ELSEVIER

Automation in Construction 3 (1994) 79-86

AUTOMATION IN CONSTRUCTION

Construction process simulation with rule-based robot path planning

R. Stouffs *,[a], R. Krishnamurti [a], S. Lee [a], Irving J. Oppenheim [b]

[a] *Department of Architecture, Carnegie-Mellon University, Pittsburgh, PA 15213, USA*
[b] *Departments of Architecture and Civil Engineering, Carnegie-Mellon University, Pittsburgh, PA 15213, USA*

Abstract

A rule-based simulation program is developed for application to building construction. From a specified task schedule, the program generates and simulates a motion path for each robot action, avoiding obstacles and incorporating interaction, safety and other considerations.

Key words: Building construction; Simulation; Path planning

1. The RUBICON Project

Simulation refers to the computational modelling of a process described by the activity network of construction tasks, the spatial description of each robot action, and an accompanying representation of the evolving building at each state in time. A simulation can be used to make observations on the feasibility of each task and to maintain a running measure of construction time and cost. This simulation can also provide a graphical display of the building project. In general, a simulation can be used to study the productivity of alternate construction plans and alternate resource mixes.

2. Project example and scope

For the demonstration of the RUBICON simulator we chose the construction of a typical Japanese precast concrete residential building. Fig. 1 illustrates a rendering of three stories of a typical 5 story, 40 unit building; Fig. 2 shows the floor plan of a single unit.

The simulation requires a building construction task plan describing the construction elements and the construction process as a task schedule. The simulator translates each task into a robot motion plan using a rule-based description of the robot agents. These motion plans reflect the respective robot's motional capabilities and limits, and they avoid any collision. The simulation also rejects impossible tasks and allows for a variety of different robot types. The result of the simulation consists of a graphical visualization of these motion plans, together with the building under construction.

3. Task planning

The input to the simulator consists of a task plan. This is a detailed plan describing the construction elements and the construction process

---

* Corresponding author.

SSDI 0926-5805(94)00010-K

as a task schedule. Each task is to be performed by a human or a robot crew. To design a task plan, one must determine the crew resources, both robot and human, and determine the construction elements and the construction sequence, that is, the order in which the different components will be handled. Subsequently, a task file can be built, specifying the construction elements, i.e., their name and geometry, and the sequence of tasks.

Declare an element (type) as

element length width height

An example of a typical construction element may be

WALL-PANEL-1 13.20 0.10 2.40

A task line specifies the estimated duration, the construction component to be handled, its start and final position and orientation and the performing crew or robot type. A component name is specified as

element.floor.part

Declare a robot task as

M *duration component*

(*start-position*, *start-orientation*)

robot-type (*end-p.*, *end-o.*)

A typical robot task may be: "Move Wall Panel Mark 1, Floor 1, Part #5 (WALL-PANEL-1.1.5) from truck site (0.0, −4.0, 2.0) to position *x-y-z* (54.4, 6.6, 1.2) with the Robot Overhead Crane. Estimated time is 0.16 days". The corresponding input line reads

M 0.16 WALL-PANEL-1.1.5

(0.0 −4.0 2.0 0 0 0)

crane_hoist (54.4 6.6 1.2 0 0 −90)

Declare a human crew task as

P *duration component* (*start-position*,

start-orientation) *crew*

A typical human crew task may be: "Attach Wall Panel Mark 1, Floor 1, Part #5 (WALL-PANEL-1.1.5) at truck site (0.0, −4.0, 2.0) to Robot

