Lightweight Schematic Explanations of Robot Navigation

Robert St. Amant (robert.a.stamant2.civ@mail.mil), MaryAnne Fields (mary.a.fields22.civ@mail.mil), Brian Kaukeinen (brian.t.kaukeinen.civ@mail.mil), Christa Robison (christopher.j.robison5.civ@mail.mil)
U.S. Army Research Laboratory
Adelphi, MD, U.S.

Abstract

This paper describes a representation to support explanation of robot navigation based on image schemas, a set of abstractions closely tied to embodiment. The representation is intended to satisfy two criteria: an explanation must provide a logical or causal account of the phenomenon to be explained and be understandable by its audience. Evidence in the literature of cognitive linguistics and related fields suggests that image schemas satisfy the second criterion. We provide evidence for the first with an Answer Set Programming formalization of navigation-related image schemas. Schema-based explanation representations are generated for a robot navigating through a simple indoor environment.

Keywords: Image schema; robot navigation; explanation

Introduction

Symbolic representations in artificial agents (robots, in this paper) are often invaluable—they allow for the concise specification and communication of potentially complex behaviors. This pragmatic, engineering motivation sometimes leaves open the question of why one representation is better than another. Our interest is in a related issue that has gained increasing attention in recent years: the ability of a robot to explain its actions.

This paper describes a representation, based on *image schemas*, for the explanation of robot navigation. "Image schema" is a term coined by Lakoff (1987) and Johnson (1987): a structured, general pattern intended to capture experience "at the level of our bodily movements through space, our manipulations of objects, and our perceptual interactions." The contribution of this paper is to show how a schematic cognitive structure can be derived from patterns of navigation actions recorded by a physical robot. Image schemas appear to be a natural fit for explanation.

Navigation may at first seem to be a trivially easy domain, for human beings if not for robots, but researchers in human spatial cognition characterize navigation as being among the most complex of cognitive operations (Wiener, Büchner, & Hölscher,2009). For example, an ontology developed for urban transportation (Timpf,2002) contains concepts for path, start, goal, connection, transportation mode, map, sign, direction, distance, and time, plus names of specific entities (e.g., streets and subway stations). When we navigate, we draw on our memories and on external maps. We think about navigation at different levels of abstraction; we chain together navigation plans and paths; we combine navigation with other activities. Many navigation tasks may be easy to understand, but they can grow complex enough to require explanation of why a specific route is followed or avoided.

What counts as a good explanation? In the philosophy of science, the standard definition of an explanation has two parts (Hempel & Oppenheim,1948): a description of a phenomenon to be explained (the *explanandum*) and the explanatory account itself (the *explanans*). Both parts can be formulated as sentences in logic, with the explanandum being one sentence and the explanans being a set of sentences for which the explanandum is a consequence. The sentences in the explanans are of two types, general laws and antecedent conditions specifying applicability, and for a sound explanation they must be true. In other words, a scientific explanation captures the causes or logic of a phenomenon.

Lombrozo and Carey (2006, p. 169) broaden the definition above: the logical or causal process identified in an explanation "can be subsumed under some kind of pattern or causal schema" that is already understood by a questioner or introduced as part of the explanation itself. That is, a *psychological explanation* (which we will henceforth call simply an explanation) is a bridge between the explanans and the explanandum that has a foundation in a questioner's prior knowledge. An explanation should be understandable by the audience to whom it is directed (as well as the explainer). Further, explanations do not exist in a vacuum but are are typically part of a larger context, which includes the goals of the audience and the situation in which the explanation is offered (Mueller, Hoffman, Clancey, Emrey, & Klein,2019).

In the next section we briefly review the relevant literature on image schemas, as a partial account of spatial reasoning. Schematic structures alone do not satisfy the causal/logical requirement of explanations, however; for this we depend on a formalization in terms of Answer Set Programming (Gebser et al.,2011), which supports inferences to necessary components of an explanation in an abstract, symbolic representation. We then describe the conversion of sensor and control data on a physical robot into the abstract representation and walk through an example.

Related work

In the infant cognition literature, Mandler and Pagán Cánovas (2014, p. 519) outline a set of spatial primitives that act as image schemas: "By themselves or in combination they structure the conceptual representations that describe events."

