December 1991

Simulating Human Tasks Using Simple Natural Language Instructions

Moon Ryul Jung
University of Pennsylvania

Jugal K. Kalita
University of Colorado

Norman I. Badler
University of Pennsylvania, badler@seas.upenn.edu

Wallace S. Ching
University of Pennsylvania

Recommended Citation

Jung, M., Kalita, J. K., Badler, N. I., & Ching, W. S. (1991). Simulating Human Tasks Using Simple Natural Language Instructions. Retrieved from http://repository.upenn.edu/hms/65

Copyright 1991 IEEE. Reprinted from *Proceedings of the Winter Simulation Conference*, December 1991, pages 1049-1057.
Publisher URL: http://dx.doi.org/10.1109/WSC.1991.185723

Simulating Human Tasks Using Simple Natural Language Instructions

Abstract

We report a simple natural language interface to a human task simulation system that graphically displays the performance of goal-directed tasks by an agent in a workspace. The inputs to the system are simple natural language commands requiring achievement of spatial relationships among objects in the workspace. To animate the behaviors denoted by instructions, a semantics of action verbs and locative expressions is devised in terms of physically based components, in particular geometric or spatial relations among the relevant objects. To generate human body motions to achieve such geometric goals, motion strategies and a planner that used them are devised. The basic idea for the motion strategies is to use commonsensical geometric relationships to determine appropriate body motions. Motion strategies for a given goal specify possibly overlapping subgoals of the relevant body parts in such a way achieving the subgoals makes the goals achieved without collision with objects in the workspace. A motion plan generated using the motion strategies is basically a chart of temporally overlapping goal conditions of the relevant body parts. This motion plan is animated by sending it to a motion human controller, which incrementally finds joint angles of the agent's body that satisfy the goal conditions in the motion plan, and display the body's configurations determined by the joint angles.

Comments

Copyright 1991 IEEE. Reprinted from *Proceedings of the Winter Simulation Conference*, December 1991, pages 1049-1057.
Publisher URL: http://dx.doi.org/10.1109/WSC.1991.185723

Proceedings of the 1991 Winter Simulation Conference
Barry L. Nelson, W. David Kelton, Gordon M. Clark (eds.)

SIMULATING HUMAN TASKS USING SIMPLE NATURAL LANGUAGE INSTRUCTIONS

Moon Ryul Jung*
Jugal K. Kalita+
Norman I. Badler*
Wallace Ching*

* Department of Computer & Information Science,
University of Pennsylvania,
Philadelphia, PA 19104-6389.

\+ Department of Computer Science,
University of Colorado,
Colorado Springs, CO 80933.

ABSTRACT

We report a simple natural language interface to a human task simulation system that graphically displays performance of goal-directed tasks by an agent in a workspace. The inputs to the system are simple natural language commands requiring achievement of spatial relationships among objects in the workspace. To animate the behaviors denoted by instructions, a semantics of action verbs and locative expressions is devised in terms of physically-based components, in particular geometric or spatial relations among the relevant objects. To generate human body motions to achieve such geometric goals, motion strategies and a planner that uses them are devised. The basic idea for the motion strategies is to use commonsensical geometric relationships to determine appropriate body motions. Motion strategies for a given goal specify possibly overlapping subgoals of the relevant body parts in such a way that achieving the subgoals makes the goal achieved without collision with objects in the workspace. A motion plan generated using the motion strategies is basically a chart of temporally overlapping goal conditions of the relevant body parts. This motion plan is animated by sending it to a human motion controller, which incrementally finds joint angles of the agent's body that satisfy the goal conditions in the motion plan, and displays the body configurations determined by the joint angles.

1 INTRODUCTION

Computer animation of human tasks in a given workspace is an effective tool for visualizing the results of task simulation. To facilitate the use of task animation for this purpose, the workspace designers should be allowed to specify tasks using a high level language which does not require extensive training or learning. Natural language is most suitable among such languages, because it is especially geared to describing and expressing human behavior. Languages have evolved rich verbal, prepositional and adverbial vocabularies to facilitate the expression of subtle as well as complex movements. Accordingly, we have been conducting research towards achieving realistic animations of tasks specified in terms of a sequence of natural language instructions (Badler et al 1990), such as *Put the block on the table*, *Close the door*, *Put the block in the box* and *Push the block against the chair*. Typically natural language instructions specify *what* is to be done, and rarely indicate the specific articulations required of body segments or parts to achieve a given goal. The agent should plan his motions so that the goal can be effectively achieved. Typical motions include taking a step, re-orienting the body, kneeling down, and reaching around an obstacle. The planned motions are dependent upon the structure of the workspace and the situations in which the agent is positioned. To support the use of natural language instructions as a human task simulation language, we should be able to represent goals conveyed by instructions in terms of fine-grained physical components, in particular geometric or spatial relations among the relevant objects. We also need the capability of human motion planning by which body motions needed to achieve the fine-grained physical relations are found. This paper addresses how to provide these two capabilities. More specifically, given a task level goal in a given workspace specified in simple natural language commands such as *Put the block on the table*, the problem of this study is to *automatically* find a set of possibly overlapping primitive motions that the relevant body segments of the agent must perform to achieve the task.