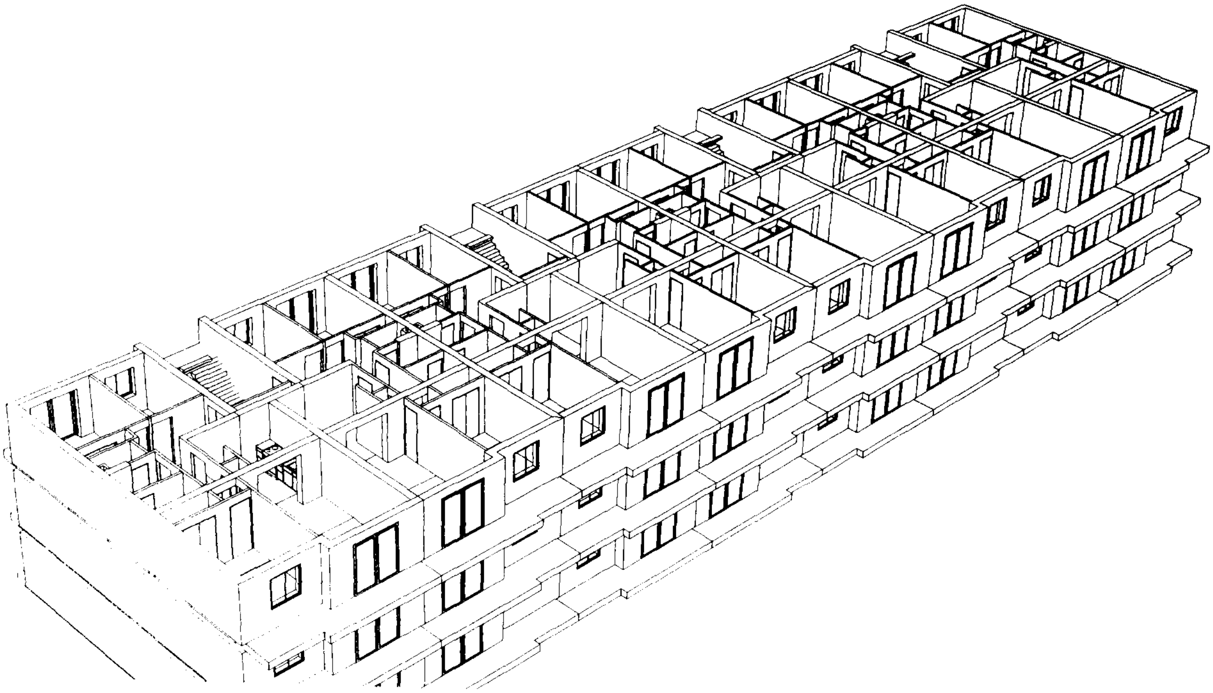


Fig. 1. Typical 40 unit building during construction.

Overhead Crane. Estimated time is 0.03 days". The corresponding input line is

```
P 0.03 WALL-PANEL-1.1.5 (0.0 −4.0 2.0 0 0 0)
    connecting_crew
```

Part of the design work may be automated with the help of a planning software such as PLANEX [1]. PLANEX is a knowledge-based expert system for process planning. It generates a project activity plan that is a general description of the sequence of activities involved in the process. Such an activity plan must be post-processed, either manually or automatically, to deliver a task plan that is detailed enough for the RUBICON program. A purpose-built task planner that would interface between RUBICON and an existing construction management software needs to address the issues of project scheduling, building geometry, construction elements, activity duration, task sequencing and resource allocation.

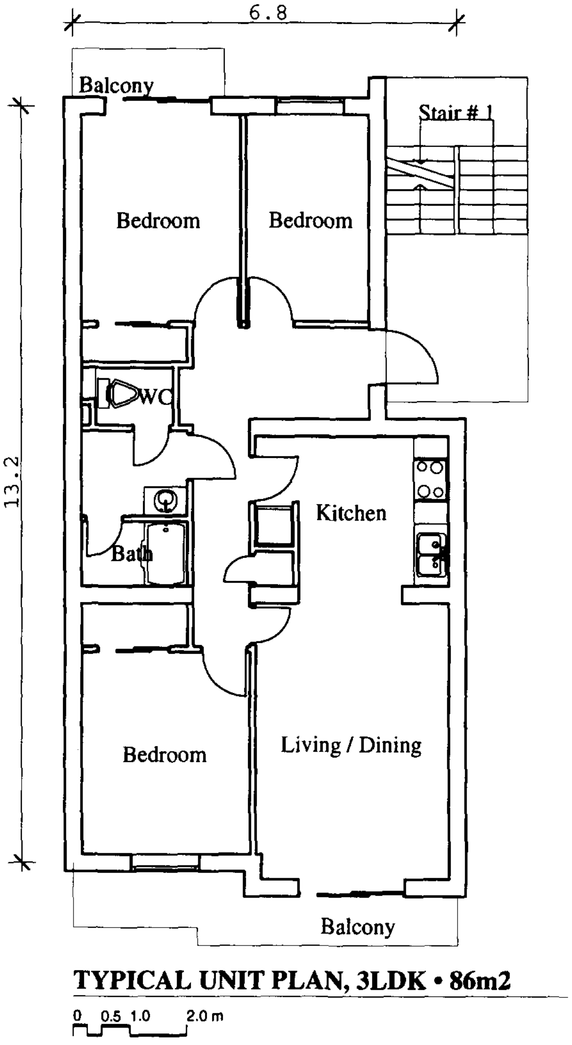


Fig. 2. Floor plan of a single unit.

