Title: The Scope and Limits of Simulation in Automated Reasoning

Authors:
Ernest Davis, Dept. of Computer Science, New York University, New York, NY, davise@cs.nyu.edu, http://www.cs.nyu.edu/faculty/davise

Gary Marcus, Dept. of Psychology, New York University, New York, NY, gary.marcus@nyu.edu, http://www.psych.nyu.edu/gary/

Abstract:
In scientific computing and in realistic graphic animation, simulation — that is, step-by-step calculation of the complete trajectory of a physical system — is one of the most common and important modes of calculation. In this article, we address the scope and limits of the use of simulation, with respect to AI tasks that involve high-level physical reasoning. We argue that, in many cases, simulation can play at most a limited role. Simulation is most effective when the task is prediction, when complete information is available, when a reasonably high quality theory is available, and when the range of scales involved, both temporal and spatial, is not extreme. When these conditions do not hold, simulation is less effective or entirely inappropriate. We discuss twelve features of physical reasoning problems that pose challenges for simulation-based reasoning. We briefly survey alternative techniques for physical reasoning that do not rely on simulation.

The Scope and Limits of Simulation

in Cognition and Automated Reasoning

Ernest Davis and Gary Marcus

1. Introduction

Computer simulations – broadly speaking, computations in which the trajectory of a temporally evolving system is traced in detail – have become ubiquitous. Programmers have created extremely detailed simulations of the interactions of 200,000,000 deformable red blood cells in plasma (Rahimian, et al., 2010); the air flow around the blades of a helicopter (Murman, Chan, Aftosmis, & Meakin, 2003); the interaction of colliding galaxies (Benger, 2008); and the injuries caused by the explosion of an IED under a tank (Tabiei & Nilakantan, undated). Software, such as NVidia PhysX, that can simulate the interactions of a range of materials, including rigid solid objects, cloth, and liquids, in real time, is available for the use of game designers as off-the-shelf freeware (Kaufmann & Meyer, 2008). In artificial intelligence (AI) programs, simulation has been used for physical reasoning (Johnston & Williams, 2007), (Nyga & Beetz, 2012), robotics (Mombauri & Berns, 2013), motion tracking (Vondrak, Sigal, & Jenkins, 2008), and planning (Zicker & Veloso, 2009).

In cognitive psychology, likewise, simulation in a much broader sense has been proposed as the primary mechanism underlying physical reasoning, reasoning about other minds, language comprehension, and many other cognitive functions (Markman, Klein, & Suhr, 2009). For instance, Kaup, Lüdtke, and Maienborn (2010) suggest that "creating simulations is necessary for [the] comprehension" of language, and Battaglia et al (2013, p 18327) propose a model of physical reasoning

> based on an "intuitive physics engine," a cognitive mechanism similar to computer engines that simulate rich physics in video games and graphics, but that uses approximate, probabilistic simulations to make robust and fast inferences.

Similarly, Sanborn et al (2013) propose that "people's judgments [about physical events such as colliding objects] are based on optimal statistical inference over a Newtonian physical model that incorporates sensory noise and intrinsic uncertainty about the physical properties of the objects being viewed."

For many reasoning tasks, simulation is unquestionably a powerful tool and an intuitively appealing one; however, it is important to recognize its limitations. Here, we analyze the scope and limits of simulation as a technique of automated physical reasoning for general AI. (In a separate paper, we analyze corresponding limits in cognitive models (Marcus & Davis, in preparation.)) We develop a categorization of tasks where simulation works well and is plausible as a reasoning mechanism; tasks where simulation does not work at all, and is therefore impossible as a reasoning mechanism; and tasks where simulation could work, but other techniques are likely to be more effective..

2. Computer simulation of physical systems

In a typical simulation, the input is a detailed description of an initial scenario. The program then uses the dynamic laws of the domain to extrapolate an equally detailed description of the state of the scenario a short time later. The program continues to extrapolate each state to the next state until some stopping condition is met. The program returns the entire trajectory of states as its prediction of what will happen. Table 1 shows a description of this process in pseudocode.

```
function Simulate(StartingState; StartingTime; BoundaryConditions, TerminationCondition)
        s ← StartingState;
        t ← StartingTime;
        Trajectory ← [⟨t,s⟩];
        repeat {
            Δ ← chooseTimeStep(s);
            s ← projectForward(s, Δ ,BoundaryConditions);
            t ← t + Δ;
            add ⟨t,s⟩ to end of Trajectory;
          } until TerminationCondition;
    return Trajectory;
end
```

Table 1: Basic algorithm for physical simulation

For instance, consider a ball falling to the ground. In this simulation the initial state at a given time t is specified in terms of the height of the ball, x(t), and its velocity v(t), both measured upward. To extrapolate from one state at time t to the next state at time t+Δ, we calculate that the height decreases by Δ times the current downward velocity, and that the downward velocity increased by Δ times the acceleration of gravity, denoted g.

$$x(t+\Delta) = x(t) + \Delta * v(t)$$
$$v(t+\Delta) = v(t) - \Delta * g$$

The simulation stops when $x(t) \leq 0$, since at that point the ball has hit the ground.

Some simulations, such as assessments of the aerodynamics of airplanes, aim for high precision; they are extremely specialized in terms of both the physical phenomena and the kinds of scenario under consideration; and they involve immense computational burdens. Others, such as simulations used for real-time animation, particularly in video games, aim at plausible images rather than physical precision, often in real time on a personal laptop, rather than off-line on supercomputers.

AI programs that deal with physical objects often use simulations, with good reason. Although numerous technical difficulties exist (many described below), simulation is conceptually and methodologically simple and comparatively straightforward to implement. Furthermore simulation can be used directly to produce a viewable animation, which is very helpful both for the end-user and for program development and debugging; moreover, physics engines of ever-increasing power, quality, and scope are publicly available for use. In some circumstances, they represent an ideal solution.

3. Simulation: Challenges in Automated Systems

It is easy, however, for the non-expert to overestimate the state of the art of physical simulation, and assume that there is a plug-and-play physics engine that works for pretty much any physical situation.

Although physics engines are now commonplace in video games, when it comes to the real-world, their fidelity is often quite limited; plug-and-play engines capture only narrowly-defined environments; more sophisticated applications require hard work from experts. A few seconds of realistic CGI in a disaster film may well require several person-days of work; an accurate and complex scientific computation may require several person-months. Nils Thuerey (personal communication) writes,

> There are ... inherent difficulties with these simulations: we are still very far from being able to accurately simulate the complexity of nature around us. Additionally, the numerical methods that are commonly used are notoriously difficult to fine-tune and control.

Plug-and-play physics engines are also subject to bugs and anomalies,[1] and may require careful tuning to work correctly. In a systematic evaluation of seven physics engines, Boeing and Bräunl (2007) found that all seven gave significantly and obviously erroneous answers on certain simple problems involving solid objects.

In this section, we review twelve challenges that arise in the construction of simulations, some widely known, others less so; together, they help to articulate the scope and limits of when simulation can and cannot serve as an appropriate tool for automated physical reasoning.

3.1 The challenge of finding an appropriate modeling approach

The first hurdle in implementing a simulator is developing a domain model. In some cases, this is well understood. However choosing an appropriate model is often difficult, even for familiar objects, materials, and physical processes. The theory of physical systems that are changing temperature rapidly ("non-equilibrium thermodynamics") currently has very large gaps[2]; the theory of liquids and gasses also has significant gaps, including turbulence (Celani, 2007) and the interaction of flexible solids with liquids, such as occurs in swimming (Shelley & Zhang, 2011). Within the theory of rigid solid objects, which are the simplest kinds of materials encountered in everyday activities, the analysis of collisions has not been wholly resolved (Kaufman, Vouga, Tamstorf, & Grinspun, 2012) (Stewart, 2000).