2 HANDLING NATURAL LANGUAGE INPUTS

To handle natural language instructions, we need to develop an approach to representing the semantics of action verbs. Action verbs are used to denote actions that an agent is instructed to perform. Instructions also contain prepositional and adverbial modifiers to provide values of arguments—obligatory as well as optional, as well as other relevant information to enable the performance of the underlying task.

Elsewhere, we have discussed in detail the scheme adopted for the representation of meanings of action verbs and their modifiers (Badler 1990, Kalita 1990a, Kalita 1990b, Kalita 1991). We specify an implementable semantics of action verbs and their modifiers in terms of physically-based components. These include geometric or spatial relations among relevant objects, relationships among sub-actions such as sequentiality, concurrency and repetitiveness, inherent specification of an action's termination condition or lack thereof, and inherent motion or force-related components of word meanings. Specification of geometric relations among objects is crucial in describing a physical environment as well as describing how an action changes an environment. Thus, specification of geometric relations must play an important role in the lexical semantics of action verbs which denote actions performed by one or more agents. Geometric relations provide information regarding how one or more objects or their sub-parts relate to one another in terms of physical contact, location, distance among them, and orientation. It will suffice to classify geometric relations into two classes—*positional* and *orientational*. Positional geometric relations specify a situation in which a geometric entity (a point, line, surface or a volume) is related to (or constrained to) another geometric entity. For example, the verb *put* which requires two obligatory objects as arguments specifies that a certain geometric relation be established between the two objects. The particulars of the geometric relation are provided by prepositional phrases as seen in the two sentences: *Put the ball on the table* and *Put the ball in the box*. The preposition *on* specifies that an arbitrary point on the ball be related to or constrained to an arbitrary point on the surface of the table. Similarly, *in* in the second sentence refers to the fact that the ball or the volume occupied by the ball be constrained to the interior volume of the box. There are two geometric entities involved in the case of both *on* and *in*. In the case of the *in* example, the two entities are both volumes, viz., the volume of the ball and the interior volume of the box. In the case of the *on* example, there are two points—a point on the ball and a point on the table. In general, a geometric relationship relates two geometric entities each of which may be 0,1,2, or 3-dimensional. We call them the *source-space* and *destination-space*, or simply *source* and *destination*.

Although not discussed in this paper, our representation also allows for specifying motion and force-related components of a word's meaning, and can be used to handle verbs such as *move, push, pull, roll, put, fill* and *turn*. Similar representations have also been developed for prepositions such as *against, along, around, from* and *to*, as well as adverbs such as *gently, quickly* and *exactly*.

Kalita (1990b) demonstrated the validity and usefulness of the representation scheme adopted, by performing animation of the underlying tasks starting from natural language instructions. An earlier attempt towards the establishment of a link between natural language instructions and the graphical animation of underlying tasks was also reported in (Esakov 1989, 90). In the current paper, we will discuss how an example command *put the block on the table* is interpreted and planned. We assume that in our environment, the block is initially sitting at the bottom of an open box and that there is a table by the side of the box. The agent is standing in a position where she can reach the block as well as the table without walking.

3 REDUCING THE TASK LEVEL GOALS

Let us consider the command *Put the block on the table*. This command will be used as a running example of planning and animation. The relevant lexical entries for the verb *put* and the preposition *on* are given below. The lexical entries for the determiner and the nouns are not shown here. In these definitions we have removed some of the details to make them more presentable. *Put the block on the table* is paraphrased as saying *Put the block so that the block is on the table*, and thereby parsed into *put(hearer, (the block), on((the block), (the table)))*. Our lexical definition of the verb *put* is given as

put(agent, object, spatial-relation) ⟶
achieve(agent, positional-goal(spatial-relation))

According to this definition, *put(ag0, block1, on(block1, table1))* is translated into *achieve(ag0, positional-goal(on(block1, table1)))*. It means that an agent *agent* is to achieve a particular positional goal between *block1* and *table1* denoted by function term *positional-goal(on(block1, table1))*. That particular positional goal is dependent on the nature of the spatial relationship *on* and the natures of the two objects *block1* and *table1*.