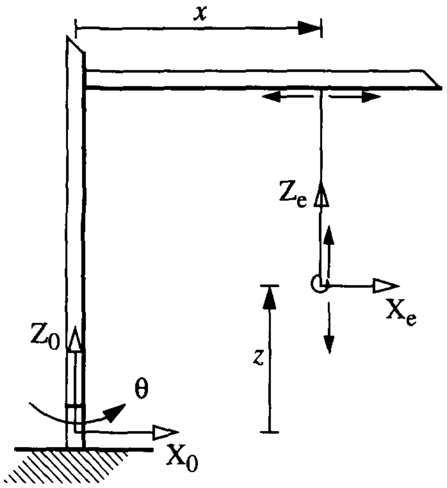
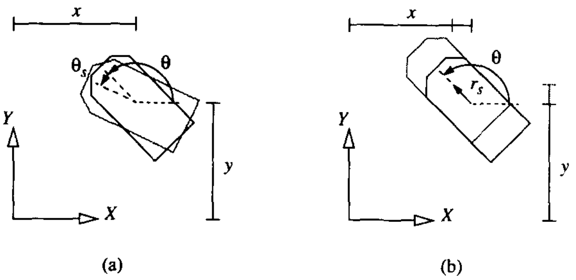


Fig. 3. A robot crane.

4. Robot motions

Robots for construction are characterized by their ability to lift, move and place sizable construction components from the delivery location to the placement location. We consider two different robot types: a robot crane for handling



(a) (b)

Fig. 4. A robot towmotor with (a) stationary rotational motion and (b) translational motion along its axis.

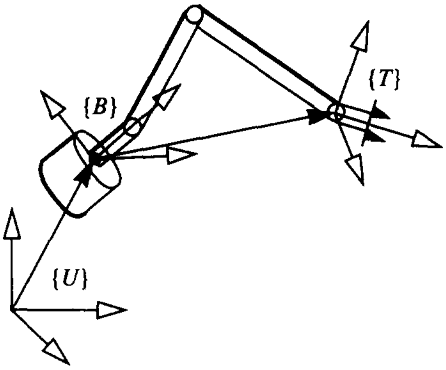


Fig. 5. Three frames to describe the configuration of a robot.

heavy components; a robot towmotor for palettized materials. These two types are illustrated in Figs. 3 and 4 respectively.

The description of a robot encompasses a representation of its configuration, i.e., position and orientation, over time, its motional capabilities and its configurational limits. We use a fixed number of frames and transforms to describe a robot's configuration at a particular time, introduce a set of control parameters to capture possible configurational changes over time, associate motion rules to these control parameters to describe a robot's motional capabilities, and add extremum values to the control parameters to represent the configurational limits.

A robot can be described in terms of the configuration of its components to each of which we rigidly attach a coordinate system or frame [2]. See Fig. 5. A transform describes one frame relative to another. A minimal description of a robot involves the following transforms: ${}^{U}_{B}\mathrm{T}$ describes the base frame relative to the universal frame; ${}^{B}_{T}\mathrm{T}$ describes the tool frame with respect to the base frame; and ${}^{T}_{O}\mathrm{T}$ describes the object frame with respect to the tool frame. The object frame is described relative to the universal frame by the composite transform ${}^{U}_{O}\mathrm{T} = {}^{U}_{B}\mathrm{T}{}^{B}_{T}\mathrm{T}{}^{T}_{O}\mathrm{T}$.

The ability of these transforms to change over time and, as a result, the relative configuration of the bodies they describe, is captured in a set of control parameters. For each transform, these parameters correspond to the respective degrees of freedom (DOF) of the body relative to its reference frame. Physically, the relationship between the end-effector and the base of the robot is defined by a set of links and joints, generally formed into an open kinematic chain. These joints can be either rotational (change of orientation) or prismatic (translational change). The number of DOF of the robot's end-effector, with respect to the base and for an open kinematic chain, corresponds in most cases to the number of joints. The set of valid control parameters consists of x, y, z, ψ, ϕ and θ, where x, y, z define translations parallel to the respective axes and ψ, ϕ, θ define rotations about the X-, Y- and Z-axis, respectively.

