(Copia modificata per una migliore consultazione on-line)

Artificial Intelligence Applied to Design of Intelligent Systems (a Soft Computing Approach)

Litvintseva L.V. and Ulyanov S.V.
(Scientific Supervisor: Ph.Doctor in Mechanics and Control Engineering, State Doctor in Physics and Mathematics, Professor S.V. Ulyanov)
Universita Degli Studi di Milano,
Polo Didattico e di Ricerca di Crema, Via Bramante, 65-26013 CREMA

Content

P a r t 3 Symbolic Artificial Intelligence for High-level Intelligent Systems Design

Lecture N 14 Temporal Knowledge Representation in Intelligent Systems

In order to represent spatio-temporal and action knowledge and to reason correctly about time, space and actions a lot of logical models were constructed. Consider the problems of temporal, spatial and action model's development, and begin our discussion from temporal model's design problems.

Discuss the problem of temporal models design from the standpoint of modeling of properties of a time considered below.

1. The time properties in temporal models for intelligent systems

The main appointment of time is the fixation of the orientation of events, actions, processes flow from a past to a future through the present.
The following properties of a time we observe in real world.

A: *One Direction of time*
This means that *the orientation of events (or processes) flow is fixed in one dimensional temporal scale.*

In some temporal models this property is violated. As example, a temporal model for a quantum system of relativistic elementary particles interaction in electromagnetic fields can include different temporal scales. The same is true for an anisotropic or with different gravitational potential space-time continuum.

B: *Noninvertibility of time*
This property can be described as follows:
if event E*1* occurs earlier event *E2* then it is false that event E*2* occurs earlier event E*1*.

In the most of temporal models this property is accepted, but in contemporary physics there are some temporal models where this property is violated.

C: *Linearity of time*
This property means also the *transitivity property* of time. Formally it can be written as:
if event E*1* occurs earlier event E*2* and event *E2* occurs earlier event E*3* , then event E*1* occurs earlier event E*3* .

Linearity property is accepted in the most of temporal models for intelligent systems. However, there are models in which this property is violated. For example, in models of the psychological time a ramification of a time may be observed.

D: *Continuity property*

It means that *for all two moments of time, T*1 *and* T2*, ordered on some one dimensional continuum* (*an analogue of a time scale*) *there is always T*3 *moment placed in between* T1 *and T*2.

This property is not evident from the pragmatic standpoint. There are many temporal models in which time is discontinuous. In quantum microphysics is postulated the unity of discontinuity and continuity. That is, the time is neither a continuous flow, nor its some phases and points. This duplicity appears in some temporal models.

E: *Infinity property*
It means that *time may be unlimitedly continued both in future and in past*.

F: *Homogeneous property*
This property means *that any moment or interval of time is not changed if it is moved along a temporal scale*.

For example, in the dynamic system theory of Newton mechanics it is possible to exclude the time by introduction of phase portraits from generalized coordinates. In this case a time has the " pure" homogeneous property.
Nevertheless there are temporal models (of a psychological time) in which this property is violated. For example, when we awake, the time of sleeping is perceived as one short period.

Thus we can see that none of the considered above properties of a time are universal. Therefore it's possible to construct different temporal models which are discerned by axioms mapping one or another properties of a time.

Finally, consider the classification of existing now logical models of time as following:
- temporal logics based on the classical predicate calculation of the first order ;
- modal logics which are an expansion of the predicate calculation by introducing of modal operators of a time ;
- special calculations including nonclassical logical models such as fuzzy temporal logics, pseudo-physical logics, nonmonotonic and default logics and etc.

2. *Brief History*

Prior (Prior 1955;1957) [1] was first who uses *modal logic* for description of a temporal factor. He introduced the *temporal operators* with following semantics:

F_a - a will true at some time in a future ; P_a - a was true at some time in a past;

G_a - a will true always in a future; H_a - a was true always in a past.

For description of dated events the modified operators have been used :

$P(t,a)$ - event a occurred at t unit of time before the present time .

$F(t,a)$ - event a will occur at t unit of time after the present time .

In order to make some inferences about metrical temporal relations, Prior has constructed the axioms system for operators P and F . Consider such axioms for P operator (for F operator it's the same):

$$P(t,\neg a) \Rightarrow \neg P(t,a),$$
$$P(t,a \Rightarrow b) \Rightarrow (P(t,a) \Rightarrow P(t,b)),$$
$$P(0,a) \Rightarrow a,$$
$$P(t_1, P(t_2,a)) \Rightarrow P(t_1+t_2,a),$$
$$P(t_1, \exists(t_2)P(t_2,a)) \Rightarrow \exists(t_2)P(t_1,a)).$$

In the Prior model the following five rules are used as the base for the inference:

1) If F is a deducible *WFF* and q is any term in F and if we change q on p in this F, then a new *WFF* will be also deducible.
2) If a and $a \Rightarrow b$ are deducible, then b is deducible (*modus ponens* rule).
3) If $a(t)$ is deducible for any t , then $a(t^*)$ also deducible, where t^* is any fixed moment of time;
4) If a is deducible, then $P(t,a)$ is also deducible;
5) If $a(t^*)$ is deducible, *then* $\exists t a(t)$ also deducible.

Remark 1. *Prior logic deals with only instant events*, and therefore it can not been used for description of interval events and processes. The models of interval logic are discussed below.
Remark 2. Modal logics are developed for describing such notions as *necessity* and *possibility*. Modal logics are often named as *logics of grammatical time* (or *tense logics*). In a natural language sentence the so named *point of temporal proposition* (PTP) is used relative to which the time of event (that had been talking in the sentence) is described. *Tense logics* with some modifications may be used for natural language understanding tasks.

Consider the simplest example of a temporal logic design - *logic of Wright* (*Wright* , 1965)) [2] based on *the propositional calculus* where two temporal predicates are introduced :

R_1 - "to be earlier"; and R_2 - "to be later".

(R_1 and R_2 are interpreted as nonstrict order relations on a temporal scale.)
Construction of this logic includes three steps.
First step.
Introduction of rules of the *well formed formulas* (*WFF*) generation.
(1) If *A* is any term, then (*A*) - *WFF*.
(2) If *A* and *B* are any *WFF*, then

$(\neg A), (A \vee B), (A \vee B), (A \Rightarrow B), (A = B)$ - *WFF*.

(3) $(AR_1B), (AR_2B) - WWF.$

(4) There are no other *WFF*.
Second step.
Introduction the axioms system in the logic. Connect the following temporal axioms to the propositional calculus axioms:

(Copia modificata per una migliore consultazione on-line)

$(A \vee B)R(C \vee D) = (ARC) \vee (ARD) \vee (BRC) \vee (BRD)$,

$(ARB) \wedge (CRD) = (A \wedge C)R(B \wedge D) \vee (BRD) \vee (DRB)$,

$(AR(B \vee \neg B))$, where $R = \{R_1, R_2\}$.

Third step.

Definition inference rules used in this logic. Following inference rules are used in this logic.

(1) If F is a deducible *WFF* and b is any term in F and if we change b on c in this F, then a new *WFF* will be also deducible.

(2) If A and $A \Rightarrow B$ are deducible, then B is deducible (*modus ponens*).

(3) If A and $B = C$ are deducible and if we change B on C in the A, then new formula is deducible formula.

By using this temporal model we can simulate simple temporal reasoning about events in a time.

The design of temporal models based on the predicate calculus of the first order consists of including a time variable into the predicate. Then formulas of the predicate calculus become, for example, as follows:

$\forall t$ *P* (Moscow, *t*),

where *P* - is the predicate "it's rain" with two arguments : a place - Moscow, a time - *t*.

Russel (Russel, 1967) [3] proposed to use a temporal factor also for other variables, for example, as follows: $\exists t$ *P* (Moscow (t), t). This formula interpreted as follows: there is time moment t when in Moscow being at moment t is rain.

Selecting certain time properties and setting temporal axioms we may construct a temporal model based on the predicate calculus of the first order.

Remark 3. Logics of Wright and Russel illustrated the simplest way of a temporal model's design. But, they don't allow to describe a complex dynamic situation because there is no possibility to consider temporal data and intervals of time. To avoid these disadvantages a special temporal models were developed.

The interesting representative of temporal models based on the predicate calculus is the *situational calculus* of McCarty and Hayes (McCarthy & Hayes, 1969) [4]. It was one of first proposals to formalize *reasoning about time* and *action* constructed as extension of the classical predicate logic.

A *situation variable* (or a *world state*) is considered instead of a time variable. The change of world states are based on an action and described by function mapping one state to another. For example, the action " to *paint*" is described by the following mapping function:

color(house1,blue,s) → *color (house1,red, s1)*,

where *s* is the old situation, *s1* is a new situation and defined as

s1 = paint (house1,blue,s)).