Here we present only one sense of the preposition *on*. This sense is associated with the most common usage of *on* and is expressed in terms of three components: *above, contact* and *support*. Using these components, the positional goal associated with spatial relationship *on(object1, object2)* is specified as follows:

```
positional-goal(on(object1, object2)) ⟶
  above(source-space, destination-space) and
  contact(source-space, destination-space) and
  support(destination-space, source-space), where
    source-space :=
          any-of(self-supporting-space-of(object1));
    destination-space:= an area X in
      any-of((supporter-space-of(object2))
      such that horizontal (X) and
            direction-of(normal-to(X))="global-up"
            and free(X);
```

Here, ':=' represents assignment of a value to a placeholder. The source space and the destination space are determined by the geometric relationship intended and the nature of the objects involved in the relationship. The source space is a *self-supporting-space* of *object1*. A *self-supporting-space* of an object is a point, line or surface on the object on which it can be supported. This is a fundamental functional property of an object. Some examples are that a ball can be supported on any point on its surface, a cubic block on any of its faces, and a table on its four legs.

The destination space is an area on the *supporter-space* of *object2*. A *supporter-space* of an object is a feature of the object that can support other things against the force of gravity. The knowledge base has information as to which face of a block is a *self-supporting-space* and which face is a *supporter-space*. For example, a table's function is to support objects on its top surface, and a bookcase supports objects on the top surface of the individual shelves. It is also required that (1) the destination space is horizontal, (2) the normal vector of the destination space, the outward going vector perpendicular to the destination space, is *up* with respect to the global reference frame, and (3) the destination space is a free area spacious enough to accommodate the source space.

Using the definition for *positional-goal(on(object1, object2))*, and the knowledge base entries for the block *block1* and *table1*, the goal *achieve(ag0, positional-goal(on(block1,table1)))* is reduced into a more refined form:

```
happen(achieve(ag0,
               holdat(above(Src,Dest) and
                      contact(Src,Dest) and
                      support(Dest,Src), T2)),
       E, [T1,T2]).
```

Here, *Src* is the source space of object *block1* and *Dest* is the destination space of object *table1*. Formulas of the form *happen(achieve(Agent, holdat(Rel,T2)), E, [T1,T2])* means that an action *E*, in which agent *Agent* achieves goal *Rel* to hold at time *T2*, happens between time *T1* and *T2*. The component subgoals *above*, *contact*, and *support* can be geometrically defined as follows.

```
above(Src,Dest) ⇐
  positioned-at(center-of(Src), Pos) such that
  minimal-vertical-distance-between(Src,Dest) > 0.
```

That is, *Src* is above *Dest* if the center of *Src* is positioned at some location *Pos* so that the minimal vertical distance between *Src* and *Dest* is greater than zero.

```
contact(Src,Dest) ⇐
  positioned-within(center-of(Src), Dest) and
  align-direction(normal-to(Src), normal-to(Dest)).
```

That is, *Src* contacts *Dest* if the center of *Src* is positioned within *Dest*, and *Src* and *Dest* are parallel by making their normal vectors aligned with each other.

```
support(Dest,Src) ⇐
 intersect(vertical-line-through(center-of-mass(Src)),
         Dest).
```

That is, *Dest* supports *Src* if the vertical line passing through the center of mass of *Src* intersects *Dest*.

4 GEOMETRY OF THE HUMAN FIGURE

Our human figure model is anthropometrically realistic. Figure 1 shows a geometry of the human figure model. The human figure is modelled as a tree of rigid segments connected to each other via joints. For example, the left lower arm segment and the left upper arm segment are connected to each other via the elbow joint. All the segments in the human figure and in the workspace are assumed to be polyhedra. The human figure model has 36 joints with a total of 88 degrees of freedom in it, excluding the hands and fingers. The human figure has an upper body consisting of 17 segments and 18 vertebral joints (Monheit & Badler 1991). Each joint is described by defining a local reference frame (coordinate system) on it. The degrees of freedom of a joint is the number of independent parameters that uniquely determine the orientation of the joint with respect to the joint reference frame. In the case of the human figure, the parameters are joint angles measured with respect to each joint's reference frame. For example, the elbow is modelled to have one degree of freedom, and the wrist three degrees of freedom. The wrist joint with the three degrees of freedom is considered to have three joint angles that determine its orientation. The spatial configuration of a figure with N degrees of freedom at a time is uniquely determined by a sequence of N joint angles in the figure at that time.