The motional capabilities of the base of a robot often are dependent on the current configuration of the base and, thus, defined locally. However, the configuration itself is defined relative to a reference frame that is fixed with respect to the body's motion. Motion rules are conceived to describe a robot's motional capabilities possibly independent of its configuration. Foremost, however, they form a tool to embody the discretization of time in the simulation process. A motion rule generally defines the value of a control parameter at time $t + \Delta t$, where Δt denotes the time step, in terms of the control parameter's values at time t and their initial and step values. The initial values are the values of the control parameters at time 0. Under the assumption of constant velocity, for each control parameter we can specify a motion step as a constant value. Each parameter also has a minimum and maximum value specified in accordance with the relative workspace of the body. This workspace is roughly the volume of space which the body can reach in at least one orientation, relative to its reference frame [2].

5. Robot behavior

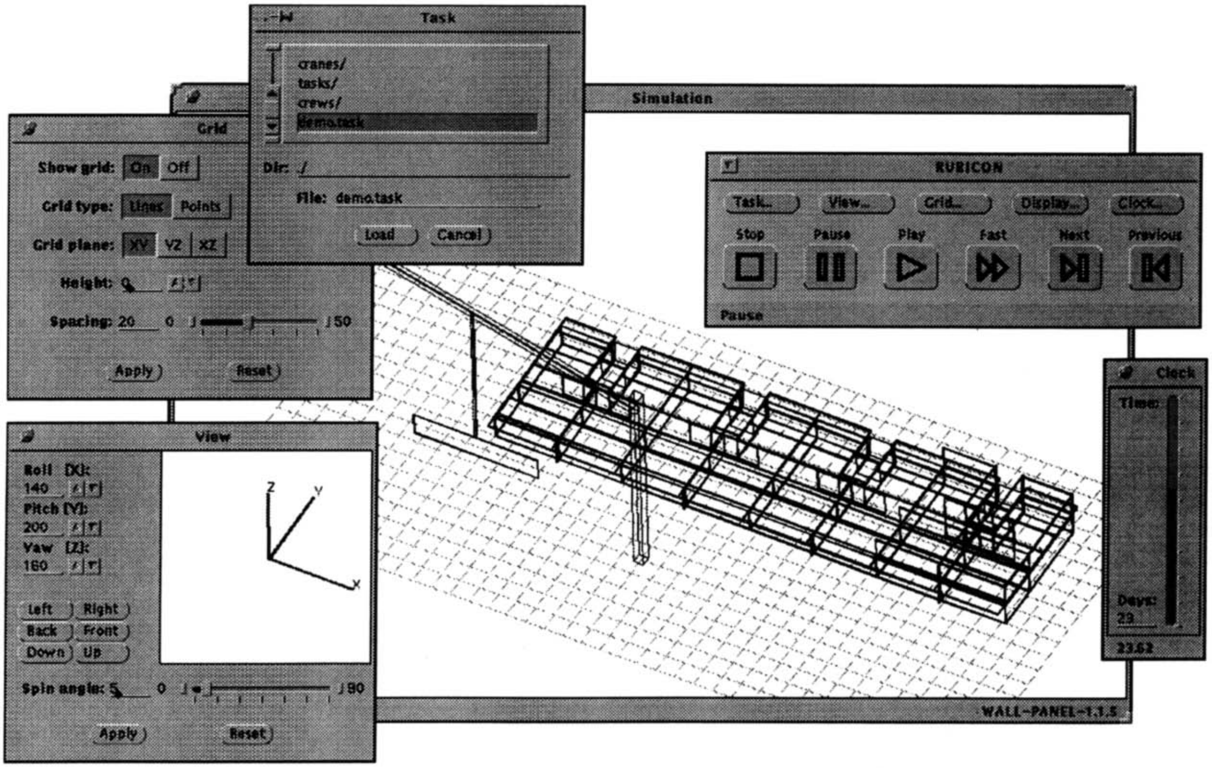
In the simulation, we are concerned with the role that a particular robot plays in the construction process and with its place in, and its relation to, the environment, that is, the site as well as the building under construction, the human labor crews and the other robots involved in the construction. This relation and, therefore, the robot's behavior, is constrained by physical obstacles, task-specified and other interactions, and safety and other considerations. A motion language is developed that serves as a means to specify this behavior and, in particular, to specify the order in which the allowed motion steps should be proposed, for each robot or body. The choice of a specific motion step is dependent on the particular situation of the robot at the moment, according to the specified behavior. Then, all motion rules that contain the motion step in question are invoked, and the corresponding control parameters are updated.

