URC97095

What Can Robots Do? Towards Theoretical Analysis

Monica Nogueira

Knowledge Representation Laboratory
Department of Computer Science
University of Texas at El Paso
El Paso, TX 79968
email monica@cs.utep.edu

Abstract

Robots have become more and more sophisticated. Every robot has its limits. If we face a task that existing robots cannot solve, then, before we start improving these robots, it is important to check whether it is, in principle, possible to design a robot for this task or not. For that, it is necessary to describe what exactly the robots can, in principle, do.

A similar problem — to describe what exactly computers can do — has been solved as early as 1936, by Turing. [n this paper, we describe a framework within which we can, hopefully, formalize and answer the question of what exactly robots can do.

1 Formulation of the problem

The question of "what can be computed" is, basically, solved. One of the fundamental problems of the traditional theory of computing is: *what can, in* principle, *be computed?* As an answer to this question, Turing developed, in 1936, the notion of a Turing machine: a one-dimensional tape with a bead that moves along it. More complicated computational devices have been proposed since then, devices that use sophisticated memory and sophisticated operations, but whatever can be computed on any of them can still be computed on a Turing machine (slower, but still computed). In this sense, Turing machine does a pretty good formal answer to the question "What can, in principle, be computed?" (for details, see, e.g., [2]).

A new question: what can be done? In the beginning, computers were mainly used for computing. Even when the ultimate goal was to change something in the world, the computer would only generate the instructions, and then these results will be used in the actual control.

Nowadays, a human being is more and more out of this loop. Computers (especially computers employed by robots) are directly linked to motors that move the robot and to the actuators that make the robot change the world.

For such computer-equipped robots, the same question appears: what are the limits of their ability? In plain English, what exactly can the robots do'?

What is known and what needs to be done. Although this question seems to be very fundamental, surprisingly, it has been formulated very recently, in [1].

The main reason why this question has never been analyzed before is that robots are mainly done by practical people, in a manner that mixes precise methods and heuristics. Only recently, when robots have become more and more complicated, and we can no longer rely on our intuitive understanding of what they do, robot designers started to appreciate the value of theoretical methods.

in particular, in [1], the question of *what can, in principle, be achieved by a robot is* formulated for a very specific (and probably, reasonably limited) formal theory of robots. As a result, the main emphasis of this paper is not so much on the analysis of what can be clone by a robot, but on checking whether the theory of robots used in this paper is general enough.

So, the question remains: if we allow most. general robots, what can these robots do?

In this paper, we describe a potential framework for answering this question.

2 Towards the desired formalism

The main objective of a robot is to change the state of the world. Therefore, we are interested in knowing which states can be changed to which. In order to describe that, we must first describe, from the robot's prospective, what are th possible states of the world.

First approximation: 2-D world. For simplicity, let us first consider a robot on a surface (e.g., on Earth). In this case, to describe the actual state oft he world, we must describe what happens at each point of this plane. If we fix a coordinate system, then we are interested in knowing. for every pair of real numbers x and y, what exactly happens at a point with coordinates (x,y).

Discreteness **in space.** In reality, all robot's sensors and actuators have only limited spatial accuracy Ax. As a result, from the viewpoint of this robot, there is no way to distinguish between, a point $(x$, y) and, e.g., a point $(x + \Delta x/2$, y – $\Delta x/2)$. In other words, a robot cannot distinguish between the events that happen within a square cell of size Δx x Ax. Hence, to describe the state of the world as it is viewed by a robot and as it can be changed by a robot, all we have to do is describe what exactly is happening within each of these cells.

As the first of such cells, we can take a cell centered in O, i e., a cell $(-\Delta x/2, \Delta x/2) \times (-\Delta x/2, \Delta x/2)$. In this case, all other cells have as centers points oft he type $(i$. Ax, $j \cdot \Delta x)$ with integer *i* and j. Thus, instead of the two *real numbers,* different cells can be characterized by two integers, and the state of the Universe can be characterized by describing, for each cell, what is happening in it.

Each cell can have only finitel y many states. How to describe what is happening in each cell? In reality, there are infinitely many possible things that can happen within each cell. However, in reality, each robot has only finitely many sensors, and each sensor has only finitely many different states. As a result, the number of different sensor reading is finite. Hence, from the viewpoint of how a robot sees the world and what this robot can do, we can distinguish between only *finitely many* different states of each cell.

Let us denote the set of all possible states of each cell by Σ.