These image schemas plausibly underlie infant spatial cognitive capabilities, gained in the first six to seven months after birth: PATH, START-PATH, END-PATH, PATH-TO; LINK; THING; $\pm$CONTACT; CONTAINER, OPEN, INTO, OUT OF; LOCATION; $\pm$MOVE, ANIMATE MOVE, BLOCKED MOVE; BEHIND; APPEAR, DISAPPEAR, EYES. It will be useful for our purposes to expand the set of spatial relationships represented by BEHIND, to include NEAR/FAR, LEFT/RIGHT, IN FRONT

OF/BEHIND. This is not a complete set of image schemas, even for navigation (Croft & Cruse,2004), but provides a reasonable starting point.

For clarity, a PATH is "the way to get [somewhere];" the most general version of the schema is commonly called SOURCE-PATH-GOAL (Lakoff,1987), with its components as given by its name. BLOCKED MOVE is sometimes referred to as BLOCKAGE (Cervel,1999), with components that include a path, a moving entity, and another entity acting as an obstacle. THING is an entity that can be perceived in space, which we will interpret here as a physical object. LINK is a general, contingent relationship that may come into being between objects or schemas.

Image schemas exist in a specialization hierarchy. For example, a path component of a SOURCE-PATH-GOAL might involve continuous motion, or it might consist of a sequence of discrete steps (each of which can be thought of as a PATH itself, with an atomic transition between locations). Image schemas can also be related by composition. The end of a PATH can be a LOCATION; a BLOCKAGE applies to a PATH.

Image schemas are implicitly associated with activities or events. We call these *characteristic operations*—informally, what an image schema is for. For example, a characteristic operation of the SOURCE-PATH-GOAL schema is for some agent to traverse it in a specific direction. One characteristic operation of a CONTAINER is that it can contain other objects, or that an agent can MOVE INTO or OUT OF one. (Some characteristic operations are schemas themselves, or incorporate schemas.) In a navigation context, regions are a type of CONTAINER: one can enter INTO or exit OUT OF a REGION. Buildings, rooms, and even deadends are CONTAINERs in the same way.

Image schemas have been adopted as conceptual primitives in many other fields aside from infant spatial reasoning. Geographic information systems are one example (Walton & Worboys,2009). In an extensive discussion, Frank and Raubal (1999, p. 67) observe that image schemas capture geographical concepts in a way that "comes close to how people use them in their everyday lives."

Another area of related work is AI planning. Robots are closely associated with planning, from the abstract level of classical planning down to the detailed level of planning paths. Our own past work used image schemas to capture patterns of behavior in planning agents (St. Amant et al.,2006). Navigation planning is typically handled as a search problem with a single operator for moving between locations, rather than distinct planning actions for reaching different states. Korpan and Epstein (2018)'s WHY-PLAN explains navigation plans, by examining differences between a robot's objective function and that of a human being, mapping their components to natural language phrases. For example, if a robot's objective function is sensitive to crowd density, it can contrast its solution plan with a human's: "This path is ⟨slightly⟩ ⟨less crowded⟩ than the alternative."

In an area closely related to explanation, Rosenthal et al. (2016) describe a system that generates narrative "verbalizations" to describe a navigation path. Rosenthal et al.'s navigation paths are representative of most such work we are familiar with: a path contains a goal location, a starting location, and an ordered list of intermediate waypoints (plus collinear subsequences of waypoints, to facilitate the identification of turns). Such representations map naturally onto the SOURCE-PATH-GOAL image schema. Landmarks, with appropriate semantic tags, and their spatial relationships to the robot can also be easily interpreted in schematic terms. None of this is surprising. Most of the other image schemas remain only implicit in navigation plan representations, however, though they could plausibly contribute to explanations. In the next section, we show how they can be made explicit.

Schemas for Navigation Paths

In this section we describe a formalization of image schemas to support explanation. Our target is a logical representation sufficient for an explanation, following the lead of other work in planning (Chakraborti, Sreedharan, Zhang, & Kambhampati,2017;Fox, Long, & Magazzeni,2017). Generation of the text and narrative of explanations, such as carried out by systems above, is part of the task that we leave for future work.

Answer Set Programming, with roots in knowledge representation and reasoning, has become a popular paradigm for declarative problem solving. ASP has shown promise in spatial and temporal reasoning (Li,2012) and commonsense reasoning (Balduccini,2009). Our implementation relies on the Potassco set of tools for ASP; rules are encoded using the input language of Gringo (Gebser et al.,2011).