The basic building blocks of the motion language are the motion command elements. Motion commands may be grouped into series of motion steps using control structure elements. The language also allows for the use of variables; it is completed with a set of functions that may be used within assignments or as command arguments. For a more elaborate explanation of the elements of the language, we refer to [3].

6. The RUBICON program

The input to the RUBICON program consists of two kinds of files. One file contains the task plan, that is, the description of the components and of the construction process as a sequence of tasks each of which is performed by a human or robot crew. For each robot agent specified in the task plan, a motion file is read in that contains a description of the motional capabilities of the robot and the motion rule set describing its intended behavior, expressed in the motion language specified above.

The output of the RUBICON program is a graphical simulation of the construction process



Dir: ./
File: demo.task

Show grid: On
Grid type: Lines
Grid plane: XY
Height: 0
Spacing: 20 0 150

Roll [X]: 140
Pitch [Y]: 200
Yaw [Z]: 160
Spin angle: 5 0 180

Pause

Time:
Days: 23
23.62

Fig. 6. Snapshot of the RUBICON graphical user interface.

Fig. 6. Snapshot of the RUBICON graphical user interface.

as specified in the task plan, with a visualization of the motional actions of the robot agents and of the transportation of the construction components by these agents, and with a specification of the process time. The output is controlled through an audio/video-like control panel with buttons for play, fast play, next, previous, pause and stop. Fast play is achieved by unmaterializing the robot agent; next and previous instantly jump to the next, respectively previous, construction component.

Other interface panels allow the user to choose and load a new task plan and corresponding motion files, alter the 3D-viewing parameters, alter the displayed grid, and view the current process time (see Fig. 6).

7. Simulation results

The RUBICON program serves the project planning engineer in studying alternate task plans,