Remark 4. This model has also a number of deficiencies: changes are discrete; there are no tools for expression of simultaneous actions; the use of a situation as a basic temporal primitive is not efficient.

Further improvement of models based on the predicate calculus of the first order results in development more complicated models.

First logical models give an impulse to development of more perfect temporal logics applicable for a complex dynamic world's description.

3. Advanced temporal models

3.1 *Allen's temporal model*

Amongst the works that has been most influential in the temporal models design area is *Allen's temporal model* (Allen, 1983;1984) [5].

The *basic primitive* in his model is a *time interval*, and 13 primitive relations define all possible mutual locations of two intervals on the time axis (Fig.14-1).

Relation	*Meaning*
x before y *y after x*	*x*___ *y*_____
x meets y *x met-by y*	*x*____ *y*____
x overlaps y *y overlapped-by x*	*x*____ *y*____
x during y *y contains x*	*x*__ *y*_____
x starts y *y started-by x*	*x*____ *y*_____
x finishes y *y finished-by x*	*x*____ *y*_____
x equals y	*x*____ *y*____

Figure 14 –1. Temporal relations between intervals.

Each time interval *T* is connected with some event (or process, action) by means of the following predicate

Occur (e, T),

which says: event *e* (action or process) occurs during interval *T*.

Each time interval *T* is connected with a property by means of predicate

$Holds(p,T)$,

which means: property *p* holds during interval *T*.

Allen introduced axioms describing properties and events in a time.

For example,

1) $Holds(p,T) \Leftrightarrow (\forall t)(IN(t,T) \Rightarrow Holds(p,t))$,

where *IN (t,T)* means that the interval *t* is a proper subinterval of *T*.

2) $Holds(not(p),T) \Leftrightarrow (\forall t)(IN(t,T) \Rightarrow \neg Holds(p,t))$.

This axiom characterizes the negation of property. Here we introduced the operator "*not*" as the property negation in order to be kept distinct from ordinary logical negation operator " ¬ ".

3) $Occurs(e,S) \,\&\, IN(S,T) \Leftrightarrow Occurs(e,T)$, *if e is durative*

$Occurs(e,T) \Leftrightarrow (\exists t)(IN(t,T) \,\&\, Occurs(e,t))$

(axioms for events).

Remark 5. The merit of Allen's model results in introducing a rich set of temporal relations and axiomatics based on these relations. The model was used for Natural Language understanding and action planing tasks. But there is also some of his model as follows: 1) this temporal model is not adequate for reasoning about continuos change (see, for example, (Galton,1990) [6]; 2) there are no tools for fuzzy temporal information processing. Nevertheless this model gave the impulse to development a lot of logical models based on it.
Consider now a more complicated temporal model for temporal knowledge representation and manipulation.

3.2 Pseudo-physical Temporal Logic

Pseudo-physical logics was introduced in the monograph by several authors in [7]. The notion of *pseudo-physical logic* reflects the fact that:
neither real physical and metrical properties of a time and a space nor human space-time perception properties are used in axioms and inference rules.
The *Pseudo-Physical Logic of Time* (*PLT*) allows to simulate human reasoning about a time. *PLT* is characterized by a lot of properties:

- this logic is based on temporal relations ;
- reasoning in *PLT* are connected with time scales and properties of temporal relations;
- there is a few components connected with each other.

PLT is designed as a formal theory based on the predicate logic of the first order. It consists of the following 4-tuple: $\{A, G, R, I\}$, where *A* (alphabet) is a set of basic elements, *G* (syntax) is a set of rules for *WFF* definition, *R* is a set of axioms, and *I* is a set of inference rules.

Alphabet
Alphabet *A* includes terms (variables and constants), predicates, functional symbols, logical operations symbols and service symbols.
Terms
In order to formalize temporal knowledge let us include events, times, temporal relations and temporal scales as objects in the universe of discourse. Then we may use names of events, times, relations and scales. So, the set of terms consists of subsets of different sort (or types) terms. We introduce special designations for each sort of terms.
We will consider the following sorts.

- a finite set of events names $P = \{p, p_1, p_2, ..., p_i\}$;
- a set of temporal points $\Theta = \{t_1, t_2, ..., t_i, ..., t_k, ...\}$, where each t_k is a point in a given temporal scale with a given measure unit;
- a set of intervals $\Delta = \{T_1, ..., T_j, ...\}$, where each T_j is an interval in a given temporal scale with a given measure unit;
- a set of temporal scales names $\Xi = \{L, L_1, L_2, ...L_n\}$;
- a set of real numbers $N = \{n, n_1, ..., n_s, ...\}$.

Two types of events are considered in the *PLT*: *instant* and *interval*.
The event type is characterized by that how the event is projected onto time scale. If a duration of event p is so that it is projected onto a *point* in a time scale, this event is an *instant*. Otherwise, the event is an *interval*. Describe this projection by functional operator $t(p)$. Then for instant events $t(p) = t^*$, where t^* is a point in a temporal scale. t^* has some numeric value defined in some temporal scale. For interval events $t(p) = T^*$, where T^* is an interval in a temporal scale. T^* has a duration defined by a numeric value as $T^* = t^{end}(p) - t^{begin}(p)$, where $t^{end}(p)$ is an ending point of event *p* and has some numeric value defined in some temporal scale, and $t^{begin}(p)$ is a beginning point of event *p* and has some numeric value defined in the same temporal scale.
So, in *PLT* model the following basic types of terms are introduced:
[*event, temporal point, temporal interval, number, measure unit, temporal scale*].
A set of *measure units* consists of a finite list of measure unit names, for example, as follows:

$$\{year, month, day, hour, minute, second\}.$$

Consider now temporal predicates defined as temporal relations by the following way.

Predicates and functional elements

The set of predicates consists of two sets: R_{ins} *representing all temporal relations for instant events, and* R_{int} *representing all temporal relations for interval events.*
Functional elements represent simple arithmetic operations over numbers.

I. Temporal relations for instant events

I.1 Nonmetrical order relations:

$pr_0 q$ - event *p* occurs simultaneously with event *q* (we will interpret it as "=" - relation between points $t(p) = t(q)$);

$pr_1 q$ - event *p* occurs earlier event *q* (we will interpret it as "<" - relation between points ($t(p) < t(q)$);

$pr_2 q$ - event *p* occurs later event *q* (we will interpret it as ">" - relation between points ($t(p) > t(q)$).

I.2 Metrical order relations:

$pr_{occur} t_k$ - event *p* occurs at moment t_k, where t_k is a point in a given temporal scale with a given measure unit ;

$pr_{M1}(n, \omega) q$ - event *p* occurs earlier event *q* on n units on the scale *L*, where ω is the measure unit of the scale *L*, (*n* = 1,2,…).

(For example, $p_1\ r_{M1}(2, "hour") q_1$ means that event p_1 realized two hours earlier event q_1.)

For simplicity, we will write also this relation as $pr_{M1}(n)q$. (In this case time scale *L* with a measure unit is defined by default or from a current context).

I.3 Periodic relations:

- $pr_P(\tau)$ - event *p* occurs τ (number) times on scale *L*.

II. Temporal relations for interval events

II.1 Nonmetrical order relations

Nonmetrical order relations describe different ways of relations between two temporal intervals (see Fig.14-1).

pR_1q - event *p* occurs *strictly earlier* event *q*;

pR_2q - event *p* occurs *strictly later* event *q*;

pR_3q - event *p* is *overlapped* with event *q*.

Different cases of temporal overlapping of events are as follows:

pR_0q - event *p* occurs *simultaneously* with event *q*;

pR_4q - event *p* occurs in a time so that its duration is the subinterval of the duration of event *q* (this means that there is *during*-relation between intervals (see Fig. 14-1));

pR_5q - event *p* occurs in a time so that its duration is the subinterval of the duration of event *q* and beginning of *p* equal beginning of *q* (this means that there is *starts*-relation between intervals);

pR_6q - event *p* occurs in a time so that its duration is the subinterval of the duration of event *q* and finishing of *p* equal finishing of *q* (*finishes*-relation between intervals) ;

pR_7q - event *p* occurs in a time so that finishing of *p* equal beginning of *q* (*meets*-relation between intervals) .

Remark 6. This set of relations is similar toAllen's set of relations.

II.2 Metrical order relations :

$pR_{begin}t_k$ - event *p* begins at moment t_k , where t_k is a point in a given temporal scale with a given measure unit ;

$pR_{end}t_k$ - event *p* ends at moment t_k , where t_k is a point in a given temporal scale with a given measure unit ;

$pR_{M1}(n,\omega)q$ - event *p* occurs strictly earlier event *q* on n units on scale *L* where ω is the measure unit of scale *L* , (*n* = 1,2,…);

$pR_{M2}(n,\omega)q$ - event *p* is overlapped with event *q* so that the distance between their beginning points equals n units on the scale *L*, where ω is the measure unit of the scale *L* , (*n* = 1,2,…);

$pR_{M3}(n,\omega)q$ - event *p* is overlapped with event *q* so that the distance between their finishing points equals n units on the scale *L*, where ω is the measure unit of the scale *L*, (*n* = 1,2,…).