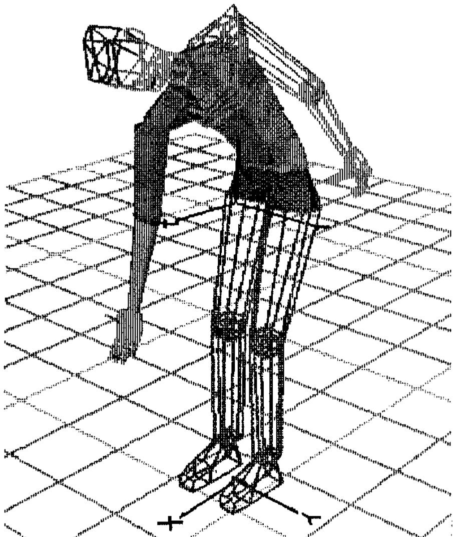

Fig. 1: Geometry of Human Body

The segments of the body whose end points are free are called *end effectors.* At the level of gross motion planning, hands and feet are typically considered end effectors. To specify a desired spatial configuration of the end effectors or other parts of the body, we define local reference frames on the body parts and state goal conditions in terms of the origin positions and orientations of these reference frames. Such reference frames will be called *reference sites*, and are considered handles on the body. The movement of a given body part is dependent on the base or starting joint with respect to which the end effector is moved. Such base joint is called the *pivot joint* and the segments from the base joint to the moving body part is called the *moving body chain.* In figure 1, the shaded part is the moving body chain determined relative to the pivot joint waist. We use two pivot joints: a *shoulder* and a *waist.* For example, if the waist is the pivot joint, the segments in the chain are the palm, the lower and upper arms, and the vertebrae forming the torso. The joints in the chain are the wrist, the elbow joint, the shoulder joint, the joints among the vertebrae, and the waist joint. This means that the palm and the arms can be moved in any direction and orientation within their physical limits. The torso can also be moved and bent at the waist.

Given a sequence of joint angles of the moving body chain, the spatial configuration of the moving body chain is uniquely determined. This mapping from the joint angles to the spatial configuration is called forward kinematics and usually computationally straightforward. The opposite process of finding the joint angles that determine a given spatial configuration is called inverse kinematics, and is difficult computationally.

5 POSTURE PLANNING

In planning human body motions for a given fine-grained geometric goal, we are concerned with gross motions rather than fine motions. The process of finding the appropriate movements of relevant body parts or segments for gross motions is called *posture planning.* The planned sequence of primitive motions are sent to a process animator, which finds and graphically displays the body configurations over time for the planned motions.

Most AI symbolic planning works treat body motions such as *put on* as primitive, and do not consider how anthropometrically realistic human agents will achieve such primitive tasks. However, we need such capability in order to use animation of the human figure for evaluation of workspaces for human operators. The techniques of robot motion planning (Lozano-Perez 1983, 1985, 1987, Donald 1987, Barraquad, Langlois & Latombe 1989) could be used for planning motions of realistic human agents. The common framework of most robot motion planning techniques is that all information about object boundaries in the environment and about movement of the manipulator is represented in terms of joint angles of the manipulator. Note that a manipulator (robot arm) at each moment can be represented by its joint angles at that moment, because a sequence of joint angles determines the spatial configuration of the manipulator. The motions of the manipulator is found by searching through the joint angle space from the initial joint angles to the final joint angles, so that a sequence of intermediate joint angles of the manipulator satisfies given constraints or motion criteria. We can use this framework, if a motion criterion under consideration can be directly characterized in terms of joint angles. Trying to characterize motion criteria in terms of joint angles of the manipulator is natural because the purpose of the motion planning is to determine spatial configurations of the manipulator over time, and in turn joint angles of the manipulator over time. However, for human agents whose behaviors are much more complicated than robots, it is not easy to characterize all the motion criteria directly in terms of joint angles of the human body. For the current robot manipulators, motion criteria can be characterized in terms of joint angles, because their behaviors are rather simple. Usually, the base of a robot manipulator is assumed to be fixed at a given location and the distance between the initial configuration and the final configuration of the manipulator is assumed to be relatively short.

In the case of human motions, various kinds of reasoning are needed before characterizing motions directly in terms of joint angles. Hence the basic idea of our approach is that before transforming the human motion problem into purely numerical problem formulated with respect to the joint angle space, we need to apply symbolic reasoning based on our commonsensical geometric notions. Symbolic reasoning is used to deal with motion criteria that are difficult to specify in terms of joint angles. More specifically, the Euclidean space positions and orientations of reference sites of the human figure are manipulated to satisfy motion criteria. The joint angles causing such positions and orientations are computed only when intermediate spatial configurations are suggested, in order to check whether the suggested spatial configurations are feasible in terms of joint angles. The joint angles for a given spatial configuration is computed by an inverse kinematic algorithm.

To plan for a given fine-grained geometric goal, the planner obtains the goal position of the end effector that would satisfy a given geometric goal, and then generates intermediate goal conditions of the relevant body parts. These goal conditions are generated based on the inherent motion constraints such as avoiding obstacles and respecting the limit of the agent's muscular strength while lifting or moving objects. More specifically, given a goal, goal-directed motion strategies are employed to generate the goal conditions about the positions and orientations of the reference sites on the relevant body parts, so that a given set of motion constraints are respected. These goal conditions comprise a chart of temporally overlapping goal conditions about the relevant body parts. Then, the joint angles that satisfy the overlapping sequence of the goal conditions are computed using a robust inverse kinematic algorithm (Zhao 1989, Phillips, Zhao & Badler 1990, Phillips & Badler 1991). Zhao's algorithm uses a numerical optimization technique.