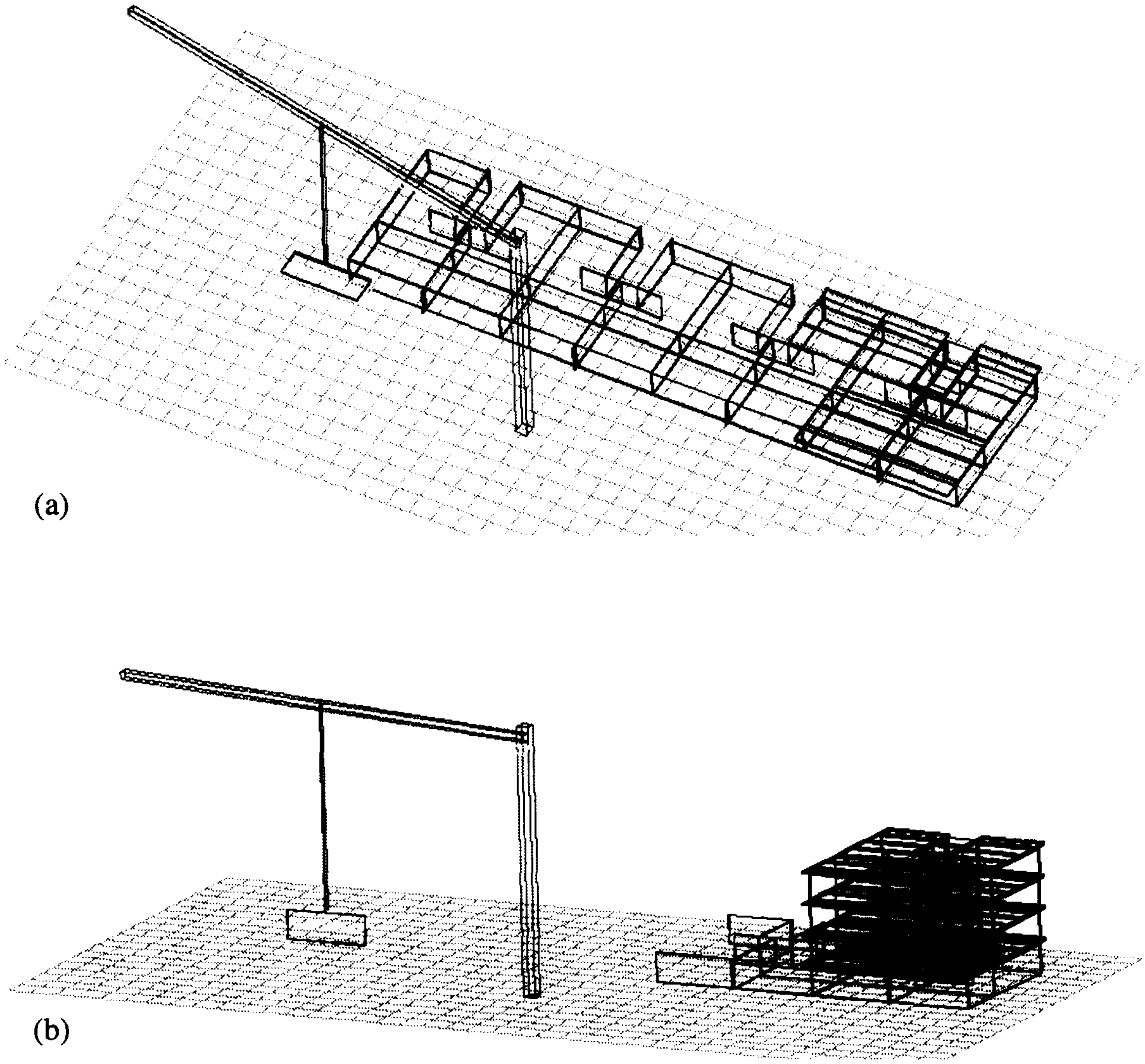


Fig. 7. Alternate task sequences (snapshots from the simulation): (a) floor-by-floor and (b) staircase-by-staircase.

alternate resource mixes and the use of alternate robot types in the construction process. Results of the program include productivity measures, such as total project time and number of human crews and robot agents used. The following are some simple examples illustrating the possibilities of the program to study the results of alternate input.

The simulation demonstrates how alternate task sequences, such as floor-by-floor or staircase-by-staircase construction of residential building examples, result in different robot paths and, therefore, productivity (see Fig. 7).

Different project durations may result from alternate resource mixes; e.g., given the two tasks of robot panel connection and panel grouting the