II.3 Periodic relations:

$pR_P(\tau)$ - event *p* occurs τ times on the scale *L*.

Syntax of PLT Model

Introduce syntactical rules for well formed formulas (*WFF).*

1. Any term is *WFF*.
2. Any relation in the form I.1, I.2, I.3, II.1, II.2, II.3 is *WFF*.
3. If α and β are *WFF*, then $(\alpha R\beta)$ are also *WFF*, $R \in R_{ins} \bigcup R_{\text{int}}$.
4. If α and β are *WFF*, then $(\alpha \wedge \beta)$, $(\alpha \vee \beta)$, $\neg\alpha$ are also *WFF*.
5. Expressions like $t(p), t(q)$, $(t(p) * t(q))$, where * is one of the $\{<, >, =, \leq, \geq\}$ and $(t(p) \pm n)$ are *WFF* .
6. If α is *WFF*, then $(\forall x)\alpha, (\exists x)\alpha$, where $x \in \Delta$ or $x \in \Theta$, are also *WFF*.
7. There are no another *WFF*.

In *PLT* Model operations $(\alpha \wedge \beta)$, $(\alpha \vee \beta)$, $\neg\alpha$ are interpreted in the set-theoretical sense, that is, as follows:

- simultaneous presence of facts α, β in the knowledge base for the operation $(\alpha \wedge \beta)$,
- presence of one of facts α, β (or two) in the knowledge base for the operation $(\alpha \vee \beta)$; and
- absence of the fact α in the knowledge base for operation $\neg\alpha$.

To simulate a human temporal reasoning we must develop a set of axioms describing properties of temporal relations.

Temporal axioms

The kernel of *PLT* consists of the set of axioms describing different properties (algebraic, semantic and human time perception) of basic temporal relations.

Temporal axioms are given in the form of general schemes as follows.

(Copia modificata per una migliore consultazione on-line)

I. Axiom Schemes for instant temporal relations

I.1 *Schemes for instant nonmetrical temporal relations.*

1. $pr_0q \Leftrightarrow t(p) = t(q)$;
2. $pr_1q \Leftrightarrow t(p) < t(q)$;
3. $pr_2q \Leftrightarrow t(p) > t(q)$;
4. $p_i r_0 p_j \Leftrightarrow p_j r_0 p_i$ (symmetry property);
5. $p_i r\ p_j, p_j r\ p_k \Rightarrow p_i r p_k, r \in \{r_0, r_1, r_2\}$ (transitivity property);
6. $p_i r_1 p_j \Leftrightarrow p_j r_2 p_i$;
7. $p_i r_0 p_j, p_j r\ p_k \Rightarrow p_i r p_k, r \in \{r_1, r_2\}$.

Remark 7. Axiom schemes are represented as two types rules: 1) $\alpha, \beta \Rightarrow \gamma$, and 2) $\alpha, \beta \Leftrightarrow \gamma$, where α, β, γ are *WFF*. The rules are interpreted as follows: the presence of facts α, β implies the presence of fact γ (for the first type of rules); or the presence of facts α, β implies the presence of fact γ and inversely (for the second type of rules).

I.2 *Schemes for instant metrical temporal relations.*

1. $pr_{M1}(n)q \Leftrightarrow (t(q) = t(p) + n)$;
2. $pr_{M1}(n)q \Rightarrow pr_1q$;
3. $p_i r_{M1}(n) p_j, p_j r_{M1}(m) p_k \Rightarrow p_i r_{M1}(n+m) p_k$;
4. $$p_i r_{M1}(n) p_j, p_i r_{M1}(m) p_k \Rightarrow \begin{cases} p_k r_{M1}(n-m) p_j, m < n \\ p_j r_{M1}(n-m) p_k, m > n \\ p_k r_0 p_j, m = n \end{cases}$$

We assume here that events p_i, p_j, p_k are projected on the same temporal scale.

II. Axiom Schemes for interval temporal relations

II.1 *Schemes for interval nonmetrical temporal relations*

1. $pR_0q \Leftrightarrow (t^{begin}(p) = t^{begin}(q))$;
2. $pR_0q \Leftrightarrow (t^{end}(p) = t^{end}(q))$;
3. $pR_1q \Leftrightarrow (t^{begin}(p) < t^{begin}(q)) \,\&\, (t^{end}(p) < t^{end}(q)) \,\&\, (t^{end}(p) < t^{begin}(q))$
4. $p_i R_1 p_j \Leftrightarrow p_j R_2 p_i$;
5. $p_i R_0 p_j \Leftrightarrow p_j R_0 p_i$ (symmetry property);
6. $p_i R p_j, p_j R p_k \Rightarrow p_i R p_k, r \in \{R_0, R_1, R_2\}$ (transitivity property);
7. $p_i R\ p_j \Rightarrow p_j R_3 p_i, R \in \{R_0, R_4, R_5, R_6, R_7\}$;
8. $p_i R\ p_j \Rightarrow p_i R_4 p_i, R \in \{R_5, R_6\}$;
9. $p_i R_0 p_j, p_j R\ p_k \Rightarrow p_i R p_k, R \in \{R_1, R_2, ...R_7\}$;
10. $p_i R p_j \Rightarrow p_j R_3 p_i, R \in \{R_4, R_5, R_6, R_7\}$;
11. $p_i R_1 p_j, p_k R\ p_j \Rightarrow p_i R_1 p_k, R \in \{R_4, R_5, R_6\}$;
12. $p_i R_2 p_j, p_k R\ p_j \Rightarrow p_i R_2 p_k, R \in \{R_4, R_5, R_6\}$;
13. $p_i R_4 p_j, p_j R_4 p_k \Rightarrow p_i R_4 p_k$;
14. $p_i R_4 p_j, p_i R\ p_k \Rightarrow p_j R_3 p_k, R \in \{R_5, R_6, R_7\}$;
15. $p_i R_4 p_j, p_k R\ p_i \Rightarrow p_k R_4 p_j, R \in \{R_0, R_4, R_5, R_6, R_7\}$;
16. $p_i R\ p_j, p_j R p_k \Rightarrow p_i R\ p_k, R = \{R_4, R_5\}$;
17. $p_i R\ p_j, p_i R p_k \Rightarrow p_j R\ p_k, R = \{R_4, R_5\}$;

18. $p_i R_7 p_j, p_j R_7 p_k \Rightarrow p_i R_1 p_k$;
19. $p_i R\ p_j, p_j R_7 p_k \Rightarrow p_i R\ p_k, R = \{R_1, R_2,\}$;
20. $p_i R\ p_j, p_j R_7 p_k \Rightarrow p_i R_1 p_k, R = \{R_4, R_5,\}$;
21. $p_i R_6 p_j, p_j R_7 p_k \Rightarrow p_i R_7 p_k$.

These schemes are obtained from semantics of different combinations of relations and events pairs.

II.2 *Schemes for interval metrical temporal relations.*

1. $pR_{M1}(n)q \Leftrightarrow t^{begin}(q) = t^{end}(p) + n$;
2. $pR_{M1}(n)q \Rightarrow pR_1 q$;
3. $p_i R_{M1}(n) p_j, p_j R_{M1}(m) p_k \Rightarrow p_i R_{M1}(n + m + t(p_j)) p_k$;
4. $pR_{M2}(n)q \Leftrightarrow t^{begin}(q) = t^{begin}(p) + n$;
5. $p_i R_{M2}(n) p_j, p_k R_{M2}(m) p_i \Rightarrow p_k R_{M2}(n+m) p_j$;
6. $pR_{M3}(n)q \Leftrightarrow t^{end}(q) = t^{end}(p) + n$;
7. $p_i R_{M3}(n) p_j, p_j R_{M3}(m) p_k \Rightarrow p_i R_{M3}(n+m) p_k$;
8. $p_i R\ p_j, p_j R_0 p_k \Rightarrow p_i R\ p_k, R = \{R_{M1}, R_{M2}, R_{M3}\}$.

We assume here that events p_i, p_j, p_k are projected on the same temporal scale.

Temporal scales

To simulate human temporal reasoning (especially given in the form of natural language texts) we will interpret a temporal scale as one of the following types: *absolute* (*A*-scale), *relative* (*O*-scale) and *fuzzy* (*F*-scale).
A-scale is a discrete oriented metrical scale. Each notice of the scale is connected with the absolute date according with the universally adopted chronology. Different *A*-scales are considered in dependence with a unit measure (year, month, day, week, hour, minute, and so on).
In order to describe the absolute date of some event according with the universally adopted chronology we will use a chronological scale represented as a set of A-scales (Fig.14-2) with the following units measures: year, month, day, hour, minute, second.
Consider, for example, the following event expressed in the natural language sentence
"24 January 2001 at 11 o'clock a.m. AI examination will be". The event "examination" can be projected on a set of A-scales as shown in Fig. 14-2.