A problem specification is separated into two parts: a specific problem instance, expressed as predicates; and an encoding, a general set of inference rules that apply to any problem instance. For explanation of a navigation path, a problem instance consists of locations and objects in environment, the initial location of the agent, the path it follows, and the commands it issues to follow the path. (We refer to an "agent" in this context as a reminder that this is a high-level abstraction of a robot—the agent is not even explicitly represented.) Time and space, in the form of a set of locations, are discretized; a time limit `M` for execution of the planned path (`moves(M)`) is also provided.

Locations are named by unique constants, e.g., `loc(`x_1`)`. A path has a source location and a goal location, `path(`x_0`,`x_f`)`, and a sequence of waypoints to be traversed in order. These are expressed as steps with paired locations; if `P` is a path, then `step(P,`x_0`,`x_1`)`, `step(P,`x_1`,`x_2`)`, ..., `step(P,`x_{f-1}`,`x_f`)`. The agent begins `at(`x_0`,1)`, where 1 is the starting time. The environment may also include landmarks, obstacles, or demarcated regions that occupy specific locations, e.g. `land(`l_1`,`x_6`)`, `obst(`o_1`,`x_3`)`. Landmark locations are disjoint from path locations; obstacles have a non-empty intersection with path locations; regions have neither restriction.

An ASP problem encoding is further divided into separate parts. In the generation part we can specify candidate solutions, actions taken by the agent. The agent may move from one location to another along the path by taking a step, a `move` action; moves are possible at any time `T = 1..M`, but no more than one move can be carried out at a given time.

```
{ move(X_i,X_j,T) :
      P = path(X_s,X_f), step(P,X_i,X_j) } <= 1 :-
   path(X_s,X_f), moves(M), T = 1..M.
```

The definition part of an encoding defines predicates for inferences contributing to a solution. For example, a change of location can be inferred based on volitional movement from the present location.

```
at(X_j,T+1) :- at(X_i,T), cmd_move(X_i,X_j,T),
                    move(X_i,X_j,T).
```

The integrity constraint part restricts inferences, including those related to the agent's movements. For example, the agent can move from a location only if it is at that location, and the agent cannot move to locations occupied by an obstacle. (An underscore, below, is an anonymous variable that can take on any value.)

```
:- move(X_i,_,T), at(X_j,T), X_i != X_j.
:- move(_,X,T), obstacle(_,X,T).
:- move(_,X,_), obstacle(_,X).
```

Spatial relationships other than `at` are also accommodated in the representation, though falling short of generality. (Commonsense reasoning and qualitative spatial reasoning pose well-known and unresolved challenges.) Our account is necessarily brief and incomplete, for reasons of space, but the description should give the flavor. The important point is what the encoding can produce, summaries of the execution of a given navigation plan in the form of predicates.

- `traversed(path(x_0,X_g))`, `at(X_g,T)`: The path was followed until the agent reached its goal, with the last action taken at time `T`.
- `blocked(X_j,P)`, `at(X_i,T)`: The path was blocked by some obstacle at location X_j; at time `T` the agent was left at X_i.
- `stopped(X,P)`, `at(X,T)`: The path was followed until the agent stopped at `X`; this predicate lets us distinguish quiescence from being prevented from moving.

These mechanics provide for the construction of a set of ground terms (i.e. predicates containing no variables) that form the "logic" of an explanation for a navigation problem instance. The representation is limited in its discretization of space and time, but it can manage simple changes over time. For example, obstacles may be permanent or temporary, as might be presented by a person or object moving across a path at a specific time steps: `obst(o_2,x_4,3)`, `obst(o_2,x_4,4)`. A strong limitation is that unpredicted errors and deviations in behavior cannot be explained by our approach.

Image schemas have been formalized in other mathematical and logical formalisms (Frank & Raubal,1999;Kuhn,2007;Walton & Worboys,2009). One subtle issue is the intended application of the formalization. Most such work aims at description of entitities and their changing relationships. We have to make an additional commitment to interpreting image schemas in logical and even causal terms, because that is what is necessary for explanations. A `move` action, for example, does more than describe what happens; an agent carries out the action *in order to* change its location. This is a commonplace assumption in planning but important to make explicit because logical/causal interpretation is central to explanation.

