

TEMPORAL LOGICS MEET TELEROBOTICS

Eric RUTTEN , Lionel MARCÉ

IRISA / INRIA - Rennes

F-35042 RENNES Cedex FRANCE

e-mail: rутten@irisa.UUCP, marce@irisa.UUCP

Abstract

The specificity of telerobotics being the presence of a human operator, decision assistance tools are necessary for the operator, especially in hostile environments. In order to reduce execution hazards due to a degraded ability for quick and efficient recovery of unexpected dangerous situations, it is of importance to have the opportunity, amongst others, to simulate the possible consequences of a plan before its actual execution, in order to detect these problematic situations. Hence the idea of providing the operator with a simulator enabling him to verify the temporal and logical coherence of his plans. Therefore, we intend to use the power of logical formalisms for representation and deduction purposes. Starting from the class of situations that we want to represent, we will adapt a *STRIPS*-like formalism and its underlying logic to the simulation of plans of actions in time. The choice of a temporal logic enables us to build a world representation, on which the effects of plans, grouping actions into control structures, will be transcribed by the simulation, resulting in a verdict and information about the plan's coherence.

1 Introduction

The specificity of telerobotics is the presence of a human operator in the classical *perception-decision-action* loop. His work is then to supervise execution and, most importantly, to decide of the tasks to be completed by the teleoperated robot, and of their ordering. It is in this context that decision-making assistance to the operator seems necessary, in order to test and validate the plans he conceives, before executing them, and this especially in hostile environments. In the case of space telerobotics, when communications take place with a very remote robot site, the time elapsing between operator reaction and decision, and robot action, can be very long; furthermore, manipulation is made difficult by the physical characteristics of the environment, unnatural to the human operator. All this degrades the ability for quick or efficient recovery, by the operator, of errors or unexpected accident.

Hence the idea of providing the operator with a decision assistance system, comprising functionalities such as planification, memorizing of events, managing of warnings and signals, and simulation, in order for him to have an information as complete as possible about the state of the environment he is working upon, including the robots he teleoperates, its past evolutions, and the possible futures entailed by the actions he could undertake. The decision assistance function of a teleoperation system goes along, and works together with other essential features: execution control and monitoring, at various levels between autonomy and manual command,

sensor information treatment and fusion, or interactive modelling of the environment. In the frame of a system comprising all these elements, a simulator can enable the operator to verify the logical and temporal correctness of his plans with regard to their effects on the world being worked upon.

A decision assistance tool needs as its base a representation of knowledge, on which various required treatments can be performed, providing a formal frame in which to store the gathered information. The expressiveness and inferential power of logics make them a particularly adapted tool therefore, especially as recent logical systems have been built in order to capture notions otherwise only difficultly mastered, such as time by temporal logics. Our work is thus directed towards a meeting of the interests of telerobotics and of temporal logics, being branches respectively of robotics and artificial intelligence, through characteristic, paradigmatic problems to be solved, leading to the construction of meaningful formalisms. As to what knowledge we have to represent, in the particular case interesting us, it consists of the world, the environment in which things happen (its state, and the laws describing what can happen into it, and what not), and the changes that can occur, especially the actions that the robot can execute. Research in artificial intelligence associated with robotics has already provided us with concepts and techniques addressing these knowledge representation issues.

2 Knowledge representation issues

A classical model of robots acting on an environment is the *STRIPS* planner (the *Stanford Research Institute Problem Solver*) [7], using a formalism based on first-order logic. In *STRIPS*, the world is represented by a set of predicates describing its state, and an action is represented by three lists: the *preconditions list*, specifying the predicates that must be found to hold in the current state of the world, in order for the action to be executable; the *delete list*, composed of the predicates that do not hold after the execution of the action, and are retracted from the world state, as a negative effect of it; the *add list*, of which the predicates are added to the world state, as a positive effect of the action. In this way, an action is an operator transforming a situation of the world into another. The plans generated by *STRIPS* are sequences of such actions, forming a transformation from the initial situation to a final situation including the goal of the plan.