Even in mundane situations, it may be challenging to find adequate models. Consider, for instance, cutting materials with tools. An ordinary household has perhaps a dozen kinds of tools for cutting: a few kinds of kitchen knives; more specialized kitchen equipment such as graters and peelers; a few kinds of scissors; a drill, a saw, a lawn mower, and so on. (A specialist, such as a carpenter or a surgeon, has many more.) Most people understand how they should be used and what would happen if you used the wrong tool for the material; if, for example, you tried to cut firewood with a scissors. But it would be hard to find good models for these in the physics or engineering literature.

3.2 The challenge of discretizing time

Most simulation algorithms employ a discrete model of time:[3] The timeline generally consists of a sequence of discrete instants; it is assumed that the system follows a uniform trajectory during the

[1] Many can be found on YouTube by searching on the name of the engine plus "Bug"; e.g. https://www.youtube.com/watch?v=Jl-A5YaiWi8 shows a bug with gyroscopic motion in ODE and https://www.youtube.com/watch?v=vfl33Tn0pYc shows a number of bugs in Skate 3.

[2] "Understanding the physics of nonequilibrium systems remains as one of the major challenges of theoretical physics." (Hurtado, Espigares, del Pozo, & Garrido, 2014)

[3] No possible representation can separately represent each instant of time in a continuous model. It is possible, nonetheless, for an automated reasoner to use representations to refer to a continuous model and axioms or inference rules that are valid over a continuous model; and these are qualitatively different from representations,

intervals separating the instants. In some instances, converting continuous physical time[4] into a discrete model is unproblematic, but in a surprisingly broad range of problems, difficulties arise from this conversion.

Consider, for example, the problem of choosing a proper time increment Δ in simulating rigid objects. If Δ is chosen too small, then many time steps must be calculated, increasing the computational burden. If Δ is too large, two objects may collide, interpenetrate, and pass through one another between one time point and the next. For instance, suppose that you are holding one compact disc (X) in the air, and you drop another (Y) directly on top of it from 1 meter above. By the time Y reaches X, it is travelling at a speed of about 4.5 m/sec. If the time increment is greater than a third of a millisecond, the collision will be missed (figure 1). As Boeing and Braunl (2007) demonstrated, current physics engines are not immune to this kind of error.

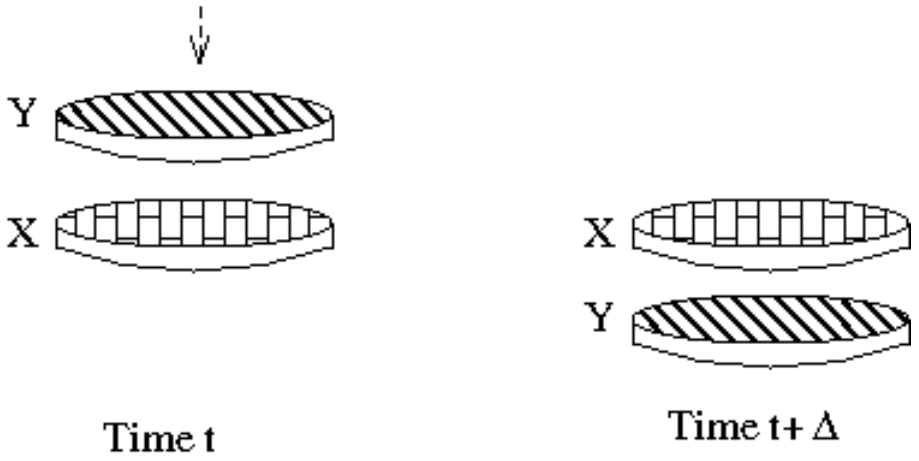

Figure 1: Object Y "passes through" X between successive instants

Alternatively, one can calculate exactly what is the time to the next collision or state change. But this kind of calculation can be extremely challenging.[5]

Discretizing time can also lead to more subtle problems. For instance if the simulation of a rigid pendulum uses a simple Euler method for the updating of the physical state from one time point to the next, the simulation will incorrectly predict that the pendulum swings back and forth a few times, reaching a higher angle on each swing, until eventually it rotates in a full vertical circle in one direction. A more sophisticated updating procedure is required to avoid this. In domains with richer physics than a pendulum, such as fluid dynamics, these kinds of problems can arise in much more complex forms (Bridson, 2008) (Hairer, Lubich, & Wanner, 2006). For instance, many basic methods for fluid simulation have the undesirable characteristic that the quantity of fluid in the system steadily decreases over time.

---

axioms, and rules based on discrete models. For instance, a differential equation refers to a continuous model of time whereas a difference equation refers to a discrete model of time.

[4] There has been work in physics on using discrete models of time. However, since these require a time quantum of about 10^{-43} seconds, these are irrelevant to the discretization of time used in simulations.

[5] It is not known to be decidable for rotating rigid objects.

3.3 The challenge of discontinuous dynamics

In some problems, a small change to the starting situation leads to a correspondingly small change in the overall trajectory. For instance, if you change the angle of a cannon by a small amount, the path of the ball changes only slightly. In other problems, two nearly identical starting situations can lead to significantly different behaviors. In these, enormous precision is required in both measurement and simulation to ensure an accurate answer. Consider the problem of a rolling die, which is the archetype of a physical process whose outcome is hard to predict, and in which slight differences in initial conditions can lead to entirely different outcomes. Although it is relatively easy to carry out an approximate computer simulation of a die rolling and to render it in an animation, it is extremely difficult to accurately predict the outcome of an actual roll of dice, even if the starting conditions are specified precisely (Kapitaniak, Strzalko, Grabski, & Kapitaniak, 2012).

A related problem is that simulators can make predictions that are correct in a mathematical sense but impossible physically because of instability. For example, all seven simulators tested by Boeing and Bräunl (2007) predicted incorrectly that if three spheres were dropped one exactly on top of the next, they would stack.

3.4 The challenge of choosing an idealization

Virtually all simulations represent idealizations; in some, friction is ignored, in others three dimensions are abstracted as two. In most situations, many different idealizations are possible; and the idealization should be chosen so that, on the one hand, the calculation is not unnecessarily difficult, and on the other, the important features of the situation are preserved. Consider, for instance, the simulation of a pendulum on a string. If you are using an off-the-shelf physics engine, then you would use one of the idealizations that the engine supports. For example, a typical engine might model the bob as an extended rigid shape and model the string as an abstract constraint requiring that the distance between the objects tied at opposite ends not exceed a fixed length. In that case, simulating the pendulum would require you to specify the structure of the pendulum, the mass and shape of the bob, the length of the string, and the initial position and velocity of the bob. A student in freshman physics, by contrast, will probably approximate the bob as a point mass which is constrained to move on a circle of fixed radius; the resulting simulation will certainly be easier to carry out and quite possibly more accurate. However, to set up the simulation in this way, the student must understand the essential form of the behavior in advance, viz. that the string remains fully extended while the bob swings.

Other scenarios are more complex. A bob may swing in a horizontal circle; spin on the axis of the string; rotate about its center of mass like a yo-yo, or fly through the air (figure 2). The string itself may be taut, loose, tangled, knotted, or twisted; it may get in the way of the bob; it may even unravel or snap. Although these behaviors are familiar to anyone who has spent time playing with objects on strings, few if any existing physics engines support any but the taut and loose conditions of the string and perhaps snapping.