In general, motion criteria (motion strategies or rules) to be used to plan human motions can vary depending on the purpose of task animation, which determines how fine-grained the simulated motions should be. The motion strategies are specified using information about the structures of the workspace and the agent, and the relationships between the agent and the world. As an example of motion strategies, we describe relatively fine-grained motion strategies by which the motions of the agent are planned for a given fine-grained geometric goal. The basic idea of the motion strategies is to try straightforward planning initially, and then, if the initial planning is not feasible, to find the causes of the failure and generate goal conditions to avoid them. The process of trial and error, however, would not be seen during the execution time. The motions are planned by exploiting the structure of the human body. It is assumed that human agents moves with their feet on the ground or some horizontal base. The human body motion planning is decoupled into *upper body planning* and *lower body planning*. The interaction between the two is controlled by the *overall motion planning*. Lower body planning is basically planning movements of the waist position, which includes stepping, walking, and raising or lowering the body. The waist may be lowered, raised, or moved horizontally depending on the goal position of the end effector. Upper body planning is moving the end effector to a given goal position. If moving the end effector to the given goal position is not possible with respect to the current body posture, the lower body planning is invoked to change the body posture. In this paper, we are concerned with the upper body planning.

6 UPPER BODY PLANNING

The upper body planning is attempted with respect to a given pivot joint suggested by the overall motion planning. If the end effector cannot directly reach the final goal with respect to the given pivot joint, the planner tries to find an intermediate goal position of the end effector, from which the final goal is expected to be reached. An intermediate goal position is selected so that it is *within the reach* of the end effector with respect to the given pivot joint, *collision-free*, and *goal-directed*. If such a selection cannot be made, backtracking to the overall motion planning occurs.

Heuristic rules used to find a *collision-free* location are defined differently depending on whether the straightforward linear movement of the end effector towards the final goal position is *horizontal* and *forwards*, *horizontal* and *to the left*, *horizontal* and *to the right*, or other combinations. Here the directions are determined using the coordinate system shown in figure 2. As an example of heuristic rules, consider the case where the linear movement of the end effector towards the goal position is *horizontal* and *forwards* and that movement causes collision with an obstacle. This movement is shown in figure 3 by the dotted line. In this case, an intermediate goal position of the end effector is *collision-free* if the end effector at that intermediate goal position is *exactly above*, *exactly below*, *exactly to the right of*, *exactly to the left of*, *above* and *to the right of*, *above* and *to the left of*, *below* and *to the right of*, or *below* and *to the left of* the obstacle, by a given margin. The other cases are treated similarly. The heuristics for collision-avoidance of the inner segments is the same as the heuristics for the end effector.

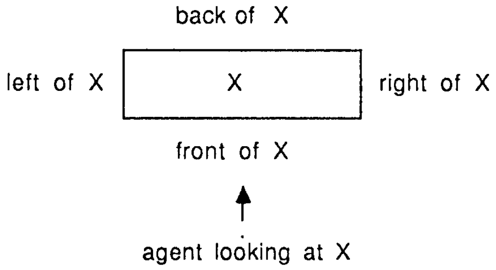

Fig. 2: Coordinate System.

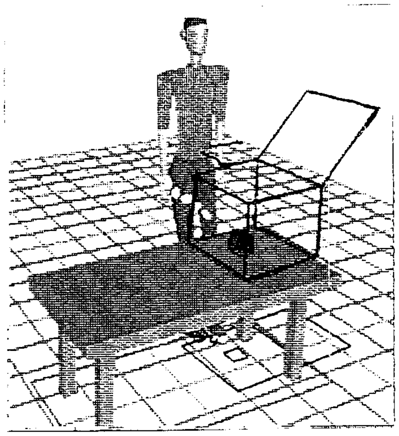

Fig. 3: The Agent Moves The Hand Linearly Towards The Block Following The Dotted Line

An intermediate goal position is *goal-directed* if it really helps the end effector to reach the final goal position. This goal condition is defined differently depending on whether the final goal position is *within* some container or not. In a workplace, every object is in some container object; the workspace itself is considered a container.

If an object is within a container, it is assumed that there is a opening through which objects in the container can be reached. The opening may be covered or not. When it is covered, it is assumed that it can be opened. In the case where the final goal position of the end effector and the current position of the end effector are immediately within the same container (including the workspace), an intermediate goal position is *goal-directed* if it is *minimally away from* the final goal position. An intermediate goal position is *minimally away from* the final goal position if it is chosen so that the length of the straight line from the intermediate goal position to the final goal position is minimized, while achieving other goal conditions and respecting the current constraints. In the case where the final goal position of the end effector is within some container and the current position of the end effector is outside of that container, a *goal-directed* intermediate goal position should be *within* that container, or *in front of* the *opening* of the container, or minimally away from the *opening* of the container.