How to describe. the state of the world? Resulting description. As a result of the above analysis, we can describe the state of the world by describing. for each pair of integers i and j, in which exactly state $\sigma \in \Sigma$ the corresponding cell is. In mathematical terms, the assignment of a state $\sigma \in \Sigma$ to every pair *(i, j)*$\in Z \times Z$ is called a *function,* Therefore, a state of the world can be described by a *function* from the set Z x Z of all pairs of integers into the set Σ of all cell states.

We are interested in doable tasks. In our analysis, we are interested in *doable* tasks, i .e., in tasks that a robot can perform during a finite amount of time. During a finite period of time, the robot can only change the state of finitely many cells; therefore, when we describe what a robot can do, we can restrict ourselves only to the contents of finitely many cel Is.

In other words, when we describe possible states of the world, it is sufficient to consider only the situations in which only finitely many cells are non-empty, and all the others are empty. [n mathematical terms, we get the following definition:

Definition 1. *Let a finite set Σ be given that* contains *the symbol Λ. Elements of this set will be called states of a cell.*

Let Z denote the set of all integers. By a *state of the world* s, we mean a function s $: Z \times Z \rightarrow \Sigma$ such *that for all but finitely many pairs of integers (i, j),* $s(i,j)$ = A. *The value $s(i,j)$ is called* a *state of the cell (i,j). If $s(i,j) =$* A, we say *that the cell (i,j) is empty.*

The state of the robot. To complete the description of the state of the world, we must also describe where the robot is, and what state the robot is in.

A robot is a finite machine; therefore, it can be in only finitely many different states. Similarly to the description of the Turing machines [2], let us denote the set of all possible states of the robot by K. This set must include the *initial state so* in which the robot starts, and the *halting state h* in which the robot reports that the task is done.

The location of a robot at any given moment of time is described by two coordinates i and j.

Definition 2. Let *a finite set K* be *given that contains two elements* s_0 and *h. Elements of the* state K *will* be *called states of the robot;* SO will called the *initial state, and* **h** *will* **be** *called* **the** *final state.*

By a configuration, we mean a tuple (,9, k, i, j), *where s is a* state *of the world, k is* a state *of the robot, and i and j* are *integers.*

The robot is the only active agent in the world. For simplicity, we assume that the robot is the only active agent in the world.

This assumption immediately excludes situations like a fire in the plant where, in addition to the robot moving things and trying to extinguish the fire, there is another powerful agent — the fire itself — which drastically changes the states of different cells (by burning their contents).

However, in space applications, this assumption is quite reasonable: a rover moving on the surface of the lifeless planet (e.g., on the Moon or on Mars) is, basically, the only active agent there.

Within this assumption, the only changes in the world are the changes that this robot causes.

How can a robot change the state of a cell? In the Turing machine, the changes were easy to describe: the states of each world cell were simply symbols written on a tape. We could always write or delete a symbol.

For a robot, the situation is not so easy: a state of a cell can mean, e.g., that there is a block in this cell, or two blocks stacked on top of each others. There is no way that a robot can simply change the state of a cell from a block in it to "empty": this would mean that a robot can somehow "eliminate" the block. It is even more unprobable that a robot will be able to "create" a new block out of nothing. All it can do is either *transform* the contents of the cell, or move it.

Let us describe the situation accordingly.

States that are simply marks and states that describe physical contents of each cell. First of all, in our description of the state of the cell, we must distinguish between the *marks* that can be easily rewritten, overwritten, etc., and the states that describe the *physical contents* of the cell.

Definition 3. *Let* a *subset* $M \subseteq \Sigma$, *be given* that *contains* **A.** *Elements of the set* M *will be* called *marks, and elements of the remaining set* $P = \Sigma - M$ *will be called physical states.*

How a robot changes things. A robot can easily make a mark, but it cannot convert one block into two. To take these restrictions into consideration, we can describe which cell states can be transformed into which. If we can transform a state σ in to a state u', and if we can also transform a state u' into a state σ'', this means that we will be able, in two steps, to transform the state σ into the state u". Therefore, it is reasonable to require that this "transformability" relation is transitive.

We are mainly interested in non-destructive transformations, so it is also reasonable to assume that if σ can be transformed into σ', then we can always "undo" this transformation and transform σ' back into U.

Thus, the "transformability" relation must be an equivalence relation.

Definition 4. *Let* $\equiv$ *be an equivalence relation on the set P.*

We will *extend this* relation *to the entire set* Σ *by assuming that* $\sigma \equiv \sigma'$ *for every two elements* U, U' $\in$ *M.* *If* $\sigma \equiv \sigma'$, *we will* say *that a state* σ *can be transformed into a state u'.*

A robot can move things. To be able to describe how a robot moves things around, we must divide its states into states in which a robot is not carrying anything, and states in which it "piggybacks" some things from the cell.