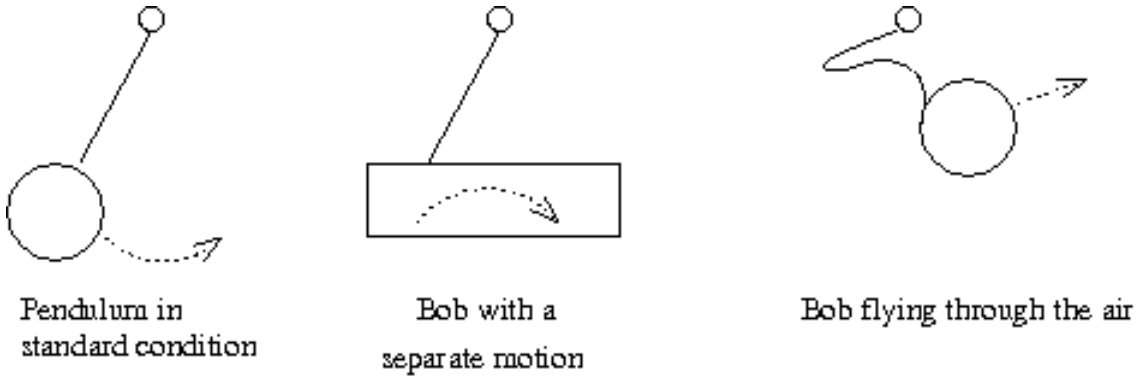

Figure 2: A pendulum in various conditions

Efficient reasoning about these different possible behaviors of the string and the bob requires using a variety of different idealizations. Space can be two-dimensional or three-dimensional. A bob can be idealized as a point object, a rigid object, or an elastic object. A string of length L can be idealized as an abstract constraint restricting the motion of the bob; a one-dimensional curve of length L, with or without mass; or a three-dimensional flexible object, either uniform or with some internal structure (e.g. twisted out of threads or a chain of links.) Influence on the system can be limited to gravity, or can include friction and air resistance. In looking at any one simulation that has been well-worked out, it is easy to lose sight of how much careful work goes on in choosing the right idealization; as yet there is no algorithmic way to guarantee an efficient solution for arbitrary problems. Using the most realistic model possible is no panacea; highly realistic models both require more laborious calculation and more detailed information.

Moreover, different high-quality physics engines can give radically difficult predictions for a single problem, particularly when the problem involves a device with feedback. Figure 3, from (Boeing & Bräunl, 2012) shows the paths calculated by three different physics engines for a submarine robot trying to follow a wall. These are radically different, even though, as Boeing and Bräunl emphasize, all three results come from “valid and accepted fluid simulation methods”.

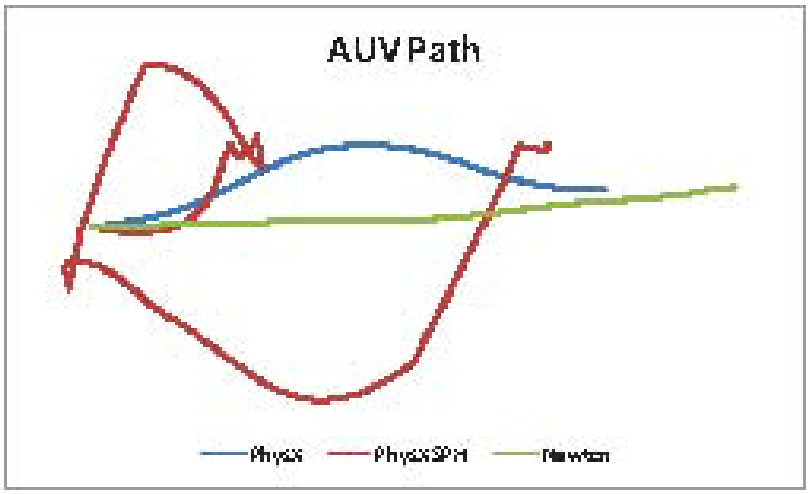

Figure 3: Alternative predictions by three different simulators. From (Boeing & Bräunl, 2012)

3.5 The challenge of rapidly drawing “easy” inferences

The output of a simulation is always precise,[6] though not always accurate. Often, however, the reasoner has no need for the level of precision that a simulation provides. Consider, for example a child who has built a tower of blocks and is now planning to knock it over by hitting it with another block. The child does not generally need to know details of the final disposition of each individual block nor a probability distribution over possible final dispositions; it may suffice to know that the tower will fall with a satisfying clatter. Likewise if you ride a bicycle on a bumpy road while carrying a half-full closed water canteen, all that matters is that the water stays inside the canteen, not the trajectory of the water splashing inside the canteen. In such cases, fully detailed simulations seem like inefficient and inappropriate tools.

There are many kinds of rules that allow quick inference or quick transference of results from a known situation to a new one. *Invariance under time and space*: If a mechanism worked in a particular way at home on Monday, it will work in the same way at work on Tuesday. *Invariance under irrelevant changes*: If a jar fits on a shelf, and you fill it with pennies, it will still fit on the shelf*. Invariance under changes of scale* (for certain physical theories, such as kinematics): A large pair of gears works in the same way as a much smaller scale model. *Approximation*: If a jug holds a gallon of water, then another jug of similar dimensions and shape will hold about a gallon. *Ordering on a relevant dimension*: If a toy fits in a box, then it will fit in a larger box, under a proper definition of “larger” (Davis, 2013). *Decomposition:* If a system consists of two uncoupled subsystems, then one can reason about each separately.

There can also be *rules of thumb*: useful generalizations for common cases. For instance, if you spill a cup of coffee in your office, it should not be necessary to resort to simulation to infer that the coffee will not end up in some other office. Rather, one can use a rule of thumb that a small amount of liquid dropped from a moderate "height onto a horizontal surface will end up not very far from the point where it was dropped, where "small amount", "moderate height", and "not very far" have some very approximate quantitative measurements.

Moreover, these alternative forms of inference may do well under circumstances where simulation scales badly. Consider the following scenario: you put a number of objects in a box, close the lid, and shake it up and down violently. We now wish to infer that the objects are still in the box. Simulating the motion of the objects will become rapidly more complicated as the number of objects increases, the shapes of the objects become more complex, and the shaking of the box becomes more complex and violent. By contrast a single simple rule, "An object in a closed container remains in the container" suffices to carry out the inference.

3.6 The challenge of incorporating extra-physical information

In some cases, reasoning about a physical system can be carried out more reliably and effectively using extra-physical information. Suppose that you see a baseball pitcher throw a ball. If you simulate the motion of the ball, using the best information you can gather about the angle, velocity, and so on, of the ball when it left his hand, and factor in the imprecision of this information, you will predict that it has a rather low probability of ending up anywhere close to the batter. You would obtain better results — predicting that the ball will end up close to the strike zone, just inside it or just outside — by relying on the known accuracy of the pitcher, plus quite specific information about the state of play and the pitcher's willingness to risk an out-of-strike zone pitch rather than a hit.

[6] Up to the precision of the underlying representation. For instance, numerical values are typically precise up to the limits of floating point precision; spatial representations are precise up to the limits of the particular spatial representation used.

3.7 The challenge of incomplete information

Carrying out a physical simulation is generally only possible if the geometric and physical characteristics of the initial condition are known precisely. In many common real-world situations, reasoning must be carried out on the basis of partial, sometimes extremely limited, information. Perception may be imperfect or incomplete. For instance, an object may be partially occluded. (An opaque object always self-occludes its own far side from the viewer.) Knowledge of the physical situation may come from natural language text or sketches. Knowledge of aspects of the situation may come from inference; for example, if you see someone attempt unsuccessfully to pick up a suitcase, you can infer that the suitcase is unusually heavy; you can then use that inference for future prediction. Or the precise details may not have been determined yet. For instance, suppose that you are going to the furniture store to buy a dining room table. You can reason that you will not be able to carry it home on foot or riding a bicycle, even though you have not yet chosen a particular table.

Of course, no representation is truly complete or entirely precise; in any representation, some aspects are omitted, some are simplified, and some are approximated. However, the simulation algorithm requires that the initial conditions of the scenario be *fully specified relative to a given level of description*. That is, the representational framework specifies some number of critical relations between entities and properties of entities. A complete representation of a situation relative to that framework enumerates all the entities that are relevant to the situation, and specifies all the relations in the framework that hold between those entities. The description must be detailed and precise enough that the situation at the next time step is likewise fully specified, in the same sense.