At the end of the next section we show how this formalism works for a real navigation problem.

A Navigation Scenario

This section lays out an example navigation scenario, one that we can use as a target for explanation.

System

The hardware for this work is a K-Bot platform, from University of Pennsylvania. The robot sensor package includes a microstrain 3DM-GX2 IMU, two Point Grey Firewire Grasshopper cameras, a Point Grey GigE Blackfly camera, a Point Grey Bumblebee2 stereo camera, and an ASUS Xtion pro RGB-D camera, as well as a two Hokuyo UTM-30LX-EW (Ethernet) Scanning Laser Rangefinder and a Velodyne HDL-32E LiDAR. Software is built on ROS (Robot Operating System) Indigo by the Open Source Robotics Foundation, plus ROS software drivers to read sensor data streams.

ROS's primary navigation stack is `move_base`, which implements a number of essential capabilities. Cost maps provide a two-dimensional representation, in the form of a grid, of the cost to traverse a space. Some cells in the map may have nominal cost (a constant `FREE_SPACE`), indicating that the robot may move freely through the location corresponding to the cell. Physical objects or obstacles also occupy cells in the grid, which means that the cost recorded in those cells is the maximum possible (a constant named `LETHAL_OBSTACLE`). The region of each object is "inflated," meaning that the cells surrounding the region occupied by the object have an intermediate cost, to indicate a location with a risk of collision.

Planners supply the robot with a path based on the values contained in the cost map, given a starting location and a goal location. The global planner creates a general path to follow, a discrete sequence of locations, between the start and the goal. The local planner is responsible for attempting to follow the global path by generating movement commands for the robot; this path includes orientation information for the robot. When we refer to the path planner in the remainder of this paper, we mean the global planner.

Task Environment and Execution

The robot is tasked with moving across a warehouse floor. For the purposes of this scenario, a region in the center of the room is traversable, but it is also considered vulnerable and should be avoided. In a military scenario such a region might correspond to an area visible to a hypothetical observer, which is undesirable for movement under concealment; in an urban search and rescue scenario, the region might be where the ceiling above has been weakened and may fall. This task was chosen for explanation because an alternative, shorter path to the goal location on the other side of the room is obvious: a straight line. If the contextual information is absent, a questioner might reasonably ask the robot to explain.

To implement the navigation task, a custom ROS node was written to add such regions to the navigation stack via a separate layer of the cost map, as virtual obstacles, with corresponding lethal obstacle cell costs. A separate layer was used because the robot updates the cost map as it moves through the environment. Virtual objects or regions are not detected by the system's raytracing algorithm, and the costs associated with such objects would be overwritten. A separate layer also limited the need to modify existing ROS `move_base` software.

April tags (Olson,2011) were used to represent the centroid of a region of vulnerability, mounted on a physical cone for easy detection. The radius of the region was set programmatically. This was an alternative to assessing vulnerability directly; it allows comparable virtual information to be integrated into the physical environment. In our discussion of the scenario below, we will treat the April tags and the cone as being invisible to the robot, which would be the case if the vulnerable region were directly assessed.

In a sample execution of the task, the robot starts at a location in the lower right of Figure 1. The goal location is in the upper left of the figure. The robot begins with an empty cost map; cells are assigned costs based on sensor information about obstacles, in this experiment both real and virtual. The path planner searches for and returns a path from the start to the goal, a sequence of a few hundred locations, each with associated bookkeeping information. The vulnerable region in this example has been given a `LETHAL_OBSTACLE` cost, but in a different variation it could be given some lower `VULNERABLE` cost. The robot ends at the goal location and its path is shown as a dashed red line.