The limitations of this model are in the strict linearity of the plans produced, and in the underlying logics. Since then, other studies have achieved the construction of models allowing a less restricted structure of the plans [6], through less commitment as to the order between actions, e.g. using a partial order on their set ... One of the directions of extension of these formalisms has been that of **temporal logics**, dealing explicitly with time [2, 10], in order to have a representation of duration, change and simultaneity. These temporal logics have been used for plan generation, the planning process then consisting in mapping the actions in time in order to obtain the goal state from the initial one through the execution of the plan. We will explore them here as a knowledge representation and deduction tool. A bibliographical study shows a great variety of existing formalisms, many of them still in evolution. Amongst them, a choice has to be made of the concepts applicable to the class of problems interesting us.

Our approach deals with the simulation of plans composed by the human operator in

teleoperation. In order for him to have at his disposal a facility for writing, reading, and modifying the plans, we will provide him with a representation for the control structures of **plans and actions**. The representation of actions is expanded with a durational information, and possibilities of a hierarchy of subplans provide modularity in plan construction. A language is defined for plan writing, introducing basic operators, as well as more customized ones, to build complex, yet clear, control structures.

The process of **simulation** then reflects the effects of the plan given as input on the world representation. Each of the primitive actions of the plan is simulated, following the control structure of the plan framing them. The output consists in the verdict about the plan executability, and information about the states of the world reached by the possible alternative executions, due to imprecisions in the actions ordering. The state of the world can be examined along its changes, that are traced. In case of an event compromising the plan's executability, warnings and error messages can be issued, guiding the fixing of the plan.

3 Temporal logics

We will here briefly present the temporal logic formalism that we have chosen, inspired by Allen [3] and Shoham [11]. The reasons for the choice of this kind of formalism, in preference to modal tense logics like those used in program specification and verification, is that their origin in “*theoretical robotics*” gives them a more immediate expressivity for the class of problems interesting us. Furthermore, a complex logical system is not yet needed in our approach, even if the use of more elaborate logic operators can provide interesting perspectives.

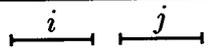
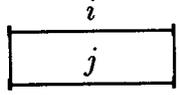
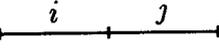
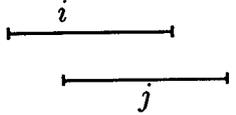
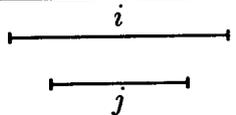
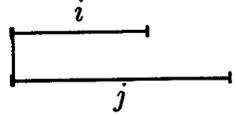
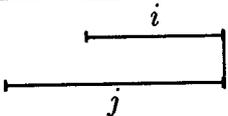
Intervals We will adopt the notion of intervals being “chunks” of time, between which relations have been determined by Allen, and are shown in figure 1. These relations can be grouped in a disjunctive relation if the precise relation is not known; e.g. $((I_1 r_1 I_2) \vee (I_1 r_2 I_2))$ can be written: $(I_1 (r_1 r_2) I_2)$. Moreover, a transitivity table has been built on these bases [2], allowing to determine, knowing two relations r_1 and r_2 between three intervals I_1 , I_2 and I_3 , i.e. $I_1 r_1 I_2$ and $I_2 r_2 I_3$, a third relation r_3 obtained transitively: $I_1 r_3 I_3$.

Temporal facts As summarized by Shoham, facts represented in classical first order logic by predicates, are here “reified” into temporal facts. Those take the form: $true(I, P)$, meaning that the fact P holds on the interval I . The rules of classical logic are assumed to hold, in the way established by Shoham [11], stating that, for example, $true(I, P_1 \wedge P_2)$ is satisfied if and only if both $true(I, P_1)$ and $true(I, P_2)$ are, and that $true(I, \neg P)$ holds if and only if, for no subinterval I' of I , we can have $true(I', P)$.

Classification by inheritance Apart from the relations transitivity table, Shoham gave a definition for another way of getting new temporal information from known facts, by classifying temporal facts according to the relation of their truth over one interval to their truth over other intervals [11]. Examples of classes are: *downward-hereditary* (e.g. “The robot travelled less than two miles.”, when true over an interval, is true over all its sub-intervals), *upward-hereditary* (e.g. “The robot travelled at the speed of two miles per hour.”, when holding for all the proper sub-intervals of a nonpoint interval, holds for the nonpoint interval itself), *liquid*:

both *upward-hereditary* and *downward-hereditary* (e.g. “The robot’s arm was in the GRASPING state”), *solid* (e.g. “The robot executed the NAVIGATE procedure (from start to finish)” never holds over two properly overlapping intervals), ... The temporal facts (*true(I,Fact)*) described in the previous paragraph can be incorporated to the *liquid* class. The *solid* class can be used to describe the execution of an action or a plan (e.g. by *exec(I,Plan)*.)