3.7.1 Limits to Monte Carlo simulation as an approach to partial knowledge

One common method for using simulations in situations with partial knowledge is to use Monte Carlo methods, or probabilistic simulation (Hoffman & Schreiber, 2002) (Pang, 2006). Given a probability distribution over a space of physical situations, a program generates some large collection of specific situations following the distribution; it carries out a simulation based on each of these; and then it considers these behaviors as a random sample of the possible behaviors. Then for any particular proposition Φ of interest, the probability that Φ will be true in the actual situation can be estimated as the fraction of simulations in which Φ is true. Conditional probabilities can be computed in a similar way: to calculate the conditional probability $P(\Phi|\lambda)$, generate random scenarios satisfying λ and find the fraction of these that satisfy Φ. This technique can be adapted to situations in which one starts with a set of constraints rather than a distribution by using a distribution that is, in some sense, uniform over the space of situations satisfying the constraints.

For example, Battaglia, Hamrick, and Tenenbaum (2013) consider a situation where an observer is looking at a tower of blocks, but has imperfect information; specifically, the perceived position of the block varies from the true position by an error that follows a Gaussian distribution. Their program uses this distribution to generate a collection of different towers, and simulates the behavior of each of these hypothetical towers. The probability that the true tower is stable is estimated as the fraction of simulations in which the tower remains stable. Conditional probabilities such as "the probability that the blue block and red block end up less than an inch apart, given that the tower falls over" can be estimated by generating random towers, selecting those in which the tower falls over, and then, within that restricted set, considering the fraction in which the blue block and red block end up less than an inch apart.

Probabilistic simulation can often be effective, but it has its own problems. First, the concept of a random shape is problematic. If you know that a block is a rectangular solid, then you can reasonably specify the distributions of its length, width, and height as distributions of some standard form (e.g. Gaussians), or collectively as a joint distribution. However, if you have imperfect knowledge of the *shape* of the blocks, then it is much less clear how to proceed. There does not seem to be any natural probability distribution over the class of all shapes.

Second, Monte Carlo simulation runs the risk of missing scenarios that are important and possible, though improbable.Suppose, for example, that you are standing underneath a scaffold next to a building, and you observe that a tool on the scaffold is precariously balanced. You wish to know whether you are safe, so you carry out a number of simulations of the tool falling off the scaffold. If the tool does not hit you in any of them, you conclude, erroneously, that there is no need to worry.

Third, the calculation of the conditional probability $P(\Phi|\lambda)$ relies on generating random elements satisfying λ. However, if λ is only rarely satisfied, it may be difficult to generate these. For instance, suppose that you see a nut in a bolt, and you have reliable information that the nut fits properly in the bolt (that is, after all, a safe default assumption when a nut is on the bolt). You cannot, however, see the threads of either. You wish to infer something about the number of turns needed to remove the nut from the bolt; almost certainly greater than half a turn and almost certainly less than 50 turns. That is, you want to estimate $P(\Phi_k|\lambda)$ where Φ_k is the proposition, "It will require exactly k turns to remove the nut from the bolt" and λ is the proposition, "The nut fits on the bolt".

If you sample random shapes for the nut and the bolt separately, then you will never come across a pair in which the nut fits on the bolt; and if you have no samples for λ, you certainly cannot estimate $P(\Phi_k|\lambda)$. Rather, you must generate the shapes for the nut and bolt concurrently. But at this point you need a quite specialized Monte Carlo shape generator that can generate fitting nuts and bolts simultaneously. If the generator is to be created by the automated reasoner, and not supplied by a human programmer, then that in itself would require powerful spatial and physical reasoning capacities that can hardly be the product of unaided simulation. (The problem of generating distributions corresponding to low-probability distributions is central to the implementation of probabilistic programming languages such as Church (Goodman, Masinghka, Roy, Bonawitz, & Tenenbaum, 2008); but it is not clear how well the techniques used there will adapt to the geometric and physical constraints involved in the kinds of examples weare considering here.)

Finally, the assumption that ignorance can always be characterized in terms of a probability distribution is inherently questionable. There is often no particularly reasonable or well-motivated way to assign probabilities to alternatives if these arise simply from ignorance of what is going on. The engineering literature (e.g. (Ang & Tang, 2007)) makes the useful distinction between aleatory uncertainty (i.e. uncertainty due to chance) and epistemic uncertainty (uncertainty due to ignorance); and caution is advised in assuming that it is meaningful to apply probabilistic methods to epistemic uncertainty.

3.7.2 Reasoning with incomplete information about containers

One important domain of physical reasoning where reasoning with incomplete information is both common and effective is in reasoning about containers — boxes, bottles, cups, bags, and so on — and their contents (Davis, Marcus, & Chen, 2013). Containers are ubiquitous in everyday life, and children learn to use containers at a very early age.

A container can be made of a wide range of materials: rigid materials, paper, cloth, animal body parts, or combinations of these. The only requirement is that the material should maintain its shape to a sufficient degree that holes do not open up through which the contents can escape. Under some circumstances, one can even have a contained space whose bottom boundary is a liquid; for instance, an insect can be trapped in a region formed by the water in a basin and an upside-down cup. It can also have a wide range of shapes. The only constraint on the shape of a closed container is that there exists an internal cavity. The only constraint on an open container is that the material of the container plus a horizontal surface at its top delimit an internal cavity. Either a closed or an open container may additionally have small holes that penetrate from inside to outside e.g. a cage or a sieve.

The material of the contents of a container is even less constrained. In the case of a closed container, the only constraint is that the material of the contents cannot penetrate or be absorbed into the material of the container (e.g. you cannot carry water in a paper bag or carry light in a cardboard box); and that the contents cannot destroy the material of the container (you cannot keep a gorilla in a balsa wood cage). Using an open container requires additionally that the contents cannot fly out the top (Davis, 2011). Using a container with holes requires that the contents cannot fit or squeeze through the holes.

Those are all the constraints. In the case of a closed container, the material of the contents can be practically anything.

If the only goal is to infer that the object remains inside the container, then simulation is at best extremely inefficient and at worst essentially impossible. Suppose, for example, that you have an eel inside a closed fish tank and you wish to infer that it remains in the tank. If we are to solve this problem by probabilistic simulation, we would need, first to understand how an eel swims, and second to simulate all kinds of possible motions of the eel and confirm that they all end with the eel inside the tank. If we do not know the mechanisms of swimming in eels, then the only way to use probabilistic simulation is to generate some random distribution of mechanisms that eels might use. Clearly, this is not plausible.

Another, similar example: You pack a shirt in a suitcase, you lock the suitcase, you check it onto a flight to Chicago. The suitcase is lost. Three days later, it turns up at the Dallas airport. It’s pretty banged up, but intact. Who knows what they did to it or how it got there. However, if it’s still locked, you can be sure that the shirt is still inside. In this case, it is impossible to effectively simulate because there is no information about the exogenous events involved.

3.7.3 Reasoning with incomplete knowledge about people, animals, and robots

Much of the physical reasoning that is important for any intelligence, natural or artificial, has to do with the physical capacities and limitations of physical agents that can move autonomously — people, animals, or robots. In animals, especially, capturing prey and evading predators is one of the most important functions of cognition.

There is at this point a large industry in developing simulators both for robots, to help in planning and to lower costs of testing (Koenig & Howard, 2004) (Laue, Spiess, & Rofer, 2006), and for various aspects of human activity and human physiology, mostly for medical purposes (Loan, Cagatay, & Rosen, 1997). However, these require detailed knowledge of (generally very complex) physical characteristics of the creature. Moreover, if one wants to simulate how the creature will behave, then it is additionally necessary to simulate its perception and possibly also the control feedback that connects perception to action.

For example, suppose that a machine reading program wishes to understand the following short text (from the Wikipedia article on ``Camoflague'')

> Some animals, such as the Horned Lizards of North America, have evolved elaborate measures to eliminate shadow. Their bodies are flattened, with the sides thinning to an edge; the animals habitually press their bodies to the ground; and their sides are fringed with white scales which effectively hide and disrupt any remaining areas of shadow there may be under the edge of the body.