Ideally, we would like the robot to be able to carry each state intact, but in reality, it may be necessary first to rearrange the things, i.e., to apply one of the possible transformations. Thus, within the set *P* of all "physical" states of a cell, we must select a subset C consisting of all "carryable" states.

In this case, in order to describe the state of the robot, we must describe two things: in what exactly the state the robot itself is, and what exactly it is carrying. In other words, the set K of all robot states is equal to the set of all pairs (proper state, carry-on).

Definition 5. *Let* C *be a subset of the set P with the property that every element* $\sigma \in P$ *is equivalent to one of the elements from the state C.* Elements *of the set* C will *be called carryable states.*

We assume that the state K is a Cartesian product K = $K_0 \times C \cup \{A\}$); if a robot is *in a state* $k = (q, \sigma)$, *we* will say that *it is in a proper state q and carrying an object u.*

When can one state of the world be transformed into another state.

Definition 6. *Let s* and s' *be two* states of *the* world. *We* say that s *can be (in principle) transformed into s'* if *in* both states, *the* total number of physical cells (i.e., cells (i, j) for *which* $s(i,j) \in P$) *is the same, and these cells* can be placed *in one-to-one* correspondence in *such* a way that *each cell state* $s(i,j)$ *corresponds* to an *equivalent cell state.*

What can a robot do. When a robot is in a ceratin state, it can, first, stay or move. Second, it can change the state of the cell on which it is currently looking. Third, if the cell is empty and the robot is carrying something, it can download its carry-on onto the cell (maybe, rearranging it along the way). Vice versa, if the cell is empty, then the robot can load its contents on itself (maybe, rearranging it so that it will fit on the robot). As a result, we get teh following definition:

Definition 7. Let M be a set consisting *of five elements* L, R, F, B, *and* S *that will be called, correspondingly,* "*move* left", "move *right*", "*move forward*", "*move* back", **and** "slay". *By* a *transition junction, we mean a mapping*

$$\delta : K_0 \times C \times \Sigma \to K_0 \times C \times \Sigma \times M$$

that *satisfies the following property: If* $\delta(q, c, \sigma) = (q', c', \sigma', m)$, *then:*

- *either* $c \equiv c'$ *and* $\sigma \equiv \sigma'$;
- *or* $c = \Lambda$, $\sigma' \in M$, *and* $c' \equiv \sigma$;
- *or* $c \neq \Lambda$, $c' = \Lambda$, *and* $c \equiv \sigma'$.

Main result. The step-by-step transformation oft he robot is defined accordingly (similar to [2]). Now, we can formulate the main result:

THEOREM.

- Let R *be a robot (described by* Definitions *1–7).* For every state s, *let* **us** denote by F(s) **the** state into *which* this *robot transforms s.* Then, *F is* a *computable mapping from the* set S of the states **of** *the world* into *itself such that for every state s, this state s can be,* **in principle,** *transformed* **into** *F(s).*
- *Let F : S → S be a computable mapping such that for every state s, this* **state s can be,** *in principle,* **transformed into F'(s).** *Then, there exists a robot* **that** *for every* starting *state s, transforms it into F(s).*

Idea of the proof. The technical description of this proof will take too much space, so we will restrict ourselves to describing the main idea of this proof.

Since in every state, only finitely many cells are used, we reserve two potentially infinite areas: one area will serve as a Turing machine tape on which the robot will simulate the computations necessary to decide what to move where, and another wi II serve as a storage area, where objects will be stored one by one, with a possibility to easily access each of them.

Starting with the borderline elements of the configuration s, we take the contents of each cell one by one and carry them to the storage area. After all these elements arc in storage, we carry them back one by one, filling the elements of the new configuration row by row and column by column so that at each moment of time, we will have easy access. (This is somewhat similar to assembling a ready-made home.).

For space applications, we must consider a 3-D problem. For space applications, especially for applications to the Space Station, we need robots that operate in a 3-D world.

All our definitions and results can be easily modified for this case: the main modification is that now we need *three* coordinates (i, j, k) (and not two, as before) to identify a cell.

Acknowledgments. This work was part ly supported by the NASA Pan American Center for Environmental and Earth Studies (PACES). The author is thankful to Chitta Baral, Dan Cooke, Michael Gelfond, Vladik Kreinovich, and Scott Starks for their help and encouragement.

References

[1] H. J. Levesgue, "What is planning in the presence of sensing?", *Proceedings of A AA I'96, Annual Conference of the American Associations on Artifical Intelligence, 1996.*

[2] C. H. Papadimitriou, *Computational Complexity*, Addison Wesley, San Diego, 1994.