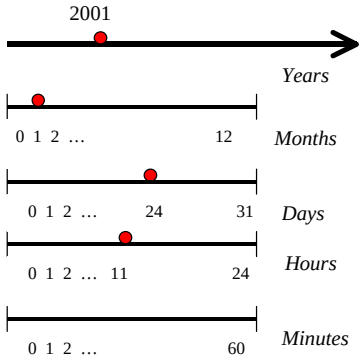


Figure 14-2. Event projection on a chronological scale.

O-scale is an *oriented ordered scale*. There are 2 types of *O*-scales: with metrics (O_M) and without metrics (*O*).

At the O_M -scale a *point of temporal proposition (PTP)* is fixed. *PTP* marks the time moment of proposition (sentence) uttering, and the event having been expressing in proposition is projected onto the *R*-scale with respect to *PTP*.

For example, "Mister X come to Japan two days ago". The described event is projected onto the metric O_M -scale according to the *PTP*. If the time of the *PTP* is known, then the O_M -scale can be mapped onto the *A*-scale.

At the *O* –scale a temporal order of events is shown only. For example, "Mister X came to Japan when he has finished to write her book". Described events are projected onto the *O*-scale which show a temporal order of given events, that is, event "write book" is before event "staying in Japan".
F-scale is a *nonmetrical fuzzy temporal scale.* The presence of such scale is connected with existence in the natural language fuzzy propositions like of " *Rapidly* it's rain" or "Yesterday I was working *so long*".
For projecting on the *F*-scale it is needed to use a fuzzy logic technique based on the fuzzy sets theory.

3.3 *Fuzzy temporal models*

In order to represent and process temporal expressions including underdetermined and incomplete temporal data *fuzzy temporal logics* are developed.

(Copia modificata per una migliore consultazione on-line)

The main temporal primitives in these models are fuzzy temporal relations between events and fuzzy intervals that are described by means of membership functions. Following examples represent expressions with fuzzy temporal relations: “event *p* took place *about* 10 minutes *before* event *f* “; “event *p* lasted *approximately* 20 minutes; “event *p* took place *approximately* in one hour after event *f* “.
Fuzzy temporal logics are widely used in fuzzy control systems (see, for example, [8,9].
The extension of PLT may be constructed by including a fuzzy component of temporal model. Let us introduce the following fuzzy temporal relations.

Fuzzy temporal relations for instant events

Nonmetric

$pr_o^f q$ - event *p* occurs *almost simultaneously* with event *q* ;

$pr_1^f q$ - event *p* occurs *substantially earlier* event *q*;

$pr_2^f q$ - event *p* occurs *not so substantially* earlier event *q*;

$pr_3^f q$ - event *p* occurs *a little earlier* event *q.*

In this case, for each relation, interval T defining the distance between two events in some temporal scales can be described by a given membership function.

Metric

$pr_{M0}^f(n,\omega)$ - event *p* occurs *at about n* unit on scale *L* , where ω is the measure unit of scale *L*;

$pr_{M1}^f(n,\omega)q$ - event *p* occurs *approximately* n *units earlier* on scale *L* event *q*, where ω is the measure unit of scale *L*.

In this case *n* is described as a fuzzy number by corresponding membership function.

Fuzzy temporal relations for interval events

Nonmetric:

$pR_o^f q$ - event *p* occurs *almost simultaneously* with event *q* ;

$pR_1^f q$ - event *p* occurs *substantially earlier* event *q* ;

$pR_2^f q$ - event *p* occurs *not so substantially earlier* event *q*;

$pR_3^f q$ - event *p* occurs *a little earlier* than event *q.*

You see here that a distance between finishing of one event and beginning of other is described by fuzzy intervals and corresponding membership functions (for example, a large distance in a time scale, a small or middle distance, etc.).

Metric:

$pR_{M0}^f(n,\omega)$ - event *p* is begun *at about n* on scale *L* , where ω is the measure unit of scale *L*;

$pR_{M1}^f(n,\omega)q$ - event *p* occurs *approximately* n *units earlier* on scale *L* event *q*, where ω is the measure unit of scale *L*. (In this case *n* is described as a fuzzy number by a corresponding membership function).

In the case of fuzzy temporal relations, points of beginning / finishing of events and event’s duration are described by means of fuzzy numbers. Representation of fuzzy numbers and operations over them has been considered in Part 2 of Lectures Notes.

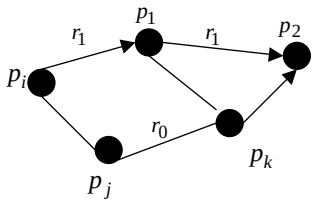
4. Temporal Data Representations

4.1 *Representing temporal relations through graphs*

Introduce some definitions.

Definition 1.

A *timegraph* (T-graph) is a graph with a set of nodes and a set of labeled arcs, where each arc (p,r,q) connects a pair of distinct nodes *p* and *q* by temporal relation *r*. Every node of the timegraph represents an event’s name, denoted as $p_i, i=0,...,n$ (Fig. 14-3).



(Copia modificata per una migliore consultazione on-line)

Figure 14-3. An example of T-graph.

Definition 2.

A simple timegraph (T-graph) is a graph where the arcs are either directed and labeled as r_1 (or "<"), r_2 (or " >") or undirected and labeled as r_0 (or "=").

Remark 6. We introduce a simple definition of T-graph with temporal relations $r \in \{<,>,=\}$ between temporal points or intervals. But we may also consider more complicated T-graphs.

Definition 3.

A *model* of T-graph is an interpretation of the node's names as elements of a totally

ordered set T with strict ordering <.

Definition 4.

A T-graph is *consistent* if and only if it has at least one model.

Definition 5.

Two or more T-graphs are *logically equivalent* if and only if they have the same models.

Definition 6.

A *path of length n* from p_0 to p_n is a sequence of *n* triples

$(p_0 r_1 p_1),...,(p_{n-1} r_n p_n)$, where $p_i, i = 0,...,n$ are nodes and r_j, $j = 1,...,n$ are relations on arcs, $r_j \in \{<,>,=\}$.

Definition 7.

A *path of length n* from p_0 to p_n is a cycle if $p_0 = p_n$.

Theorem 1.

A *T-graph is consistent if and only if it does not contain any " <"-cycle or" >"-cycle.*

Proof is based on the transitivity axiom described above (see axiom I.1(5)).

T-graph –based representation can be used for Temporal Data Bases design and temporal consistence checking (see paragraph 5.1 below).

4.2 Frame-based Representation of Temporal Data and Knowledge Bases

Consider the task of Temporal Data and Knowledge Base design by using frame formalism (see Lecture 2).
In this case we must introduce frame's prototypes *of events, temporal scales, relations, intervals and data* (examples see in Table 14-1). We will consider the set of prototypes as a part of *Temporal Knowledge Base*.
The set of frame's representatives of above prototypes represents *Temporal Data Base* (see examples in Table 14-2).

Table 14-1. Examples of frame's prototypes contained in the Knowledge Base.

(interval-event:
isa = *prototype*;
name = <symbolic-name>or <frame-representative>;
duration = interval or fuzzy interval ;
begin = tempoint or date;
end = tempoint or date;)
 	
(instant-event:
isa = *prototype*;
name = <symbolic-name>or <frame-representative>;
time = tempoint or date
 	
(tempoint:
isa = *prototype*;
measure-unit=
(year,month,day,minute,second)
value = <number> or <fuzzy number>)
	(date:
	isa = *prototype*;
	time-scale =
	<frame-representative>;
	year = <number>;
	month = <number>;
	day = <number>;
	minute = <number>;
	second =<number>)
	(fuzzy-interval:
	isa = *prototype*;
	name = *morning*;
	measure-unit = "hour";
	support =<[number;number]>;
	membership = <function>)
	
	(interval:
	isa = *prototype*;
	length = <number>;
	measure-unit = <symbol>;
	begin = date or < number>;
	end = date or <number>)
		(A-scale:
		isa = *prototype;*
		type = *"chronological"*;
		measure-unit= (year,month,day,minute,second))
		
		(O-scale:
		isa = *prototype;*
		type = *"relative-metric"*;
		PTP-point = <number> ;
		Measure-unit = <symbol>)
		
		(<temp-relation>:
		isa = *prototype*;
		name = "earlier";
		type = ("temporal-nonmetrical-relation",…,);
		argument1 = event;
		argument 2 = event;
		parameters = <list>;)

Remark 8. The Data and Knowledge Bases are designed by a so-called *knowledge engineer* according to a given problem domain and tasks. Structures of frames, frames slots and its values are defined by the knowledge engineer in accordance with the problem area semantics.
In the Table 14-1, in the left parts of frame-prototypes structures the *names of slots* are defined, and in right parts possible values of these slots are described. For example, the slot "begin" in the prototype "instant-event" may have one of the following values: it may be a representative of the "tempoint" prototype or may be a representative of the "date" prototype. Or, for example, the slot "duration" may have the value $008, which is the concrete representative of fuzzy interval named "morning".