The reader can perhaps simulate the appearance of the sunlit lizard on the ground, and understand how this results in there being no visible shadow. However, to understand the passage, one also has to understand why it is important not to cast a shadow; which requires understanding that an implicit predator, of entirely unspecified characteristics, would be able to detect the presence of the lizard, and catch and eat it, if it saw the shadow, but will not detect the presence and therefore cannot catch it if there is no shadow. One could perhaps simulate that as well, but obviously the reasoning needed to figure out that this simulation is relevant and to set up the simulation is much more complex than the simulation itself.

3.7.4 The physical dynamics of unknown entities

Suppose that you are walking along the beach and you see an oddly shaped green object. The object has a cavity at the top, and inside the cavity is some white stuff. You want to know what will happen if you try to lift the green thing, putting one hand on either side.

Since you have no idea what the green thing or the white stuff is, you obviously cannot predict very much about this. Still you are not entirely at a loss for conclusions to draw. It is very unlikely that picking up the green thing will cause the white stuff to fly up twenty feet into the air or will cause a large sinkhole to appear under your feet. It is impossible that the green thing will turn out to be a mountain or that the white stuff will turn into a canary.

With some more observation, further conclusions can be drawn. If the green thing has sharp edges, then it is probably not soft or malleable, and therefore is likely to preserve its shape when you pick it up, unless it consists of multiple pieces, in which case you might be able to see cracks between the pieces. If the white stuff has a horizontal top surface and fills the cavity below the top surface, then it may well be a liquid. Simulation-based reasoners cannot readily draw these sorts of inferences.

3.8 The challenge of irrelevant information

In physical reasoning, as in most forms of reasoning, it is important to focus on the relevant aspects of the situation and ignore the irrelevant.

In modeling and simulation, irrelevancies are generally excluded manually. The human being who sets up the simulation formulates the problem in terms of the relevant features and ignores the irrelevant ones. However, this expedient will not do for a general AI program or a cognitive theory. There must be a well-defined process to exclude irrelevant information in physical reasoning, and it is hard to see how this exclusion process itself can rely on simulation.

Suppose for example that you are eating a picnic in the middle of a country fair. You are pouring lemonade from a thermos, while all around you the fair is bustling; the Ferris wheel is turning, carnival touts are barking, hot dog vendors are grilling hot dogs and so on. In most cases, in reasoning about pouring the lemonade, you can safely ignore all this other activity.

As a second example, let us return to the problem of automatically generating samples of nuts and bolts discussed in section 3.7. Suppose that you have some irrelevant information about the shape of the bolt: it is a regular heptagon and the manufacturer's monogram is engraved on one side. If this is included as part of the given information λ, then the problem of generating random samples satisfying λ becomes that much harder.

3.9 The challenge of range of scales

Consider the following scenario. At night, you look up at the Little Dipper, and see Polaris, the North Star. However, if you close your eyes, then you don't see Polaris.

How do we know this? Although in principle it would be possible to compute this using a physics engine, in practical terms, such an approach is wildly unwieldy. The natural way to use simulation to understand this scenario is to simulate a photon being emitted in a random direction from Polaris, and then keep simulating until you have a respectable collection of photons that meet the pupil of the observer. (A minimum of three photons are needed just to get the correct color.) The trouble derives from the scales involved; to see Polaris at its observed brightness and color because every 100 ms, about 500,000 photons from Polaris reach your pupil. Polaris is 433 light-years away, so these were emitted in 1580 AD. The probability that a particular photon emitted from Polaris will meet one particular 1 cm^2 pupil is about $3*10^{-42}$; even with high-speed computers, it is not practical to compute this by tracking individual photons over such vast scales .[7]

Less straightforward solutions that still depend primarily on simulation raise their own complications. Suppose that we can avoid that problem of probabilistic simulation and directly generate a simulation of a photon that starts at Polaris and travels in a straight line to the eye. This involves the problem of interpolating a simulation with constraints at multiple time points, discussed in the next section. Even so, the problem of discretizing time remains. If we discretize time into equal time-steps, then to detect the interaction of the photon with the 1 mm thick eyelid would require a time step of 10^{-11} seconds, and therefore about 10^{20} time steps would be needed to span the 400 years. In all but one of these time steps, the photon moves forward 1 millimeter through empty space or the atmosphere. Clearly this is not feasible. Rather the simulation must represent only the time points when something interesting happens. One can preserve a small role for simulation, but most of the work then goes into decide what to include in the simulation; executing the simulation is providing at most a very small final piece in the puzzle.

Of course, there is no difficulty in computing the appearance of the night sky; simple astronomy programs can do that. The difficulty is in integrating this with reasoning about objects that might interfere with passage of light. For example, an astronomy student should be easily able to determine, or infer, that

if he closes his eyes, he will not see Polaris; that an mile-wide asteroid in the asteroid belt could occlude Polaris, but only for a fraction of a second; that a mile-wide object orbiting Polaris would not occlude Polaris, and so on.

[7] The probability of a successful simulation is much higher if we go backward in time from the eye to Polaris, along the lines of the "ray-tracing" algorithms of computer graphics. If we use the maximum angular resolution of the human retina in the dark, and trace a line backward, the probability that the line will intersect Polaris is about $4*10^{-11}$. (Thanks to Rob Fergus and Yann LeCun for their assistance.)

Reasoning that integrates wide ranges of scales is common in science. Stellar evolution is related to nuclear reactions; this involves a mass ratio of about 10^{57}. Species evolution is related to genes; this is a factor of about 10^{34}. Everyday life also involves scale ratios that are large, though not astronomical. You are setting out for a trip to California of 3000 miles, and you turn the keys in the ignition through a distance of 1 cm; this is a ratio of $5*10^{8}$. A soldier is hit by a bullet in an event taking 10^{-3} seconds and carries it for the rest of his 70 year lifetime; this is a factor of $2*10^{12}$.

Specialized simulation techniques have been developed that can deal with a wide range of scales; for example (Vogelsberger, et al., 2014) described a simulation of cosmological evolution that spans 6 orders of magnitude in linear dimension, from 100 megaparsecs to 48 parsecs. But it should be possible to carry out the simple inferences described above without calling on this very sophisticated, very computationally demanding technology.

3.10 The challenge of tasks other than prediction

Simulations are most readily used for the task of generating predictions, such as where a pendulum will be at a given instant. There are, however, many other important kinds of physical reasoning tasks, to which simulation-based techniques are in general less well suited. Some examples:

Interpolation between two states at two different times: Looking at Polaris is a simple example. The photon left Polaris in 1580 AD; it arrives at the eye now; one wishes to interpolate between the two.

Planning: Given a starting situation and a goal, find a plan that, when executed in the starting situation, will bring about the goal. For example, given a pitcher full of water and an empty glass in the starting situation and goal of having water in the glass, plan to pour the water from the pitcher into the glass. Most simulations are ill-equipped to develop plans of this sort, and Monte Carlo sampling from a vast space of possible plans seems like an unrealistic solution.[8]

Inferring the shape of an object from its physical behavior: For instance, a human who observes that a bucket of water is slowly dripping from the bottom can readily infer that there is a small hole. A simulation-based reasoner can only derive such an inference by using a supervisory system that samples from a large number of possible buckets with arbitrary alterations.

Inferring the physical properties of an object from its physical behavior: For example, if an object is blown away by a gentle wind, a human reasoner can readily infer that it is a light object. Few simulation systems are set up to derive this sort of inference.

Design: Construct an object to achieve a specified functionality. Suppose, for instance, you want to design a terrarium for your pet toad that will sit on the window sill. One can immediately infer a number of constraints; it can't be completely closed, or the toad will suffocate; it can't be open on top, or the toad will jump out; it must be possible to get food and water to the toad and clean the terrarium; it needs to have a flat bottom; it needs to fit on the shelf; and so on. These constraints must be derived *before* the design and dimensions of the terrarium can be chosen or simulated.

Comparative analysis: How does a modification to a problem affect the solution? For instance, you can reason that if you make an object heavier, then it will become more difficult to lift. This inference can be done without knowing the actual weight of the object (de Kleer & Brown, 1985).