The visual representation shows the cost map that the path

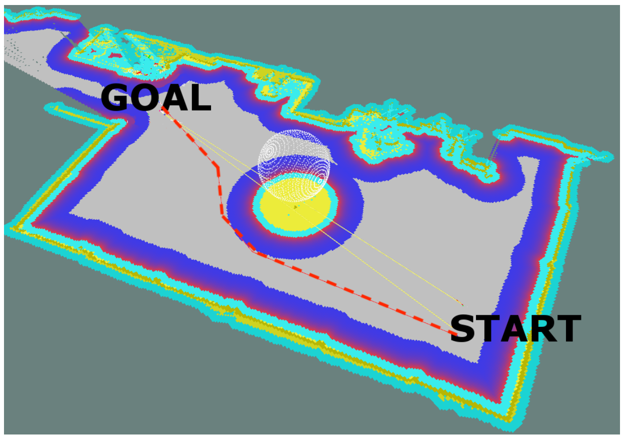

Figure 1: Movement around a region of vulnerability

planner accesses to construct the path that the robot follows. Different colors correspond to different cost values, in particular yellow for lethal obstacle cost, light gray for free space cost, and other colors for intermediate values produced by the "inflation" mentioned above. The outermost color is for regions with unknown cost. The physical borders of the room are clearly visible on the map; the upper area shows work tables and equipment. The circle on the map, with a sphere floating above, is the visual representation of the vulnerable region, derived from the April tag located at the center of circle. A physical obstacle would look similar, except that the region opposite the robot would not be visible to the its sensors and would thus have unknown cost values. In this scenario, the robot begins in a location from which all relevant cost values for path planning can be determined directly, which means that it can follow the path produced by the planner without the need for replanning.

Generating a representation

Figure 1 might be seen as an explanation: it indicates an obstacle in the center of the room, which the robot will avoid. By comparing a map with the physical environment it becomes clear that the object is virtual, representing a vulnerable region; the robot would presumably plan around it. But the phenomenon to be explained and the explanation are only implicit in the visualization. Even the basic vocabulary of navigation concepts is missing. The robot has no internal representation of these concepts, to impose an explanation on the visualization. At best we might say that the image helps viewers explain the robot's behavior to themselves.

What we want instead is the generation of an explicit abstract representation that captures the robot's behavior.

- *Locations*: All world locations are translated into 2D cells on the cost map, and those cells generate unique symbols. Not all locations or cells are used; relevant ones are identified in translating other kinds of objects.

- *Paths*: Three different types of paths are generated. A planned path is constructed from the set of waypoints produced by the ROS path planner. An external path is generated by an external process directly on the cost map (e.g., a hypothetical straight line path between two locations). An executed path is constructed from the sequence of the robot's sensed locations as it moves through the world. In the last case locations are sampled whenever a `cmd_vel` message is issued. After the conversion to cost map cells, path locations are filtered to remove jitter.

- *Commands*: For planned and external paths, locations are walked to create a set of `cmd_move` predicates, with programmatically generated timestamps. For executed paths, `cmd_vel` messages are converted to `cmd_move` predicates.

- *Landmarks, regions, and obstacles*: These are treated similarly, and in our navigation scenario, the vulnerable region can be any one of them, depending on the path and cost values. Generation is from an object's external specification. A directional relationship is computed to the object from each location on the path. These relationships are determined by a validated cognitive model of spatial projective terms, AVS (Regier & Carlson,2001), which we have used in previous robotics projects (Ward, St. Amant, & Fields,2017). The process that walks an executed path records the robot's pose (i.e., its location and the direction it is facing); for other paths, the direction is determined by projecting through the locations of future steps. The robot's pose plus the relative bounds of a landmark are sufficent for the model to estimate the acceptability rating of a description—in front of, behind, left, or right—for a spatial relationship. The highest-rated description is used.

The conversion is straightforward, but we present this level of detail to highlight the judgment calls necessary to make the problem of generating an explanation tractable. Locations are the main issue: the only locations generated for a problem instance are those relevant to the following of a given path, producing a few hundred locations in contrast to more than 5 million distinct cost map cells. Similarly, the entire region occupied by a landmark or an obstacle is not represented explicitly but only through relationships to path locations.

Explanations

Finally we reach explanations. Explanations can be divided into two types (Leddo & Abelson,1986). A *constructive* explanation is a direct application of the definition of scientific explanation, identifying the causal or logical factors that give rise to some result. *Contrastive* explanations make comparisons with one or more alternatives that may not be given explicitly. Generation of alternatives in general is challenging (Leddo & Abelson,1986) and we will assume that any alternatives are provided as input to the ASP inference process (as explained in more detail below).