When goal conditions are generated as a result of the upper body planning, some of them are to be achieved at the same time. For example, in the case where the end effector's linear movement towards the final goal position *FinalGoalPos* is forwards and horizontal, and the end effector collides with an object *Obstacle* while moving, goal conditions about an intermediate goal position *Pos* of the end effector may contain goal conditions such as (a) *EndEffector* at *Pos* is above *Obstacle* by a given margin *D*, (b) *Pos* is within the reach of *EndEffector*, and (c) *Pos* is minimally away from *Opening* of the container. All these goal conditions about *Pos* should be achieved at the same time.

7 TRANSLATING GOAL CONDITIONS TO OBJECTIVE FUNCTIONS

To achieve generated goal conditions, they are first translated to objective functions, which are then to be minimized by a numerical optimization technique (Zhao 1989). For example, consider a goal about an end effector *EndEff* such that its reference site *ref* is at (*Pos*, *Ori*), that is, a goal such that the position of *ref* is equal to a position *Pos* and the orientation of *ref* is aligned with a coordinate frame *Ori*. That goal is translated to two objective functions $MIN_{ref_{pos}}(ref_{pos} - Pos)^2$ and $MIN_{ref_{ori}}(ref_{ori} - Ori)^2$, where ref_{pos} refers to the position at which the reference frame ref is positioned and ref_{ori} refers to the axes of the reference frame ref. A goal about an end effector *EndEff* such that it is above an obstacle *Obstacle* by a given margin *D* is translated to an objective function that tries to make the minimal distance between the end effector and the obstacle equal to the given margin *D*. That function is represented by $MIN_{(ref_{pos},ref_{ori})}[minimaldistance(X,Y) - D]^2$, where X is $EndEff_{(ref_{pos},ref_{ori})}$, and Y is *Obstacle*. $EndEff_{(ref_{pos},ref_{ori})}$ is the end effector *EndEff* whose reference site *ref* is located at (ref_{pos}, ref_{ori}). As another example, the objective function F corresponding to collision-avoidance goal condition about a segment *Seg* colliding with an obstacle *Obstacle* from above is a function that tries to make the minimal distance between the segment and the obstacle equal to a given margin if that distance is less than the margin, and does nothing otherwise. That function is represented by

$$F = \begin{cases} MIN_{(ref_{pos},ref_{ori})}[minimaldistance(X,Y) - D]^2 \\ \quad \text{if } minimaldistance(X,Y) < D, \text{ where} \\ \qquad X = Seg_{(ref_{pos},ref_{ori})}, \\ \qquad Y = Obstacle. \\ \text{zero, otherwise.} \end{cases}$$

Here, $Seg_{(ref_{pos},ref_{ori})}$ is the segment *Seg* whose reference frame *ref* is located at (ref_{pos}, ref_{ori}).

8 PLANNING AND ANIMATION

To specify motion strategies that require overlapping temporal relationships among goal conditions about and actions of body parts, a time-based action formalism is devised. The action formalism is Kowalski's Event Calculus (Kowalski 1986), extended to allow actions with durations. A motion plan is defined to be a set of actions together with the temporal ordering goals on the actions. Motion plans for goals which often overlap are found using a planner that uses a goal-driven backwards-chaining reasoning based on given motion strategies and checks the physical feasibility of the current plan using an inverse kinematic algorithm.

Here we show an example of planning. Consider the workspace of figure 3, where block *block1* is inside box *box1*. Consider a goal *achieve(ag0, holdat(contact(block1.side6, (X in table1.top)),T))*. It says that agent *ag0* should achieve a goal so that it holds at time T. That goal is a relationship in which the side *side6* of the block *block1* contacts an area X in the top of the table *table1*. It is assumed that *block1.side6* is the bottom of the block *block1* in the current situation. Relative positions such as *top* and *front* are determined with respect to the agent's view direction. The above goal is reduced to a subgoal *achieve-with(ag0, Pivot, ag0.right-hand, holdat(contact(ag0.right-hand, block1.side1),T)*, which means that agent *ag0* achieves *holdat(contact(ag0.right-hand, block1.side1),T)* using his right hand with respect to pivot joint *Pivot*. It is assumed that *block1.side1* is the top of the block *block1* in the current situation. It is assumed that the agent *ag0* decides to use the right hand as the end effector. It is also assumed that the box is open, that the opening of the box is the top side of the box, and that the agent will use the waist joint as the pivot joint *Pivot*.