[8] A salient counterexample is the program described in (Zicker & Veloso, 2009) which constructs plans for playing soccer and miniature golf by a heuristic search through the space of actions, using simulation to predict the behavior of the ball when kicked or putted. However, the vocabulary of actions and of physical behaviors is small and high precision is required. It is not likely to yield a generally applicable solution to the problem of planning.

3.11 The challenge of the frame problem

Simulation theories, might at first blush, seem to be immune to the so-called frame problem – of efficiently reasoning about the aspects of a situation that do not change over time-- that has often bedeviled knowledge-based theories (McCarthy & Hayes, 1969); in reality, the frame problem arises in simulation in very much the same ways and to the same extent as in knowledge-based theories, avoidable In simple cases (Shanahan, 1994; Reiter, 2001) daunting in many more complex cases.

An example of a case where simulation runs into the frame problem is as follows. Consider a situation where there are two stable towers of blocks on the ground with a substantial space between them, tower A and tower B. You now drop a block onto tower A. It is immediately obvious to a human onlooker that whatever happens in tower A, tower B will be unaffected. To determine this, a physics engine, such as we discuss in sections 2 and 3, would need compute all the forces between blocks in tower B, and calculate that the net force and net torque on each block is zero. Moreover, in order to accurately predict how tower A will collapse, the engine would need to use a very fine discretization of time with many time steps; and it would need repeat this identical calculation over tower B at each time step. The key hallmark of the frame problem here is the needless repetition of the calculation that nothing in tower B changes ; in particular categories of cases, it may be possible to recognize that this part of the situation is repeated, and to retrieve the previous results rather than redo the calculation; but as the situation becomes more complicated, this rapidly becomes increasingly difficult to do.

3.12 The challenge of using common sense to check simulations

As we have seen, technical problems in simulations can give results that are not merely wrong in detail, but physically nonsensical. If the wrong time increment is used in simulating solid objects, the simulation can predict that one object passes through another. If a simulation of a pendulum uses the wrong updating rule, it will predict that the pendulum swings higher and higher (section 3.2).

If there is a human in the loop, they can examine the output of the simulator and at least see that this cannot be right, particularly if the simulator outputs an animation. If there is no human in the loop, then it would be very desirable for the physical reasoning system to be able to detect these kinds of problems by itself. If the simulation is the only reasoning mechanism, this cannot be done.

There have been a number of reasoning systems that uses qualitative reasoning to exclude Modelica or Matlab simulations with nonsensical outcomes (Klenk, Bobrow, de Kleer, & Jansenn, 2014).

3.13 Summary

Simulation is effective for automated physical reasoning when the task is prediction, when complete information is available, when a reasonably high quality theory is available, and when the range of scales involved, both temporal and spatial, is not extreme. If these conditions do not hold, then simulation becomes to some extent problematic. We have identified many categories of problems where these conditions do not hold, and demonstrated by examples that these categories include many natural and simple examples.

Furthermore, even when these conditions do hold, there may be simpler approaches either based on other kinds of physical knowledge or based on non-physical knowledge.

Probabilistic simulation can be effective when a plausible probability distribution can be posited, and when it is easy to generate a random sample of instances satisfying the known constraints. In many cases these conditions are not met. If the given information is too weak, as in the cases of objects of unknown shape or material, then there may be no natural probability distribution. If a complex constraint

is known, as in the case of the screw and bolt that fit together, then it may be difficult to generate a random sample satisfying the constraint.

Table 2 enumerates the categories of challenges we have discussed, and reviews the examples.

Some of these problems are well-discussed in the literature of modeling and simulation; others not. The problems of finding a dynamic model, of discretizing time, and of discontinuity are discussed very extensively; these are central issues in the theory of modeling and simulation, and there is a large technical literature that addresses them. The problems of choosing an idealization and of checking that simulations make sense are discussed, but generally in informal terms e.g. (Robinson, 2004) (Willemain, 1995). The problems of incomplete information, irrelevant information, range of scale, non-predictive tasks, and the frame problem have been examined only to a limited degree, and the technical literature within the theory of modeling and simulation is small.[9] The issues of easy inference or of inference from non-physical information are hardly discussed. For simulation to serve as a *general* solution to the problem of physical reasoning, in either human or automated form, *all* would need to be solved. At present, no actually implemented automated system has come close.

[9] There is, of course, a large AI literature on incomplete information, irrelevant information, and the frame problem, but these challenges have largely left unaddressed in the modeling and simulation literature

Challenges	Examples
Finding an appropriate model	Tools for cutting.
Discretization of time	Rigid objects passing through one another; pendulum swinging higher and higher
Discontinuity and instability	Dice, Stacking spheres
Choosing an idealization	Pendulum
Easy inference	Knocking over a tower of blocks, containers, spilling a cup of coffee
Non-physical information	Baseball pitch
Incomplete information	Bolt on screw, containers, unknown entities
Irrelevant information	Picnic at a fair, irrelevant shape details
Range of scale	Astronomical examples
Tasks other than prediction	Planning, inferring shape and properties, design
Frame problem	Separated towers of blocks
Checking that simulations make sense	Rigid object, pendulum

Table 2: Challenges of simulation, with examples drawn from this paper

4. Alternatives to simulation

What alternatives are there? One possibility is a knowledge-based reasoning engine in which both the knowledge of the domain physics and the problem specifications are largely declaratively expressed. Several different aspects of knowledge-based physical reasoning have been developed in the AI literature (see (Davis, 2008) for an overview):

- Symbolic reasoning based on a theory of the physical domain. The theory may be composed of valid qualitative rules such "An object inside a closed container remains inside." (Hayes, 1979; 1985; Davis, 1988; 2011); or it may be an "intuitive" or "naïve" theory such as an impulse theory of motion.
- Qualitative reasoning: A collection of algorithms developed for reasoning about relative size of physical quantities and direction of change (Forbus, 1985), (Kuipers, 1986) (Weld & de Kleer, 1989). For instance, these algorithms support the qualitative inference that, if two objects at different temperatures are in thermal contact, then heat will flow from the hotter to the colder until they reach the same temperature.

- Reasoning by analogy (Gentner & Forbus, 2011). For instance the analogy between pressure and flow in hydraulic systems to voltage and current in electronic systems allows the reasoner to conceptualize electric circuits in terms of water flow, which is more familiar.
- Meta-level reasoning; reasoning explicitly about the structure and use of theories (Lehmann, Chan, & Bundy, 2012), (Forbus, 1985) (Kuipers, 1986) (Falkenhainer & Forbus, 1988) (Weld, 1990) (Weld, 1992) (Nayak, 1994). For instance, one can use a rule that states that the theory of rigid solid objects will suffice to describe wooden blocks in the context of a child playing, but not in the context of architecture.

These different approaches are largely complementary rather than in opposition; they deal with different aspects of reasoning; a complete theory would presumably involve integrating these, together with further techniques for other issues that have not yet been addressed.

Qualitative reasoning has been by far the most extensively studied, and achieved some notable successes, e.g in text understanding ,(Kuehne (2004) and analogical mapping and geomeytic reasoning (Lovett et al, (2009). The Flame system (Price, Pugh, Wilson, & Snooke, 1995) for failure mode and effects analysis in automotive electrical systems has been used by multiple companies for years. Fromherz, Bobrow, and de Kleer (2003) discuss a system for automatically generating model-based control strategies for high-end printing systems.[10] Other suggestive analogues in other forms of reasoning include non-simulative planning methods such as partial-order planning (Chapman, 1987) (Weld, 1994) and hierarchical planning (Erol, Hendler, & Nau, 1994) and on symbolic reasoning for verification of computer software (Cousot, 2012), such at the software that controls complex physical systems such as airplanes (Souris & Delmas, 2007); the latter inherently combines physical reasoning with complex logical reasoning.

Highly trained physical reasoning, such as catching a fly ball or predicting the trajectory of a billiard ball with spin, probably also involves interpolation from a database of stored examples, derived from personal experience combined with using a tight perception-action feedback loop, learned using reinforcement learning or some similar strategy.