Table 14-2. Examples of frame-representatives contained in the Data Base.

(Copia modificata per una migliore consultazione on-line)

($001:
isa = interval-event;
name = “examination”;
duration = $007 ;
begin = $003;
end =)
	($007:
	isa = interval;
	length = 2;)
	measure-unit = “hour”)
		($004:
		isa = time-scale;
		type = “*chronological”;*
		measure-unit = (year, month, day, hour) ;)

($002:
isa = instant-event;
name = “breakfast”;
time = $005;)
	($005:
	isa = tempoint;
	time-scale = $006;
	value = 8)
		($006:
		isa = time-scale;
		type = “relative”;
		measure-unit = “hour”)

($003:
isa = date;
time-scale = $004;
year = 2000;
month = 6;
day = 7,
hour = 5)
	($008:
	isa = fuzzy-interval;
	name = *morning*;
	measure-unit = “hour”;
	support =<[7;10]>;
	membership = <f1>)
		($009:
		isa = interval-event;
		name = “meeting”;
		duration = $008 ;)

Remark 9. In frame prototype representatives, slots with unknown values may be omitted, as, for example, in $009.

5. Examples of Temporal Models Application

5.1 Managing large sets of qualitative temporal information.

In the case of large sets of qualitative temporal information we need efficient algorithms for validation and temporal reasoning. One of them is based on T-graphs mentioned above [10]. In this application T-graphs are used for checking temporal consistency and for answering on different queries to the T-graph.

5.2 Synchronization for interactive multimedia presentations

Consider the task of a multimedia presentation. The multimedia presentation can consists of several media objects (texts, images, video, audio) with different temporal characteristics. All these objects are contained in different databases, and transmitted via different network channels.

Define the following task of temporal control for this multimedia representation.

The task is to control that temporally related media units from different databases will play synchronously at the client site.

For this task a formal description of multimedia presentation and synchronization control are needed. For this purpose a temporal model based on interval events and relations is used.

Table 14-3. Semantics of temporal relationships.

Relationsh ip	*Semantics*
a	$t_{begin}(a) \le t_{end}(a)$
a *equals* b	$t(a) = t(b)$
a *starts* b	$t_{begin}(a) = t_{begin}(b); t_{end}(a) < t_{begin}(b)$ $\left[\Delta^{starts}\ t = t_{end}(b) - t_{end}(a)\right]$
a *before* b	$t_{begin}(a) < t_{begin}(b); t_{end}(a) < t_{end}(b)$ $\left[\Delta^{before}\ t = t_{begin}(b) - t_{end}(a)\right]$
a *meets* b	$t_{begin}(a) < t_{begin}(b); t_{end}(a) = t_{begin}(b)$ $\Delta^{meets}\ t = 0$
a *during* b	$t_{begin}(a) > t_{begin}(b); t_{end}(a) < t_{end}(b)$ $\left[\Delta_0^{during}\ t = t_{begin}(b) - t_{begin}(a); \Delta_1^{during}\ t = t_{end}(b) - t_{end}(a)\right]$
a *overlaps* b	$t_{begin}(a) < t_{begin}(b); t_{end}(a) < t_{end}(b)$ $\left[\Delta_0^{overlaps}\ t = t_{begin}(b) - t_{begin}(a); \Delta_1^{during}\ t = t_{end}(b) - t_{end}(a)\right]$
a *finishes* b	$t_{begin}(a) > t_{begin}(b); t_{end}(a) = t_{end}(b)$ $\left[\Delta^{fin}\ t = t_{begin}(b) - t_{begin}(a)\right]$

A presentation stage is a semantic cut of a multimedia presentation. For example, in a distant-learning system, a user might request a presentation introducing African animals [11].
The presentation consists of three stages:

1. A video clip shows a tiger eating a rabbit. During this period, audio information – birds singing and bugs humming – plays in the background with slides and text.
2. When the video clip is over, a narrator explains tiger’s behavior with the aid of another slide show and some text.
3. Later, narration of the video clip continues with the aid of more slides and text.

Figure 14-2 shows the presentation’s temporal schedule in the form of a display-time bar chart.

How to describe formally this presentation? Let us introduce the following notations:

(Copia modificata per una migliore consultazione on-line)

We will denote interval events by small Latin symbols as *a,b,c* and so on. A duration of a temporal interval of event *a* is denoted as $t(a)$, a beginning point of the temporal interval – as $t_{begin}(a)$ and an ending point – as $t_{end}(a)$. Then semantics of temporal relationships may be described as shown in Table 14-3.

The formal description of the illustrated in Fig.14-4 multimedia presentation (MP) is the following:

MP = (MS , TS)
MS = (video, audio, slide, text)
TS = (Stage-1, Stage-2, Stage-3) = (Section-1, Section-2, Section-3, Section-4, Section-5)
Stage-1 = (Section-1, Section-2)
Stage-2 = (Section-3)
Stage-3 = (Section-4, Section-5),
TR (Stage-1) =

(Video1 $[t_0, t_5]$ *is-finished-by* Audio1 $[t_3, t_5]$ and Slide2 $[t_3, t_5]$ and Text2 $[t_4, t_5]$),

(Video1 $[t_0, t_5]$ *starts with* Slide1 $[t_0, t_1]$ and with Text1 $[t_0, t_2]$),

TR (Stage-2) =

(Audio1 $[t_3, t_5]$ *meets* Audio2 $[t_5, t_8]$) (Audio2 $[t_5, t_8]$ *overlaps* text3 $[t_6, t_9]$),

(Slide3 $[t_5, t_7]$ *starts* Audio2 $[t_5, t_8]$) (Slide2 $[t_3, t_5]$ *meets* Slide3 $[t_5, t_7]$),

TR (Stage-2) = (Audio3 $[t_9, t_{10}]$ *starts* Video2 $[t_9, t_{12}]$),

(Audio4 $[t_{11}, t_{12}]$ *finishes* Video2 $[t_9, t_{12}]$) (Audio3 $[t_9, t_{10}]$ *meets* Slide4 $[t_{10}, t_{13}]$)

(Audio4 $[t_9, t_{10}]$ *overlaps* Slide4 $[t_{10}, t_{13}]$), (Audio3 $[t_9, t_{10}]$ *starts* Text4 $[t_9, t_{11}]$)

(Audio4 $[t_9, t_{10}]$ *meets* Text5 $[t_{12}, t_{13}]$) (Text4 $[t_9, t_{11}]$) *meets* Audio4 $[t_9, t_{10}]$)

(Video2 $[t_9, t_{12}]$ *meets* Text5 $[t_{12}, t_{13}]$).

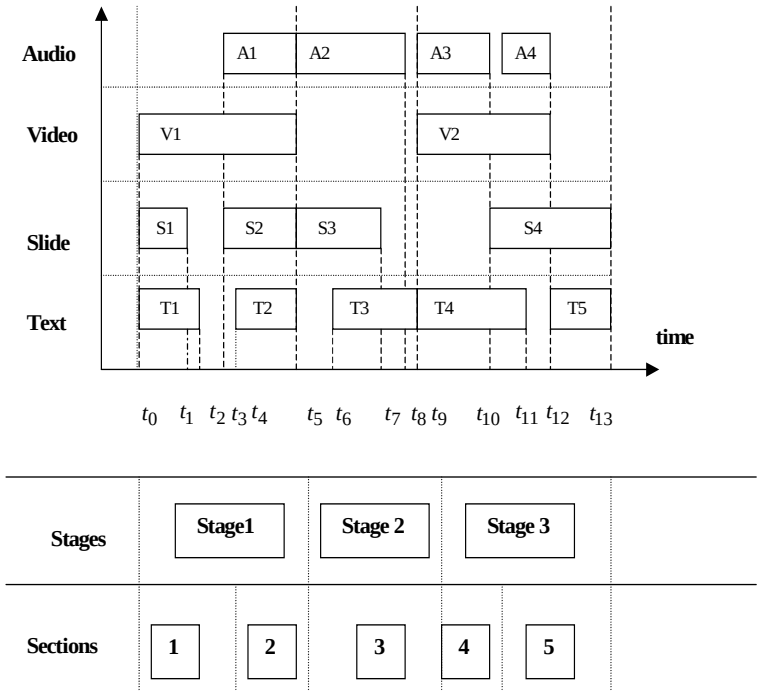


Figure 14-4. The display-time bar chart of a presentation's temporal schedule.