Here we show what goal conditions should be generated by the planner to find an intermediate goal position *Pos* of the end effector *ag0.right-hand*. In the current situation, the linear movement of the end effector toward the destination space *block1.side1* would cause the end effector to collide with the front side of the box *box1* (fig. 3). Among the collision-free intermediate goal positions, a position *TopPos1* satisfying *above(TopPos1, box1.opening)* will be chosen because it is goal-directed, being ***in front of*** the opening of *box1*.

Suppose the end effector is moved to the chosen intermediate goal position (fig. 4). Consider what would happen when the end effector is moved from the intermediate goal position in figure 4 to the final goal position. Suppose that the (vertical and downwards) linear movement of the end effector causes the lower arm to collide with the top edge of the front side of the box.

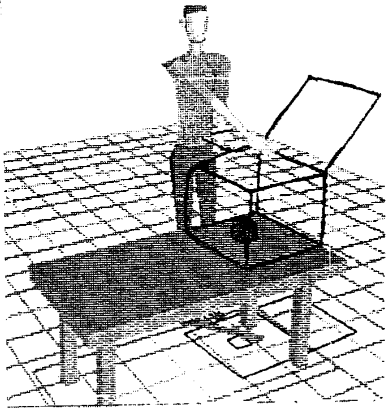

Fig. 4: The Hand Is Above The Opening Of The Box

To prevent the collision of the lower arm, the planner generates a goal condition about the elbow position. Candidate intermediate goal positions of the elbow include ***to the left of***, ***to the right of***, ***in front of*** and ***behind*** the edge of the collision (the top edge of the front side of the box). If the left, the right, or the front of the edge of the collision is chosen as intermediate goal positions of the elbow, it will fail. When the elbow is constrained to satisfy such intermediate goal positions, the end effector will be placed in such a way that the final goal position is outside of the reach of the end effector. Thus, the intermediate goal position of the elbow is constrained to be *behind* the edge of the collision (fig. 5), with respect to the view direction of the agent. If that goal condition in conjunction with the other goal conditions enables the end effector moved onto the block inside the box without collision, the planning for the goal of reaching the block is done, which is the case in this example.

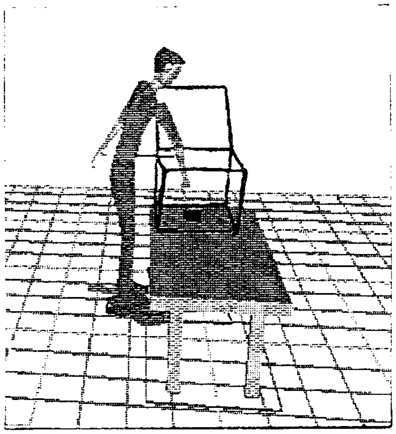

Fig. 5: The Hand Is In The Box With The Elbow Away From The Top Edge Of The Front Side Of The Box

9 CONCLUSION

The present study is a step towards achieving the goal of specifying human task animation using natural language instructions (Badler et al 1990). To link a given instruction with the corresponding action in the workspace, a physically-based semantics of motion verbs and prepositional phrases is presented, in which the task level goal denoted by the instruction is reduced into relevant fine-grained geometric relations among the relevant objects. We also show how to plan gross human body motions needed to achieve the fine-grained geometric goals. To generate the relevant goal conditions about body parts needed to achieve the goal obtained from the input instruction, we have identified the important goal conditions of the end effector, those of *collision-avoidance* and *goal-directedness*, and have provided commonsensical motion strategies to achieve these goal conditions. The commonsensical motion strategies are devised with respect to the ordinary Euclidean space. Human motion planning based on the heuristic motion strategies is not complete, in that it may not find a plan even when there is one. However, for the purpose of simulating human behaviors in workspaces, our approach has an advantage over the joint angle space reasoning used by robot motion planners, which permits a complete planner. Our approach allows the planner to use a wider range of motion strategies including commonsensical motion strategies, and that is especially needed to animate behaviors described by natural language commands.

ACKNOWLEDGMENTS

This research is partially supported by Lockheed Engineering and Management Services (NASA Johnson Space Center), NASA Ames Grant NAG-2-426, NASA Goddard through University of Iowa UICR, FMC Corporation, Siemens Research, NSF CISE Grant CDA88-22719, Air Force HRL/LR ILIR-40-02 and F33615-88-C-0004 (SEI), and ARO Grant DAAL03-89-C-0031 including participation by the U.S. Army Human Engineering Laboratory, Natick Laboratory, and TACOM.

We would like to thank members of Animation and Natural Language Group at the University of Pennsylvania for their insightful discussions. We would also like to thank Cary Phillips and Jianmin Zhao, who freely shared their time explaining the technical details of the human motion animation tools. We thank Welton Becket and Suneeta Ramaswami for reading the draft of the paper.