5. Conclusion: The Limits of Simulation

With infinite computational power, infinite time and unlimited memory, perhaps anything could be simulated, from the bottom up, much as Pierre Simon Laplace (Laplace, 1814) once imagined,

> An intellect which at a certain moment would know all forces that set nature in motion, and all positions of all items of which nature is composed, if this intellect were also vast enough to submit these data to analysis, it would embrace in a single formula the movements of the greatest bodies of the universe and those of the tiniest atom; for such an intellect nothing would be uncertain and the future just like the past would be present before its eyes.

But in the real world, computational power is finite, decisions must be made in finite time, and memory resources are limited. Neither the mind nor an automated problem solver can hope to model the

[10] Admittedly, rule-based systems as applied to physical reasoning have had only limited success in either automated commonsense reasoning or cognitive modeling and in particular few successful practical applications; possible reasons for this are discussed in (Davis, 1998).

interactions of everyday objects, let alone the interactions of complex agents like human beings, by simulations that originate at a quantum or even a molecular level.

Instead, real-world simulations that run in realistic time with reasonable resources must use approximations and idealization at a higher level of abstraction. In many cases, setting up the simulation — choosing the approximations and idealizations appropriate to the problem — and interpreting the output of the simulation — deciding how accurate the results of the simulation are likely to be, which parts of the simulation are valid, and which parts are artifacts of the idealization — can be much more difficult than executing the simulation. It is simply naïve to imagine that, for example, an automated reasoner could infer all that it needs to know about physics by running a Newtonian physical simulator on every entity that it encountered. With the many examples that we have reviewed in this paper, we hope to have made clear that the full-simulation view of cognition is entirely unrealistic and by the same token that simulation, however precise, is equally unlikely to solve the problems of automated commonsense physical reasoning.

At this juncture, it is difficult or impossible to quantify what fraction of the physical reasoning that would be needed for general artificial intelligence or for specific applications such as household robots or narrative understanding could be carried out using simulation. We have argued, however, that the range of examples we have presented in this paper suggests that there are significant limits to the use of simulation. In particular, although we have suggested that simulation is effective for physical reasoning when the task is prediction, when complete information is available, when a reasonably high quality theory is available, and when the range of spatial or temporal scale involved is moderate, in many other cases simulation is to some extent problematic. In particular, physical reasoning often involves tasks other than prediction and information is often partial.

Moreover, even when these conditions hold, there are many cases in which it would appear that alternative non-simulative modes of reasoning are likely to be easier, faster, or more robust. Finally, setting up and interpreting a simulation requires modes of physical reasoning that are not themselves simulation. For all these reasons, we suggest that non-simulative forms of reasoning are not an optional extra in automated physical reasoners but are centrally important.

Acknowledgements

Thanks to Peter Battaglia, Benjamin Bergen, Thomas Bräunl, Alan Bundy, Garnet Chan, Jonathan Goodman, Philip Johnson-Laird, Ken Forbus, Casey McGinley, Andrew Sundstrom, and Ed Vul for helpful discussions; and to Nils Thuerey for information about simulation for film CGI.

Bibliography

Ang, A., & Tang, W. (2007). *Probability Concepts in Engineering, Planning, and Design* (2nd ed.). New York: Wiley.

Battaglia, P., Hamrick, J., & Tenenbaum, J. (2013). Simulation as an engine of physical scene understanding. *Proceedings of the National Academy of Sciences, 110*(45), 18327-18332.

Benger, W. (2008). Colliding galaxies, rotating neutron stars, and merging black holes -- visualizing high dimensional datasets on arbitrary meshes. *New Journal of Physics, 10*. doi:http://dx.doi.org/10.1088/1367-2630/10/12/125004

Boeing, A., & Bräunl, T. (2007). Evaluation of real-time simulation systems. *GRAPHITE.* doi:http://dx.doi.org/10.1145/1321261.1321312

Boeing, A., & Bräunl, T. (2012). Leveraging Multiple Simulators for Crossing the Reality Gap. *12th International Conference on Control, Automation, Robotics, and Vision.* Guangzhou, China. From http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=06485313

Bridson, R. (2008). *Fluid Simulation for Computer Graphics.* Wellesley, Mass.: A.K. Peters.

Celani, A. (2007). The frontiers of computing in turbulence: Challenges and perspectives. *Journal of Turbulence, 8*.

Chapman, D. (1987). Planning for conjunctive goals. *Artificial Intelligence, 32*(3), 333-379. doi:http://dx.doi.org/10.1016/0004-3702(87)90092-0

Cousot, P. (2012). Formal verification by abstract interpretation. In A. Goodloe, & S. Person (Ed.), *4th NASA Formal Methods Symposium. 7211*, pp. 3-7. Norfolk, Virginia: Springer, Lecture Notes in Computer Science. From http://www.di.ens.fr/~cousot/publications.www/Cousot-NFM2012.pdf

Davis, E. (1988). A logical framework for commonsense predictions of solid object behavior. *AI in Engineering, 3*(3), 125-140. doi:http://dx.doi.org/10.1016/0954-1810(88)90029-5

Davis, E. (2008). Physical Reasoning. In F. van Harmelen, V. Lifschitz, & B. Porter, *Handbook of Knowledge Representation* (pp. 597-620). Elsevier.

Davis, E. (2011). How does a box work? *Artificial Intelligence, 175*(1), 299-345. doi:http://dx.doi.org/10.1016/j.artint.2010.04.006

Davis, E. (2013). Qualitative spatial reasoning in interpreting text and narrative. *Spatial Cognition and Compujtation*. From http://www.cs.nyu.edu/faculty/davise/papers/cosit.pdf

Davis, E., Marcus, G., & Chen, A. (2013). Reasoning from Radically Incomplete Information: The Case of Containser. *Advances in Cognitive Systems*, (pp. 273-288).

de Kleer, J., & Brown, J. (1985). A qualitative physics based on confluences. In D. Bobrow, *Qualitative Reasoning about Physical Systems* (pp. 7-84). Cambridge: MIT Press. doi:http://dx.doi.org/10.1016/0004-3702(84)90037-7

Erol, K., Hendler, J., & Nau, D. (1994). UMCP: A sound and complete procedure for hierarchical task-network planning. *Proceedings, International Conference on AI Planning Systems*, (pp. 249-254).

Falkenhainer, B., & Forbus, K. (1988). Setting up large-scale qualitative models. *Procceedings of the National Conference on Artificial Intelligence (AAAI-88).* From http://www.aaai.org/Papers/AAAI/1988/AAAI88-054.pdf

Forbus, K. (1985). Qualitative process theory. In D. Bobrow, *Qualitative Reasoning about Physical Systems* (pp. 85-168). Cambridge, Mass.: MIT Press. doi:http://dx.doi.org/10.1016/0004-3702(84)90038-9

Fromherz, M., Bobrow, D., & de Kleer, J. (2003). Model-based Computing for Design and Control of Reconfigurable Systems. *AI Magazine, 24*(4), 120.

Gentner, D., & Forbus, K. (2011). Computational models of analogy. *Wiley Interdisciplinary Reviews: Cognitive Science, 2*(3), 266-276. doi:10.1002/wcs.105

Goodman, N., Masinghka, V., Roy, D., Bonawitz, K., & Tenenbaum, J. (2008). Church: A Language for Generative Models. *Proc. Uncertainty in Artificial Intelligence (UAI).*

Hairer, E., Lubich, C., & Wanner, B. (2006). *Geometric numerical integration structure-preserving algorithms for ordinary differential equations.* New York: Springer.

Hayes, P. (1979). The naive physics manifesto. In D. Michie, *Expert Systems in the Microelectronic Age.* Edinburgh: Edinburgh University Press. From http://aitopics.org/publication/naive-physics-manifesto

Hayes, P. (1985). Naive physics 1: Ontology for liquids. In J. Hobbs, & R. Moore, *Formal Theories of the Commonsense World* (pp. 71-107). Norwood, NJ: Ablex. From http://www.issco.unige.ch/working-papers/Hayes-1978-35.pdf

Hoffman, K., & Schreiber, M. (Eds.). (2002). *Computational Statistical Physics: From Billiards to Monte Carlo.* New York: Springer.