Lecture N 15 Spatial Knowledge Representation in Intelligent Systems

In this lecture we will talk about spatial models design problem. This problem deals with the *spatial knowledge representation* and *spatial reasoning*.

Nature of human spatial reasoning

Qualitative spatial reasoning about locations of different physical entities in a space plays very important role in our daily life.

Human beings make their judgments on the base of territory analysis including recognition and analysis of "*significant features*, called as *landmarks*, in the environment. Research in cognitive science has clearly demonstrated that human beings and animals record distinctive visual landmarks and use the structure and spatial relations between landmarks to guide navigation.

Navigation in an environment is performed by observing a position changing relative to the various landmarks in the environment.

The actual definition of a landmark is domain dependent. For example, it may be different objects in environment, and some or all may be visible in any given scene.

(Copia modificata per una migliore consultazione on-line)

So, human spatial reasoning refers in general to reasoning about problems dealing with entities occupying space.
These entities can be either *physical entities* (for example, books, chairs, cars, etc.) or *abstract entities* (like enemy territory).
Physical entities are tangible and occupy physical space while *abstract entities* are intangible but nevertheless can be associated with a certain space in some coordinate system.
Before spatial models considering discuss what properties are characteristic for a space.

Space properties in spatial models for intelligent systems

We shall differentiate the properties of a "pure" space and a space of human life environment. The following properties are characteristics for the "pure" space perceived by humans.

A: *No Direction At All*
Contrary to a time, a space itself has no direction at all. Humans create an artificial coordinate system so as to have an orientation in it. Space doesn't change after the introduction of a coordinate system. That is why one can choose any coordinate system.

B: *Continuousness*
We shall understand a continuousness of a space in the following way. If we define a *local bounded piece of space* as *LOC*, then for any two *LOC*'s that are close to each other we can find another *LOC* that is located in between them.

C: *Endlessness*
It is presumed that a space exists not only in our earth but in other worlds also.
D: *Homogeneity*
This property means that any two parts of the space have same properties.

The properties of "pure" space can be different from properties of a space, which is occupied by some objects and living beings in their living environment. For example, living environment is perceived by a human being as a space that has a direction and as an interrupted, a finite, and a nonhomogenous space. There are different metrical systems that allow to evaluate sizes of LOC's, landmarks, objects and the distances between them.
Considered above properties of a space are not universal. Therefore it's possible to construct different spatial models which are discerned by axioms mapping one or another properties of the space.
Discuss some spatial models for the representation of spatial knowledge and simulation human spatial reasoning in different tasks.

1.Brief History of Spatial Models Design

1.1 Navigation in two dimensional plane and qualitative spatial reasoning

Consider the problem of qualitative spatial knowledge representation and reasoning for the navigation in a physical space such as a two dimensional (2d) plane. Simple spatial model developed by Freksa (Freksa,1992) [12] is based on the representation of information about the direction and orientation in a space.
For representing qualitative orientation information, let us introduce the notion of the *orientation grid*.
From a cognitive point of view we are conceptualize people, animals, robots, etc. as having an "intrinsic front side". This results in an implicit dichotomy between a front region and a back region and a forward and backward orientation.
Orientation determined by two points in a space, a *start point* and an *end point* of a movement, for example *a* and *b*.

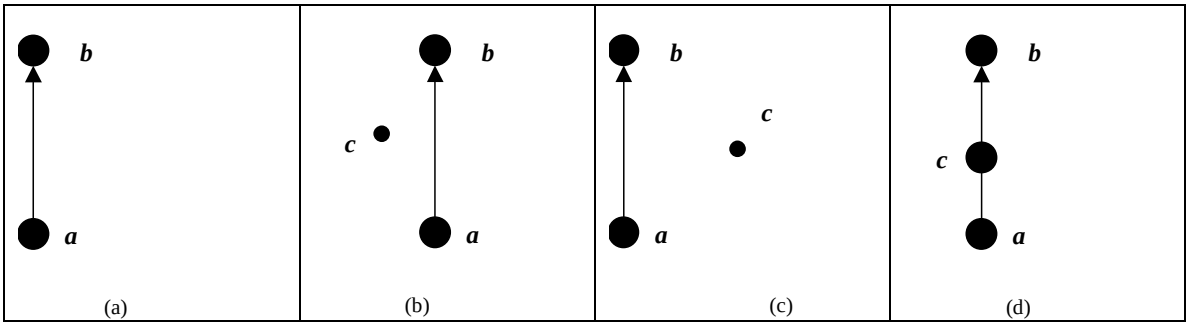


Figure 15–1. Oriented path vector (a) and position relations representation.

We introduce the vector *ab* denoting the *oriented path from a to b* (Fig. 15–1 (a)).
Consider now the position of additional point *c* with respect to the line of vector *ab*.
It may be three different possible positions: *left* of the line, *right* of the line, and *on* the line (Fig.15-1 (b,c,d)). These *position relations* are called: *left*, *right*, and *straight*. We will denote them as follows (*c left*: *ab*), (*c right*: *ab*), (*c straight*: *ab*), or (*c* : *ab*). (The last relation means one of a set of possible position relations).
Consider now the orientation of two points *c* and *d* so that the vector *cd* intersects the point *b* and orthogonal to the vector *ab*. A lot of this kind of vectors forms an orientation grid (Fig.15-2 (a)).

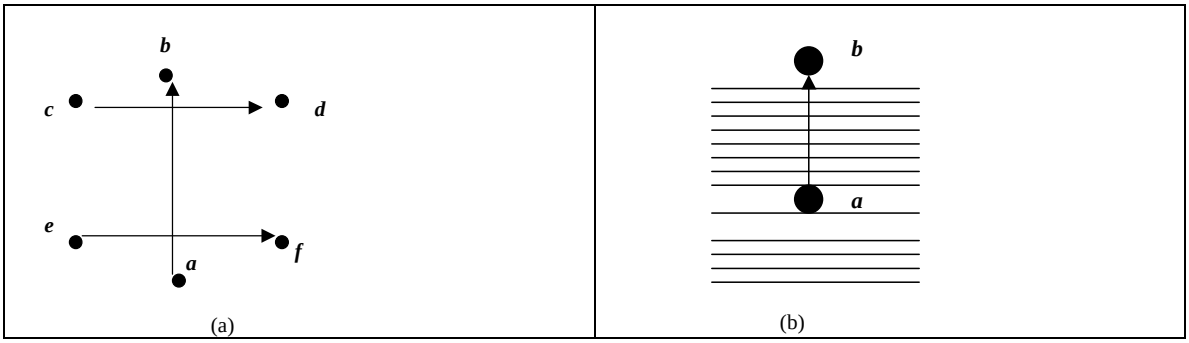


Figure –2. Orientation grid representation

So, the *orientation grid* is a segmentation of a plane into a *front/back*, and *left/right* planes relative to the given vector of navigation *ab* (Fig.15-2,(b)). It may be also two positions – *in front* and *back* – relative to the given navigation vector. We will denote them as (*c front*: *ab*), (*c back*: *ab*) (Fig.15-3).

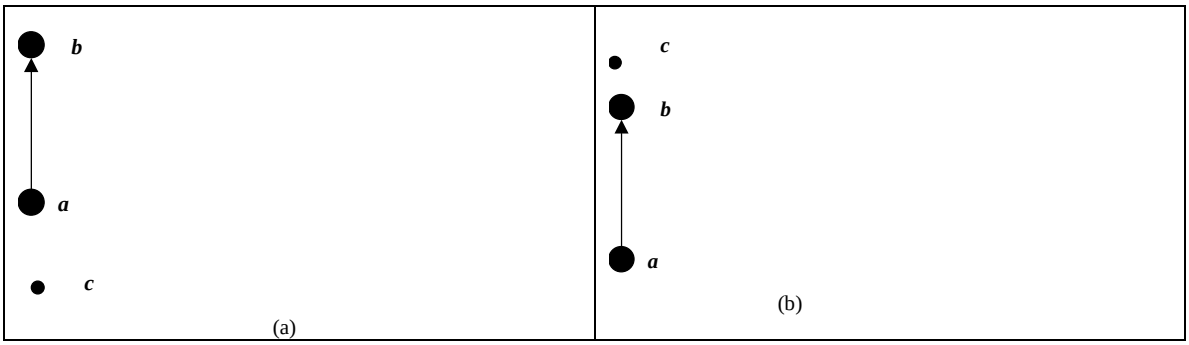


Figure 15–3 . Back (a) and in front (b) position relations representation.

Consider three basic operators applied to spatial vectors*: inversion*, *homing*, and *shortcut*.