REFERENCES

Badler, N., B. Webber, J. Kalita, J. Esakov. Animation from Instructions. In *Making Them Move: Mechanics, control, and animation of articulated figures*, N. Badler, D. Zeltzer, B. Barsky (ed.). Morgan-Kaufmann, 1990.

Barraquad, J., B. Langlois, & J. Latombe. *Numerical Potential Field Techniques for Robot Path Planning.* Technical Report STAN-CS-89-1285, Department of Computer Science, Standford University, October 1989.

Donald, Bruce. A Search Algorithm for Motion Planning with Six Degrees of Freedom. *Artificial Intelligence, 31 (1987).* 295-353.

Esakov, J., Norm Badler, & Moon Jung. An Investigation of language input and performance timing for task animation. In *Graphics Interface '89*, pp. 86-93, Morgan-Kaufmann. Waterloo, Canada, 1989.

Esakov, J. & Norm Badler. An architecture for high-level human task animation control. In Paul A. Fishwick and Richard B. Modjeski, editors, *Knowl-*

edge Based Simulation: Methodology and Application, Springer Verlag, Inc., 1990.

Kalita, J.K., & Norman Badler. A Semantic Analysis of a Class of Action Verbs Based on Physical Primitives, *Annual Meeting of the Cognitive Science Society*, Boston, July, 1990(a), pp. 412-419.

Kalita, J.K. *Analysis of Action Verbs and Synthesis of Tasks for Animation.* Ph.D. Dissertation. Computer and Information Science. Univ. of Pennsylvania. 1990(b).

Kalita, J.K., & Norman Badler. Interpreting prepositions physically, *Annual Meeting of the American Association for Artificial Intelligence*, Anheim, California, 1991.

Lozano-Perez, T. Spatial Planning: A Configuration Space Approach. *IEEE Transactions on Computers.* Vol C-32. No. 2. (1983).

Lozano-Perez, T. Task Planning in: Brady M (ed.) *Robot Motion: Planning and Control.* MIT press, Cambridge, MAss, 1985.

Lozano-Perez, T. A Simple Motion-Planning Algorithm for General Robot Manipulators. *IEEE Journal of Robotics and Automation*, Vol RA-3, No. 3, June 1987.
1987.

Monheit, Gary, & Norman I. Badler. *A Kinematic Model of the Human Spine and Torso.* Technical Report MS-CIS-90-77, Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA, 1990.

Phillips, Cary, Jianmin Zhao, & Norman I. Badler. Interactive Real-time Articulated Figure Manipulation Using Multiple Kinematic Constraints. *ACM Computer Graphics 24,2 (1990).* pp. 245-250.

Phillips, Cary & Norman I. Badler. Interactive Behaviors for Bipedal Articulated Figures. To appear in the Proceedings of SIGGRAPH 1991.

Zhao, Jianmin, & Norman I. Badler. *Real Time Inverse Kinematics with Spatial Constraints and Joint Limits.* Technical Report MS-CIS-89-09, Computer and Information Science, University of Pennsylvania, PA, 1989.

AUTHOR BIOGRAPHIES

MOON RYUL JUNG is a Ph.D. student in Computer Science at the University of Pennsylvania. He received the BS degree in Computer Science and Statistics from Seoul National University, Korea, in 1980, the MSc in Computer Science from Korean Advanced Institute of Science and Technology in 1982. His research interests include human figure modeling and animation, human figure motion planning, reactive/incremental planning, and natural language semantics.

JUGAL KALITA graduated from the Indian Institute of Technology, Kharagpur (B.Tech, 1982), the University of Saskatchewan, Saskatoon (M.Sc., 1984), and the Univeristy of Pennsylvania, Philadelphia (Ph.D., 1991). He is an assistant professor of computer science at the University of Colorado at Colorado Springs. His research interests include natural language processing, expert systems, and computer graphics especially task-based human figure simulation and animation.

Dr. NORMAN I. BADLER is the Cecilia Fitler Moore Professor and Chair of Computer and Information Science at the University of Pennsylvania and has been on that faculty since 1974. Active in computer graphics since 1968 with more than 85 technical papers, his research focuses on human figure modeling, manipulation, and animation. Badler received the BA degree in Creative Studies Mathematics from the University of California at Santa Barbara in 1970, the MSc in Mathematics in 1971, and the Ph.D. in Computer Science in 1975, both from the University of Toronto. He is Co-Editor of the Journal *Graphical Models and Image Processing.* He also directs the Computer Graphics Research Laboratory with two full time staff members and about 40 students.

WALLACE CHING graduated from Oxford University, England in 1988 with a B.A. in Engineering Science. He is a Ph.D student in Computer Science at the University of Pennsylvania. His research interest includes robotics path planning, human figure motion simulation, dynamics-based motion simulation and computer graphics.