We are thus provided with elements, from which a representation formalism can be developed, allowing us to describe a world, and to manipulate this information, using rules with which further information can be obtained, i.e. reasoning can be made.

relation	symbol	symbol for the inverse relation	graphical example
<i>i before j</i>	<	>	
<i>i equals j</i>	=		
<i>i meets j</i>	m	mi	
<i>i overlaps j</i>	o	oi	
<i>i during j</i>	d	di	
<i>i starts j</i>	s	si	
<i>i finishes j</i>	f	fi	

After Allen [2].

Figure 1: The 13 possible relations between two intervals *i* and *j*.

4 Plans and actions

As in our approach, a human operator is involved in the process, i.e. he writes himself the plans for the robot, we have to provide him with a language in which to express the plans,

that will allow him to build complex, yet clear, control structures. The basic elements of this language are the *primitive actions*, which have a form inspired by *STRIPS*. These actions can be grouped into structures determined by *control operators* specifying in what way the actions they frame will take place in time.

Primitive Actions A classical representation for an action is that of *STRIPS*. We however want to add a temporal information to this, which is absent in *STRIPS*. We will therefore associate, with each action occurrence or execution, an interval, the extent of which is the duration of the action. An action is noted as shown in figure 2.

representation	example
$action(\textit{name(parameters)},$ $\textit{duration},$ $\textit{preconditions},$ $\textit{negative effects},$ $\textit{positive effects}).$	$action(\textit{putoff(Cube1,Cube2)},$ $3.14 \textit{ s},$ $[\textit{clear(Cube1),on(Cube1,Cube2)}],$ $[\textit{on(Cube1,Cube2)}],$ $[\textit{clear(Cube2),on(Cube1,table)}]).$

Figure 2: Actions representation.

As Vere did [12], we make the “changes on termination” assumption, deciding that all the changes entailed by the action occur at the end of its occurrence interval. Nevertheless, the preconditions have to be holding on all the interval of the action. Such an action takes place in time in the way shown in figure 3. The “changes on termination” assumption, however, is not limitative: the definition of *compound actions*, presented a little further, allows to define actions having effects in another way. At the moment of simulating such an action, the deduction tools of the temporal logics will be useful to find out whether the preconditions hold over the interval, and how the effects interact with other facts in the world representation.

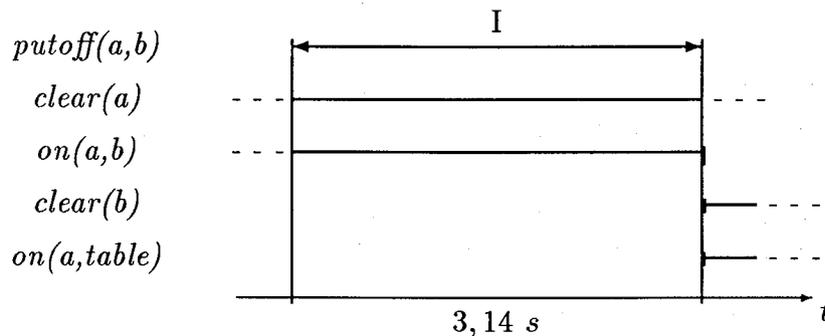


Figure 3: Example of an action: $putoff(a,b)$ in time.

Plans A plan, as said earlier, is considered here as a set of actions, provided with a *control structure*. A plan is then composed of subplans, which are plans themselves, recursively, down

to the *primitive actions* defined earlier, which each constitute a plan themselves. The basic constructs of the language are:

sequentiality : noted $seq(<subplans\ list>)$. For a subplans list $[P_1|P]$, where P_1 is the first subplan of the list, and P the remainder of the sequence, this control operator states that the subplans of the list are executed one after the other, in the order of the sequence, as shown in figure 4.

