, it may be implemented within other cognitive architectures. Other implementation efforts of necessity will have to be concerned with out information gets into the FORMS system, which means some kind(s) of perceptual representations must be given to FORMS so that FORMS may have a view of the world with which to construct its maps. Another concern is in making FORMS' representations available to the rest of cognition. One of FORMS' cornerstones is its tenet that multiple spatial representations, each in different FORs, are available to a person at any given time. Finally, one of the things that ACT-R gives to this particular implementation of FORMS is some instantiations of salience. ACT-R's production system can select among spatial representations according to ACT-R's internal state, including goals. Another cognitive architectural implementation of FORMS will have to solve the same salience problem.

As the FORMS module uses ACT-R's imaginal buffer to store spatial-object chunks whenever it generates an intrinsic-SR chunk—without ACT-R firing a production that references the imaginal buffer on the production's left-hand side nor requesting it on the production's right-hand side—it runs the risk of colliding with other cognitive activity that may be using the imaginal buffer. However, aside from the implementational potential for trouble posed by this scheme this is partially a theoretical issue, too, as ACT-R's imaginal module is partly theorized to account for spatial imagery as it reflects that sort of cognitive activity and its reflection in parietal BOLD response (Anderson, 2007; Anderson, Qin, Jung & Carter, 2007).

7. References

ACT-R Research Group (2010). ACT-R: Theory and architecture of cognition. [Web page]

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of the mind. Psychological Review, 111(4), 1036-60.

Anderson, J. R. (2007). *How can the human mind exist in the physical universe?* New York, NY: Oxford University Press.

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of the mind. *Psychological Review*, *111* (4), 1036-60.

Anderson, J. R., Qin, Y., Jung, K. J., & Carter, C. S. (2007). Information-processing modules and their relative modality specificity.. *Cogn Psychol*, *54* (3), 185-217.

Bothell, D. (2010, June 5). *ACT-R 6.0 Reference Manual*. ACT-R Research Group. Retrieved June 29, 2010, from act-r.psy.cmu.edu

Byrne, M. D. (2001). ACT-R/PM and menu selection: Applying a cognitive architecture to HCI. *International Journal of Human Computer Studies*, *55* (1), 41-84.

Byrne, M. D. & Kirlik, A. (2005). Using computational cognitive modeling to diagnose possible sources of aviation error. *The International Journal of Aviation Psychology*, *15* (2), 135 – 155.

Gray, W. D., Schoelles, M. J., & Myers, C. W. (2004). Meeting Newell's other challenge: Cognitive architectures as the basis for cognitive engineering. *Behavioral and Brain Sciences*, *26* (05), 609-610.

Gunzelmann, G., Anderson, J. R., & Douglass, S. (2004). Orientation tasks with multiple views of space: Strategies and performance. *Spatial Cognition & Computation*, *4* (3), 207-253.

Klatzky, R. L. (1998). Allocentric and Egocentric Spatial Representations: Definitions, Distinctions, and Interconnections. *Spatial Cognition - An Interdisciplinary Approach to Representation and Processing of Spatial Knowledge (Lecture Notes in Artificial Intelligence 1404)*, 1–17.

Newell, A. (1973). You can’t play 20 questions with nature and win: Projective comments on the papers of this symposium. In W. G. Chase (Ed.), *Visual information processing*. (pp. 283-308). New York: Academic Press.

Wang, H., Johnson, T. R., & Zhang, J. (2001). The mind's views of space. In *Proceedings of the Third International Conference of Cognitive Science*.