A problem instance is generated as described for the path above. The ASP inference process fills in other predicates:

path(x_{4107},x_{4232}),
landmark(vul,x_{5446}),
at(x_{4107},1),move(x_{4107},x_{4108},1),at(x_{4108},2), ...
right(vul,58), ...
at(x_{4231},125),move(x_{4231},x_{4232},125),at(x_{4232},126),
finished(arrived(x_{4232},126))

Because the vulnerable region does not intersect the path, it is translated into a landmark; from the start until `T=58` the region is `in front of` the robot; it is to the `right` until `T=90` and thereafter `behind`. The necessary information for an explanation is provided, in general laws for the domain (e.g., requirements for and constraints on movement) and antecedents as specified in the problem instance. In words, "Why did the robot end up at(X_{4232},126)?" "Because it was at(X_{4231},125) [antecedent], there was a `step` [antecedent], it executed a `cmd_move` [antecedent], and the result was a `move` to that location [general law]." A set of such statements, working backward to the antecedent of the robot at the start of the path, constitutes a complete explanation representation.

For contrast, consider accounting for a hypothetical alternative path. In navigation, a straight line is a natural default for a human navigator (Korpan & Epstein,2018). Bresenham's algorithm is used to generate a straight-line sequence of cells on the cost map between source and goal locations. In the navigation scenario, with the vulnerable region given a `LETHAL_OBSTACLE` cost, the generation process puts obstacles on path locations inside the region. The inference process then generates the following:

path(x_{8037},x_{8292}),obst(vul,x_{8147}),obst(vul,x_{8148}),
at(x_{8037},1),move(x_{8037},x_{8038},1),at(x_{8038},2), ...
at(x_{8145},109),move(x_{8145},x_{8146},109),at(x_{8146},110),
finished(blockage(x_{8147},path(x_{8037},x_{8292})))

The inferred predicates describe the robot moving along the path until reaching a location just outside the vulnerable region; actions from that point `wait` until the time limit is reached. As with a constructive explanation, there exists a chain of domain laws and antecedents that account for this result. Said differently, it cannot be inferred that the robot reaches the goal location; it *can* be inferred that the robot will be at a different location at the time limit. The relevant domain law is the constraint that the agent cannot move to a location where there is an obstacle, which results in the inference of a blockage.

As a final example, we can change the cost of the region to `VULNERABLE` and use the same straight path. The explanation changes. In this case the robot can enter the region and that the path can be completed. As with the relationship to landmarks, the information that the robot was inside the vulnerable region is inferred, and this information forms the basis for comparison with other paths.

Discussion

We have presented a system for producing explanations of a robot's path planning and path following behavior. Part of our work is analytical. We adopted two well-known criteria for explanations: that they be understandable and that they provide a logical or causal account of a system's behavior. Image schemas satisfy the first criterion, by assumption; we have also presented evidence from the psychology literature that this assumption is plausible. Satisfying the second criterion involved showing that combinations of image schemas could be interpreted in logical or causal terms. The ASP formulation is a good match for the semantics of image schemas.

We are interested in the specific domain of navigation, though we expect to move next to consider the larger context of robot planning and acting. ASP supports inferences related to commonsense reasoning (e.g., that a landmark remains to my right from one step to the next as I move past).

Tradeoffs and limitions apply to our work. We call our explanations "lightweight" because the explanation is observational, not coupled with the robot's control processes. The main advantage of this approach is pragmatic: We can make many fewer assumptions about whether the robot is being controlled by a script, a state machine, an AI planner, or some other possibility. The disadvantage is that "why" inferences must be based on observations and domain knowledge without limited control information. For example, we can imagine a robot being programmed with a preference to pass on the left, with obstacles being observable on the right, but if this preference is not made public, the spatial relationship to an obstacle is descriptive rather than part of an explanation for why the robot chose its path.

Among the obvious limitations is the scope of the navigation task. It is reasonable to ask whether the representation and processing described in previous sections are necessary to explain such a simple activity. Activities with a cognitive component (in this case navigation tasks in general but also our use of image schemas for representation) do often turn out to be subtle underneath, but it will require human-robot interaction studies to evaluate the need for and the adequacy of explanations in this domain.

This limitation suggests another: representations are difficult to evaluate in the abstract. Mueller et al. (2019), in an extensive literature review of explainable AI, identify properties of good explanations and empirical techniques for evaluation. Our work produces representations but not the surface form of explanations, and we have not yet subjected them to evaluation. Once textual (or multimedia) explanations can be generated, human studies will be needed.