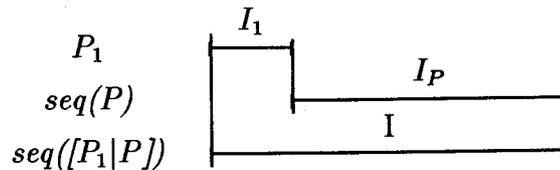


Figure 4: A sequence in time.

parallelism : noted $par(<subplans\ list>)$, where the list of subplans contains the plans constituting each a branch concurrent to the others. In this construct, all branches B start together, and the parallelism ends when all branches have ended, i.e. with the longest lasting branch, B_P , as shown in figure 5.

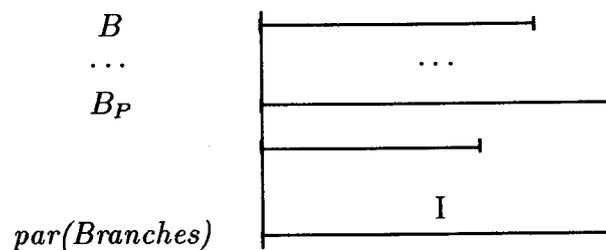


Figure 5: Parallelism in time.

conditionality : noted $cond(C, P_{true}, P_{false})$. The condition C is evaluated at the beginning of the interval, and according to the result, the corresponding subplan is executed.

Compound Operators and Actions From these operators, *compound operators* can be constructed, like, for example, *conditional iteration*, defined recursively as:

```
while( condition, iteration-body )
≡
cond( condition,
      seq([ iteration-body,
            while(condition, iteration-body) ]),
      nothing ).
```

where *nothing* is a null action, taking no time, requiring no precondition, and entailing no effect. In the same manner, *compound actions* can be defined, using a plan, in the following way: $action(name(arguments), plan)$. This allows to consider actions with effects dispatched along their duration, or depending on the context where they are executed. For example, a way of defining an action a realizing effects inside (e.g. at a time d_1 after the beginning of) its interval (of total duration $d = d_1 + d_2$), is to decompose it in the following way: $action(a, seq([a_1, a_2]))$, where we have an action a_1 of duration d_1 , with all the preconditions and effects of the action a , and another action a_2 , having the preconditions and not the effects of a , being there to continue the verification of the precondition.

This language should enable the writing of complex plans, composed of actions defined by their effects on the world. The temporal dimension of these actions and plans having been specified, we will now see how simulation on these bases is considered.

5 Simulation

The simulation of a plan then consists, given a representation of the world to which it is applied, in a modification of this representation by transcribing onto it the effects of the actions of the plan, taken following the control structure leading their flow.

The simulation process The world representation consists of a set of temporal facts of the kind described earlier. A classical example is that of a blocks world, as shown in figure 6. The intervals associated to the facts (I_1, I_2, I_3) have unconstrained ends: the truth of the facts holds from the beginning of each interval (in the “past” or “present”, before the action) to an undetermined, or uninstantiated “future” (which can be determined by the events in the world, later in time.) A set of predefined actions is at the disposal of the planner, who can build a plan, constructed with control operators like those seen before. This plan is given as input to the simulation process, that begins with decomposing it down to primitive actions in basic control structures. Then, primitive actions are simulated one after the other, in a succession determined by the control structure of the plan.

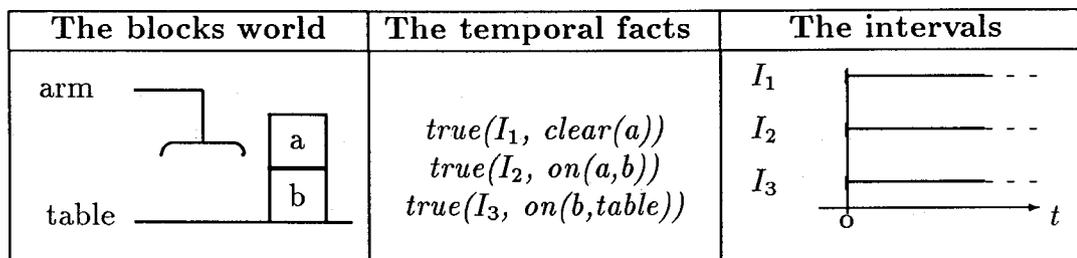


Figure 6: Example of an environnement and its representation.