Inversion operator I means the inversion of a navigation vector: *I*(*ab*) = *ba*.
Homing operator H describes the following operation: given a relation "*c* : *ab*" , find a relation " *a* : *bc*", i.e. *H*(*c*: *ab*) = *a*: *bc*.
Shortcut operator S describes the operation: given a relation "*c* : *ab*" , find a relation " *b* : *ac*", i.e. *S*(*c*: *ab*) = *b*: *ac*.
This model was be applied for the simulation of human orientation and path description based on landmarks. Consider, for example, some physical space consisting of the following landmarks: the road (designated by *ab*), the church (*c*), the road (*bd*) and the walking human being (designated by *h*). Let we have the following route description given to *h* by some human person when he is asked by *h* "where is university?:
"Walk down the road (*ab*). You will see a church (*c*) on the left. Before you reach the church turn right, The road (*bd*) leads forward to the university (*g*)." (Fig.15-4).
This route description may be formally represented as follows:

(*h straight*: *ab*) $\wedge$ (*c left*: *ab*) $\wedge$ (*d right*: *ab*) (for path1)
and (*h straight*: *bd*) $\wedge$ (*g front*: *bd*), (*c back*: *bd*) (for path2).

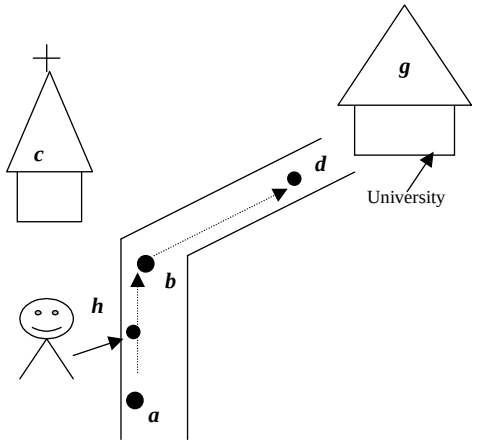


Figure 15-4. Graphical picture of a scene.

Of course, this model has a lot of limitations, for example, it can't describe 3D relative positions between objects, there are no tools for the description of distance between objects, etc.

1.2 Spatial model for interpretation of graphical representation of scenes

Spatial model of Reiter-Mackworth (Reiter& Mackworth, 1990) [13] has been developed for the interpretation of graphical representation of scenes (in particular for maps showing roads, rivers, regions, etc.) and was based on *the predicate calculus of the first order including special spatial predicates*. The main primitive of this model is the concept of "*image of object*" which is described by the following axiom:

$$(\forall x)\, image-of-object(x) = chain(x) \vee region(x),$$

$$(\forall x) \neg((chain(x) \wedge region(x)).$$

Primitives for objects of the given scenes (such as a road, a river, a lake, ets.) and relationships between them have been introduced . For example, *tee* (*c*,*c*1) means that chain *c* intersects with chain *c*1 .
The axioms defining primitive's properties and relations between them are considered,
for example, as follows:

$$(\forall x, y) tee(x, y) \Rightarrow chain(x) \wedge chain(y).$$

In the spatial model of Reiter-Mackworth, a special type of axioms (called *scene axioms*) showing spatial relations for the given scene is introduced.

1.3 Extension of Allen's temporal logic to spatial case

Donikian and Hegron (Donikian & Hegron, 1991) [14] try to construct spatial model as the extension of Allen's temporal logic to the case of three dimensional (3d) space. In this case, temporal intervals are replaced by spatial intervals on *X*-*Y*-*Z* axes and time moments are replaced with points in the 3d space. For intervals on the every axis in a space, seven Allen's relations between intervals ($x, y \in X(Y, Z)$) are considered (Fig.15-5).

(Copia modificata per una migliore consultazione on-line)

x before *y*	*x* ___ *y* _____	*x* starts *y*	*x*____ *y*_____
x meets *y*	*x*____ y____	*x* finishes *y*	*x*____ *y*_____
x overlaps *y*	*x*____ *y*____	*x* equals *y*	*x*____ *y*____
x during *y*	*x*__ *y*_____		

Figure 15-5. Relations between spatial intervals on the axis.

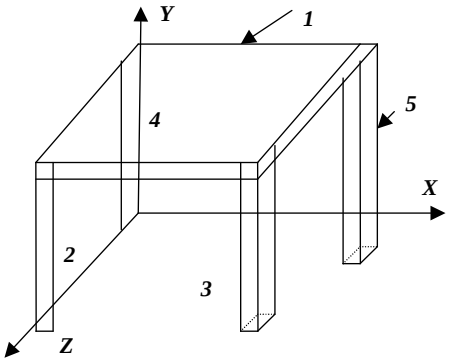


Figure 15–6. Graphical representation of a “desk”.

Objects in this spatial model are built by means of boxes, and spatial relations between object’s components are described by using mentioned above relations. For example, the object “a desk”is described as follow:

(2 starts 1) $\wedge$ (4 equals 2) $\wedge$ (3 finishes 1) $\wedge$ (5 equals 3), (for the *X* axis)

(2 meets 1) $\wedge$ (3 equals 2) $\wedge$ (4 equals 2) $\wedge$ (5 equals 2), (for the *Y* axis)

(2 finishes 1) $\wedge$ (3 equals 2) $\wedge$ (4 starts 1) $\wedge$ (5 equals 4), (for the *Z* axis)

where 1,2,3,4,5 are corresponding to *X*-*Y*-*Z* segments of components of the desk (see Fig.15-6) .

Each of Allen’s relations is described by a *subgraph* representing four final points of segments that take place in the relation. For example, the “*X overlaps Y*” relation is described by the subgraph structure shown in (Fig.15-7). Here X_b, X_e, Y_b, Y_e are beginning and ending points of *X* and *Y* segments respectively, and “*les*” is the “before” relation between spatial points.

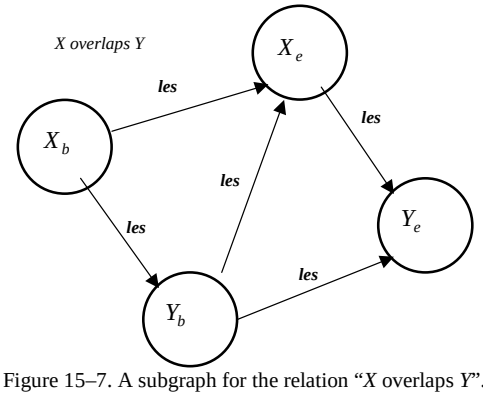


Figure 15–7. A subgraph for the relation “*X* overlaps *Y*”.

Object’s location and size in this model are defined as geometric constraints which are given by labels of corresponding arc in the subgraph. The following geometric constraints are considered:

- the distance between two nodes of subgraph (it may be fixed, or belongs to some interval of values, or may be labeled as unknown);
- multidimensional constraints in the form of analytical equations (for example, for the objects forms, surfaces, volumes description).

Some restrictions between segments may be defined as axioms of spatial logic.

Real spatial situation is described as the joining of graphs corresponding to restrictions.

Special module checks the logical consistency of the given description (in particular it is regarded that logical inconsistency is exposed if a cycle is found in the graph).

Remark 1. From the computational point of view, the use of spatial intervals on *X*-*Y*-*Z* axes as basic spatial primitives is not efficient. This approach can be used for static space descriptions and may be interesting for some CAD applications, but the model is not applicable for a complex dynamic world’s description

2. Advanced Spatial Models

2.1 Qualitative spatial reasoning based on a fuzzy logic

Qualitative spatial reasoning about space forms an integral part of our daily life. This kind of reasoning can be important for intelligent autonomous agents (e.g., robots), especially for those operating in uncertain or unknown or dynamic environments. In such situations, several factors can enhance the benefits of using qualitative spatial reasoning techniques:

- it may be difficult (or often impossible) to collect precise information about the environment;
- there may be real constraints of memory and time which prevent either the collection of large volumes of data or the utilization of a large amount of computation time;

- in hostile environments, quick reactions to sudden stimuli may be facilitated by qualitative reasoning;
- man-machine interaction can be enhanced ,e.g., instructions to robot can be given in natural language including spatial expressions.

A lot of research is connected with qualitative spatial reasoning based on the use of *fuzzy logics*. Let us illustrate how spatial relations expressed in natural language can be represented in a spatial model based on the fuzzy technique. Consider, for example, the following spatial information:
"Object A is about 5 miles from object B in a north-easterly direction" (see Fig.15-8).
We can represent relative positions of objects *A* and *B* by using fuzzy numbers *M* and *N* as follows (see Fig.15-9).

$$\text{"About 5 miles"} = M = (5,1,1)_{LR}$$

and

$$\text{"North-easterly direction"} = N = (45,10,10)_{LR}.$$

Assume that we have the following spatial descriptor statement:
Object A is about 2 miles from object B, and object C is more or less 3 miles of object B. (Note that in these two statements we are only concerned with the one-dimensional case.) How to find the answer on the following spatial query: *What is the spatial relation between objects A and C*?
Consider the following spatial fuzzy relation:

R (*n*) – object *A* is at about *n* unit distance from object *B*.
Then we can describe the mentioned above spatial descriptor statement as following expression:

$$(AR(2)B) \wedge (CR(3)B).$$

Unknown spatial relation between objects *A* and *C* can be inferred by using the following axiom:

$$(AR(n)B) \wedge (CR(m)B) \Rightarrow (AR(n \oplus m)C,$$

where $n \oplus m$ is defined as a fuzzy addition of two fuzzy numbers.