References

Balduccini, M. (2009). How flexible is answer set programming? An experiment in formalizing commonsense in ASP. In *Proceedings of the International Conference on Logic Programming and Nonmonotonic Reasoning* (pp. 4–16).

Cervel, M. S. P. (1999). Subsidiarity relationships between image-schemas: an approach to the force schema. *Journal of English Studies*(1), 187–208.

Chakraborti, T., Sreedharan, S., Zhang, Y., & Kambhampati, S. (2017). Plan explanations as model reconciliation: Moving beyond explanation as soliloquy. *arXiv preprint arXiv:1701.08317*.

Croft, W., & Cruse, D. A. (2004). *Cognitive linguistics*. Cambridge University Press.

Fox, M., Long, D., & Magazzeni, D. (2017). Explainable planning. *arXiv preprint arXiv:1709.10256*.

Frank, A. U., & Raubal, M. (1999). Formal specification of image schemata—a step towards interoperability in geographic information systems. *Spatial Cognition and Computation*, *1*(1), 67–101.

Gebser, M., Kaufmann, B., Kaminski, R., Ostrowski, M., Schaub, T., & Schneider, M. (2011). Potassco: The Potsdam answer set solving collection. *AI Communications*, *24*(2), 107–124.

Hempel, C. G., & Oppenheim, P. (1948). Studies in the logic of explanation. *Philosophy of Science*, *15*(2), 135–175.

Johnson, M. (1987). *The body in the mind: The bodily basis of meaning, imagination, and reason*. University of Chicago Press.

Korpan, R., & Epstein, S. L. (2018). Toward natural explanations for a robot's navigation plans. In M. de Graaf, B. Malle, A. Dragan, & T. Ziemke (Eds.), *Notes from the Explainable Robotic Systems Worshop, Human-Robot Interaction 2018.*

Kuhn, W. (2007). An image-schematic account of spatial categories. In *Proceedings of the International Conference on Spatial Information Theory* (pp. 152–168).

Lakoff, G. (1987). *Women, fire, and dangerous things*. University of Chicago Press.

Leddo, J., & Abelson, R. P. (1986). The nature of explanations. In *Knowledge structures* (pp. 103–122). LEA.

Li, J. J. (2012). Qualitative spatial and temporal reasoning with answer set programming. In *Proceedings of ICTAI* (Vol. 1, pp. 603–609). IEEE.

Lombrozo, T., & Carey, S. (2006). Functional explanation and the function of explanation. *Cognition*, *99*(2), 167–204.

Mandler, J. M., & Pagán Cánovas, C. (2014). On defining image schemas. *Language and Cognition*, *6*(4), 510–532.

Mueller, S. T., Hoffman, R. R., Clancey, W., Emrey, A., & Klein, G. (2019). Explanation in human-AI systems: A literature meta-review, synopsis of key ideas and publications, and bibliography for explainable AI. *arXiv preprint arXiv:1902.01876*.

Olson, E. (2011). AprilTag: A robust and flexible visual fiducial system. In *Proceedings of ICRA* (p. 3400-3407). IEEE.

Regier, T., & Carlson, L. A. (2001). Grounding spatial language in perception: an empirical and computational investigation. *Journal of experimental psychology: General*, *130*(2), 273.

Rosenthal, S., Selvaraj, S. P., & Veloso, M. M. (2016). Verbalization: Narration of autonomous robot experience. In *Proceedings of IJCAI* (pp. 862–868).

St. Amant, R., Morrison, C. T., Chang, Y.-H., Mu, W., Cohen, P. R., & Beal, C. (2006). An image schema language. In *Proceedings of ICCM* (pp. 292–297).

Timpf, S. (2002, Mar 01). Ontologies of wayfinding: a traveler's perspective. *Networks and Spatial Economics*, *2*(1), 9–33.

Walton, L., & Worboys, M. (2009). An algebraic approach to image schemas for geographic space. In *Proceedings of the International Conference on Spatial Information Theory* (pp. 357–370). Springer.

Ward, J., St. Amant, R., & Fields, M. (2017). Spatial relationships and fuzzy methods: Experimentation and modeling. In *Proceedings of ICCM.*

Wiener, J. M., Büchner, S. J., & Hölscher, C. (2009). Taxonomy of human wayfinding tasks: A knowledge-based approach. *Spatial Cognition & Computation*, *9*(2), 152–165.