Primitive Actions The simulation of each primitive action, on an interval of which the extent is given by the duration, consists in constraining the truth intervals of the temporal

facts concerned so that they verify the preconditions and transcribe the effects in the world representation. For example, the application of the action *putoff(a,b)* shown earlier in figure 3, on the blocks world of figure 6, gives the new set of temporal facts, after the action, shown in figure 7. In this example, the application of the action *putoff*, on an interval *I*, entails that:

- I_1 and I_2 , being the intervals corresponding to the preconditions, must contain I ;
- I_2 , as a result of the negative effect of *putoff*, is constrained to end with I (i.e. I_2 (*finishes equals finished-by*) I);
- two new intervals, I_4 and I_5 , are associated with the positive effects, and constrained to be *met-by* I (i.e. I *meets* I_4 and I *meets* I_5), which determines their beginning, but leaves their end unconstrained yet.

I_3 , not being concerned by this action, is not imposed any particular constraint.

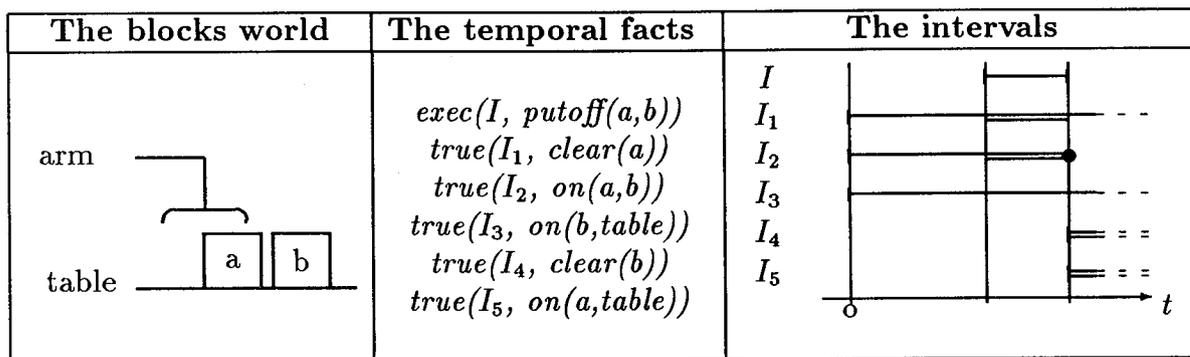


Figure 7: The environment after the execution of *putoff*.

At this stage, the temporal logic is used to check the coherence of the new constraints introduced with the others, and to verify the compatibility of the effects with the temporal facts according to rules specified in the world representation. This can be done using classical deduction methods. Another method for maintaining a coherent set of temporal relations is to build a lattice of time-points, and to define retrieval and updating operations upon it [9]. An *incoherence* is said to be encountered in the simulation, if either a precondition fails to be satisfied, or a contradictory constraint or an incompatibility of the effects is detected.

Plans The simulation of plans such as those described before, then consists in the succession of simulations of the primitive actions they contain, following the order specified by the control structure. The task of simulating the plans can then be described as that of choosing the next primitive action to be simulated, and determining the associated interval:

sequentiality : the subplans are simulated in the specified order (see figure 4);

parallelism : here the choice is made between the actions of the different branches, the criterion being the following of the chronological order, the earliest terminating primitive action being simulated first. An interesting case is that of orderings between actions that are not uniquely determined: their intervals can have one of several possible relative

positions. In this case, the simulator will try each relation, "forcing" the intervals into an order, and simulating further for this possibility, until reaching the end of that simulation. Then a backtracking takes place, back to the last choice made, where, if another relation is left untried, it is taken as a new choice, and the simulation is resumed from this point.

conditionality : the satisfaction of the condition is checked for in the world representation, and according to the result, the corresponding subplan is simulated.

compound operators and actions are rewritten, following their definitions in terms of basic elements, and simulated in that state, as normal plans.