Hurtado, P. I., Espigares, C. P., del Pozo, J. J., & Garrido, P. L. (2014). Thermodynamics of currents in nonequilibrium diffusive systems: theory and simulation. *Journal of Statistical Physics, 154*(1-2), 214-264.

Johnston, B., & Williams, M. (2007). A generic framework for approximate simulation in commonsense reasoning systems. *International Symposium on Logical Formalizations of Commonsense Reasoning.* From http://epress.lib.uts.edu.au/research/bitstream/handle/10453/2532/2007001119.pdf

Kapitaniak, M., Strzalko, J., Grabski, J., & Kapitaniak, T. (2012). The three-dimensional dynamics of the die throw. *Chaos, 22*(4). doi:doi: 10.1063/1.4746038.

Kaufman, D., Vouga, E., Tamstorf, R., & Grinspun, E. (2012). Reflections on simultaneous impact. *SIGGRAPH.* doi:http://dx.doi.org/10.1145/2185520.2185602

Kaufmann, H., & Meyer, B. (2008). SImulating educational physical experiments in augmented reality. *SIGGRAPH Asia.* doi:http://dx.doi.org/10.1145/1507713.1507717

Kaup, B., Lüdtke, J., & Maienborn, C. (2010). "The drawer is still closed": Simulating past and future actions when processing sentences that describe a state. *Brain and Language, 112*(3), 159-166. doi:http://dx.doi.org/10.1016/j.bandl.2009.08.009

Klenk, M., Bobrow, D., de Kleer, J., & Jansenn, B. (2014). Making Modelica Applicable for Formal Methods. *Proceedings of the 10th International Modelica Conference.* Lund, Sweden.

Koenig, N., & Howard, H. (2004). Design and use paradigms for GAZEBO, an open-source multi-robot simulator. *Intelligent Robots and Systems (IROS).*

Kuehne, S. (2004). *Understanding natural language description of physical phenomena.* Evanston, I.: Northwestern University.

Kuipers, B. (1986). Qualitative simulation. *Artificial Intelligence, 29*(3), 289-338. doi:http://dx.doi.org/10.1016/0004-3702(86)90073-1

Laplace, P. (1814). *A Philosophical Essay on Probabilities.*

Laue, T., Spiess, K., & Rofer, T. (2006). SimRobot–a general physical robot simulator and its application in robocup. In *RoboCup 2005: Robot Soccer World Cup IX* (pp. 173-183). Berlin: Springer.

Lehmann, J., Chan, M., & Bundy, A. (2012). A higher-order approach to ontology evolution in physics. *Journal of Data Semantics*, 1-25. From http://link.springer.com/article/10.1007%2Fs13740-012-0016-7#

Loan, P., Cagatay, B., & Rosen, J. (1997). Surgical simulation: An emerging technology for training in emergency medicine. *Presence, 6*(2), 147-159.

Lovett, A., Tomei, E., Forbus, K., & Usher, J. (2009). Solving Geometric Analogy Problems through Two-Stage Analogical Mapping. *Cognitive Science, 33*(7), 1192-1231.

Marcus, G., & Davis, E. (in preparation.). The Scope and Limits of Simulation in Cognitive Models.

Markman, K., Klein, W., & Suhr, J. (2009). *Handbook of Imagination and Mental Simulation.* New York: Psychology Press.

McCarthy, J., & Hayes, P. (1969). Some philosophical problems from the perspective of artificial inteligence. In B. Meltzer, & D. Michie, *Machine Intelligence 4* (pp. 463-502). Edinburgh: Edinburgh U. Press.

Mombauri, K., & Berns, K. (2013). *Modeling, simulation and optimization of bipedal walking.* Springer.

Murman, S., Chan, W., Aftosmis, M., & Meakin, R. (2003). An interface for specifying rigid-body motions for CFD applications. *41st AIAA Aerospace Sciences Meeting.* From http://people.nas.nasa.gov/~wchan/publications/aiaa2003-1237.pdf

Nayak, P. (1994). Causal Approximations. *Artificial Intelligence, 70*(1-2), 277-334. doi:http://dx.doi.org/10.1016/0004-3702(94)90108-2

Nyga, D., & Beetz, M. (2012). Everything robots always wanted to know about housework (but were afraid to ask). *IEEE/RSJ International Conference on Intelligent Robots and Systems.* From http://ias.cs.tum.edu/_media/spezial/bib/nyga12actioncore.pdf

Pang, T. (2006). *An Introduction to Computational Physics.* Cambridge: Cambridge University Prss.

Price, C., Pugh, D., Wilson, M., & Snooke, N. (1995). The FLAME System: Automating electrical failure mode and effects analysis (FEMA). *IEEE Reliability and Maintainability Symposium*, (pp. 90-95).

Rahimian, A., Lashuk, I., Veerapaneni, S., Chandramowlishwaran, A., Malhotra, D., Moon, L., . . . Biros, G. (2010). Petascale Direct Numerical Simulation of Blood Flow on 200K Cores and Heterogeneous Architectures. *Supercomputing*, (pp. 1-11). doi:http://dx.doi.org/10.1109/SC.2010.42

Robinson, S. (2004). *Simulation: The Practice of Model Development and Use.* Hoboken, N.J.: Wileyh.

Shelley, M., & Zhang, J. (2011). Flapping and bending bodies interacting with fluid flows. *Annual Review of Fluid Mechanics, 43*, 449-465.

Souris, J., & Delmas, D. (2007). Experimental assessment of Astree on safety-critical avionics software. In F. Saglietti, & N. Oster (Ed.), *Computer Safety, Reliability, and Security, 26th Intl. Conf. 4680*, pp. 479-490. Nuremberg, Germany: Springer, Lecture Notes in Computer Science.

Stewart, D. (2000). Rigid-body dynamics with friction and impact. *SIAM Review, 42*(1), 3-39. From http://www.jstor.org/stable/2653374

Tabiei, A., & Nilakantan, G. (undated). *Reduction of acceleration induced injuries from mine blasts under infantry vehicles.* Cincinatti: Dept. of Aerospace Engineering and Engineering Mechanics, University of Cincinatti. From http://www.ase.uc.edu/~atabiei/pdfs/J.pdf

Vogelsberger, M., Genel, S., Springel, V., Torrey, P., Sijacki, D., Xu, D., . . . Hernquist, L. (2014). Properties of galaxies reproduced by a hydrodynamic simulation. *Nature, 509*, 177-182. doi:doi:10.1038/nature13316

Vondrak, M., Sigal, L., & Jenkins, O. (2008). Physical simulation for probabilistic motion tracking. *Computer Vision and Pattern Recognition.* From http://robotics.cs.brown.edu/projects/dynamical_tracking/

Weld, D. (1990). Approximation Reformulations. *Processings of the National Conference on Artificial Intelligence (AAAI-90).* From http://aaaipress.org/Papers/AAAI/1990/AAAI90-062.pdf

Weld, D. (1992). Reasoning about Model Accuracy. *Artificial Intelligence, 56*(2), 255-300.

Weld, D. (1994). An introduction to least-commitment planning. *AI Magazine, 15*(4), 27-61. From http://www.aaai.org/ojs/index.php/aimagazine/article/view/1109

Weld, D., & de Kleer, J. (1989). *Qualitative Reasoning about Physical Systems.* San Mateo, Calif.: Morgan Kaufmann.

Willemain, T. (1995). Model Formulation: What Experts Think About and When. *Operations Research, 43*(6), 916-932.

Zicker, S., & Veloso, M. (2009). Tactics-based behavioural planning for goal-driven rigid body control. *Computer Graphics Forum*. From http://www.cs.cmu.edu/~mmv/papers/09cgf-stefan.pdf