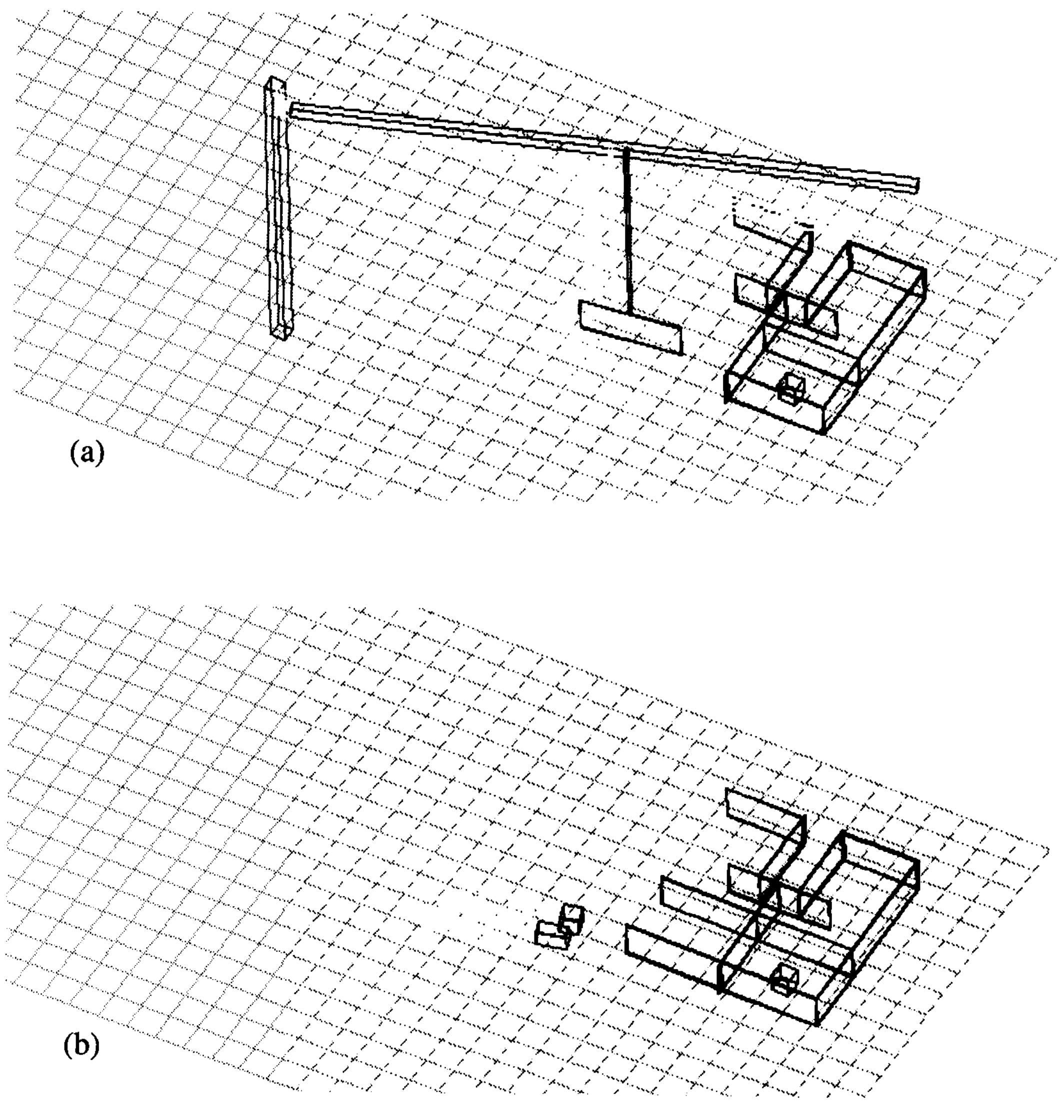


Fig. 8. Dual robot task plan (snapshots from the simulation): (a) a crane positions the wall-panels and (b) a towmotor places the palettized materials.

difference in duration for either one or two human crews—one connecting and one grouting—is obvious.

The productivity of a robot crane depends on the position of the crane on the site, e.g., mid-site or at the edge of the site, and on its ability to be repositioned during construction. Multiple cranes also influence productivity.

The simulation program has the ability to recognize impossible plans and to reveal inefficiencies in task plans; e.g., when using a towmotor, it is imperative that there exists a path from the pick-up location to the delivery location, large enough for the towmotor. Also, whether an efficient path exists for the towmotor may depend on the order of positioning of the construction components, such as wall panels.

Productivity may be further enhanced through the (simultaneous) use of different robot types, e.g., crane robot and towmotor, using each for materials it handles best. Fig. 8 illustrates the use of two different robot types in a construction.

Other considerations that are important are motion constraints such as the reach of a crane, or safety considerations that regulate robot–human crew interaction, e.g., a load may not pass overhead of a grouting crew.

8. Conclusion

A distinctive feature of robotic construction is the need for a path to be associated with each robot action. The simulation program described here incorporates, within it, an automatic path planner which returns a path for each action. The path planner was developed as a rule-based planner. The kinematic capability of each robot is described by a motion rule set, and rule chains for finding paths can subsequently be entered. Two simple examples were used within this study: a crane, and a towmotor. The simulation program, RUBICON, is demonstrated for a residential building example constructed with precast concrete panels. Applicable task plans are manually generated; a task plan produced by a knowledge-based expert system, PLANEX, is also examined. Examples are chosen to show that the RUBICON simulation can recognize impossible plans, can reveal inefficiencies in task plans, can illustrate the influence of crane placement, and can simulate influences on productivity of human and robot crew interaction.

RUBICON may serve as a tool for construction company engineers to perform studies on real project task plans; engineers/developers may use RUBICON to develop better task planners.

Acknowledgements

The development of the RUBICON prototype described in this paper was funded in part by the Japan Research Institute.

References

[1] C. Zozaya-Gorostiza, C. Hendrickson and D.R. Rehak, *Knowledge-Based Process Planning for Construction and Manufacturing*, Academic Press, San Diego (1989).

[2] J.J. Craig, *Introduction to Robotics: Mechanics and Control* (2nd ed.), Addison-Wesley Reading, MA (1989).

[3] R. Stouffs, R. Krishnamurti and I.J. Oppenheim, A behavioral language for motion planning in building construction, *Automat. Construct.*, submitted.