Results In this way, the tree of possible executions of the plan, each one related to a possible disposition of the actions in time, is explored. The progression in a branch of this tree stops in two cases: either the end of the plan is reached: we then have the result that it is executable, and that the consequent state of the world is the one reached at this point; either an *incoherence* is encountered, the result then consists in the verdict of unexecutability of the plan, with some information on the reasons of this failure, in order to help modifying the plan so that it would give a success. In the problem of detecting incoherences through simulation, it is important to make an exhaustive examination of the possible situations accessible in the model [8]. This stresses the need for methods of exploration through the tree of possible executions, in order to sort out the incoherent ones.

To summarize, a plan built as seen before can be simulated with regard to its effects on the environment being worked upon, and also to the "internal" coherence of the temporal ordering of its actions, along its control structure. Exploring the possible consequences of that plan provides an operator with an assistance for the evaluation of the correspondance between the effects of the plan and what is expected of its execution.

6 Perspectives

The problems addressed by our work are manifold, and the solutions proposed can be improved. The knowledge representation aspects of it must be extended and strengthened, in order to achieve the detection of difficult (e.g. indirect) incoherences. The treatment of indeterminacy in the actions order entails a problem of exhaustivity of the search in the tree of possibilities. All these problems are still worked upon, and will be confronted to experimentations on situations studied in collaboration with MATRA-Espace (in link with the European Space Agency), in the case of a teleoperated arm [4], and with the C.E.A. (the French Nuclear Energy Agency), for mobile telerobots. These experimentations will provide us with a testbed for our models, which will enable us to adapt them to the representational needs of the real world problems, and to develop the functionalities of our representation according to this feedback.

The extensions expected, and further studies to be made, will thus concern the representation of actions (alternative, or even non-deterministic in their consequences, ...), the development of control operators not derivable from the basic ones (synchronization on the satisfaction of a condition, ...), "intelligent backtracking" for a better search in the tree of

possibilities, other logical formalisms (modal logics, tri-valued logics, non-monotonic logics ...), as well in the way they are used in approaches close to ours [5], as in their application to less directly linked problems, such as distributed algorithms verification, ... The motivation of the application of logical formalisms to this kind of problems lies in their providing us with a clear knowledge representation frame as well as tools for manipulating this knowledge, i.e. reasoning. Our approach consists in applying these formalisms to simulation, as a hopefully profitable alternative to the more classical, yet difficult, plan generation paradigm [1]. The results will apply to the same issues, from the point of view of knowledge representation, and a connection can be made with temporal relations management techniques [9].

References

- [1] J.F. Allen, J. Koomen. Planning using a temporal world model. In *Proceedings of the IJCAI '83*, pages 741–747, Karlsruhe, August 1983.
- [2] J.F. Allen. Maintaining knowledge about temporal intervals. *Communications of the A.C.M.*, 26(11):832–843, November 1983.
- [3] J.F. Allen, P.J. Hayes. *Moments and points in an interval-based temporal logic*. Technical Report TR 180, University of Rochester, December 1987.
- [4] G. André, T. Blais. Space teleoperation and control concept, experimental evaluation for the Hermes robot arm (HERA). In C.A. Mason, editor, *Proceedings of the International Symposium on Teleoperation and Control*, Bristol, England, 12-15 July 1988.
- [5] G. Boy, N. Mathé. Using non-monotonic reasoning for operator assistant systems in reactive environments. In *Proceedings of the ESA-ESTEC Workshop on Artificial Intelligence Applications in Space Projects*, 15-17 November 1988.
- [6] D. Chapman. Planning for conjunctive goals. *Artificial Intelligence*, 32:333–377, 1987.
- [7] R. Fikes, N. Nilsson. STRIPS : a new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189–208, 1971.
- [8] K. Forbus. Intelligent computer-aided engineering. *AI magazine*, 9(3):23–36, Fall 1988.
- [9] M. Ghallab, R. Alami, R. Chatila. Dealing with time in planning and execution monitoring. *Robotics Research, R. Bolles (Ed.), MIT Press*, 4, 1988.
- [10] D. MacDermott. A temporal logic for reasoning about processes and plans. *Cognitive Science*, 6:101–155, 1982.
- [11] Y. Shoham. *Reasoning about change : Time and causation from the standpoint of artificial intelligence*. PhD thesis, Yale University, December 1986.
- [12] S.A. Vere. Planning in time : windows and durations for activities and goals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5(3):246–267, May 1983.