Remark 2. Note that in the axiom we are only concerned with the one-dimensional case.

Assume that fuzzy sets "about 2 miles" and "more or less 3 miles" are represented by fuzzy numbers $M = (2,1,1)_{LR}$ and $N = (3,1,2)_{LR}$ respectively. Then the spatial relation between objects *A* and C can be calculated as $M \oplus N = (5,2,3)_{LR}$ which is to be interpreted as saying that object *C* is about 5 miles from object *A*.

First spatial models give an impulse to development of more perfect models applicable for a complex dynamic world's description.

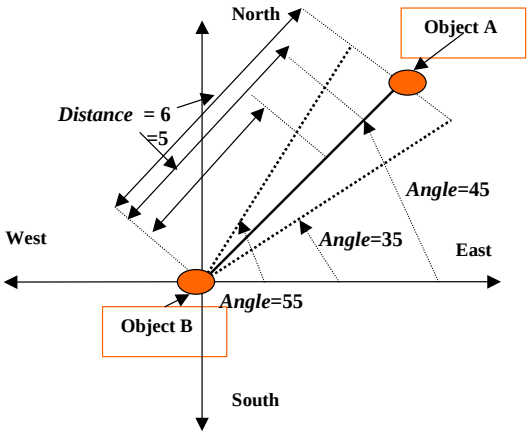


Figure 15-8. Fuzziness in location of object A relative to object B.

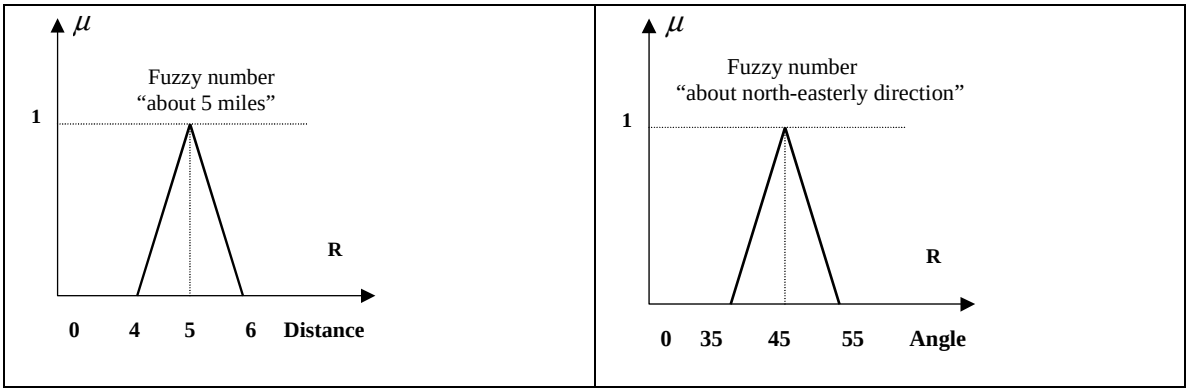


Figure 15-9. Relative positions representations as fuzzy numbers.

2.2 Pseudo-Physical Spatial Model

Consider the pseudo-physical spatial logic for spatial knowledge representation and reasoning [7]. This model will be used for robot's intelligent behavior simulation and artificial life description [15].

The *Pseudo-Physical Logic of Space* (*PLS*) is the logical deductive system that allows to imitate human reasoning about spatial situations in a real environment space.

PLS consists of three main parts:

- the *objects system*, -

- the *spatial relations system*, and
- the *axioms system* , describing algebraic and semantic properties of relations, and human space perception properties.

The *objects system* includes the following main classes of objects that are characteristic for the human spatial perception:

- unique objects that are perceived as separate entities (for example, a house, a chair, a human being) ;
- objects parts that are distinguished according to the definite criteria (a chair's leg, a house's chimney) ;
- structural elements that are elementary components in more complicated objects descriptions ;
- "localizations" (places) that are some parts of a living space containing a set of objects (for example, rooms, buildings, cities, etc.).

The *system of relations* has to reflect spatial aspects of a real world: a mutual location of objects in static; a mutual location of objects in dynamics; an objects orientation and location in a space; spatial connections between object's parts, and so on.

PLS is designed as a formal theory based on the predicate logic of the first order. It consists of the following 4-tuple: $\{A, G, \Omega, I\}$, where *A* (alphabet) is a set of basic elements, *G* (syntax) is a set of rules for *WFF* definition, Ω is a set of axioms, and *I* is a set of inference rules.

Alphabet

Alphabet *A* includes terms (variables and constants), predicates, functional symbols, logical operations symbols and service symbols.

Terms

In order to formalize spatial knowledge let us include objects, spatial relations and spatial scenes as objects in the universe of discourse. Then we may use their names in spatial statements describing some real world situations. So, the set of terms consists of subsets of different sort (or types) terms. We introduce special designations for each sort of terms.

We will consider the following sorts:

- a finite set of objects names $O = \{o_1, o_2, ..., o_i\}$;

Remark 3. The set of object's names includes names of different objects (such as physical objects, human beings, animal, etc.).

- a set of all spatial scenes (situations, states) of a given problem domain $S = \{s, s_1, ..., s_S\}$;
- a set of spatial points $X = \{x_1, x_2, ..., x_i, ...x_k, ...\}$, where each x_k is a point in a 3D space with a given coordinate system;
- a set of spatial intervals $D = \{D_1, ..., D_j, ...\}$, where each D_j is an interval in a given axis (*X* or *Y* or *Z* axes) with a given measure unit;
- a set of measure units names $U = \{u_1, u_2, ...u_n\}$;
- a set of localizations (places) names $L = \{l_1, l_2, ...l_m\}$;
- a set of real numbers $N = \{n, n_1, ..., n_k, ...\}$;
- a set of fuzzy numbers $\tilde{N} = \{(\tilde{n}, \mu_{\tilde{n}}), ..., (\tilde{n}_k, \mu_{\tilde{n}_k}), ...\}$, where $\tilde{n}_k$ is a fuzzy number described by membership function $\mu_{\tilde{n}_k}$;
- a set of linguistic variables for objects sizes $Z = \{z_1, z_2, ..., z_k\}$ described by membership function μ_{z_k} .

So, in *PLS* model the following basic types of terms are introduced:

[*object*, *point*, *interval*, *localization*, *number*, *measure unit*].

A set of *measure units* consists of a finite list of measure unit names. For example, km, m, cm, mile, etc.

Consider now spatial predicates defined as spatial relations by the following way.

Predicates and functional elements

The set of predicates consists of static spatial relations and dynamic spatial relations.

Functional elements represent arithmetic operations over numbers and fuzzy numbers.

II. Static Spatial relations

To simulate human spatial reasoning we introduce the following sets of static spatial relations.

I.1 Nonmetrical static spatial relations:

Relative position relations are used for the description of mutual locations of objects or spatial localizations. We will consider the following set of relative positions relations (Fig.15-10):

$o_1 R_1 o_2$ - object o_1 is located in object o_2 (or in localization l_1),

$o_1 R_2 o_2$ - object o_1 is located on object o_2 ,

$o_1 R_3 o_2$ - object o_1 is located above object o_2 ,

$o_1 R_4 o_2$ - object o_1 is located under object o_2 ,

$o_1 R_5 o_2$ - object o_1 (l_1) is located in front of object o_2 (l_2) (from an observer point of view),

$o_1 R_6 o_2$ - object o_1 (l_1) is located behind of object o_2 (l_2) (from an observer point of view),

$o_1 R_7 o_2$ - object o_1 (l_1) is located to the left (side) of object o_2 (l_2) (from an observer point of view),

$o_1 R_8 o_2$ - object o_1 (l_1) is located to the right (side) of object o_2 (l_2) (from an observer point of view),

$o_1 R_9 (o_2, o_3)$ - object o_1 (l_1) is located in between object o_2 (l_2) and object o_3 (l_3).

For example, "a book is on a table", "John is in Milan", "A park is located in between a river and a road" , "Mary is to the right of the dog".

Proximity relations are used for the qualitative description of a distance between objects or localizations in a space (Fig.15-11):

(Copia modificata per una migliore consultazione on-line)

$o_1 r_0 o_2$ - object $o_1(l_1)$ is located closely (so that it is in contact with) to object $o_2(l_2)$,

$o_1 r_1 o_2$ - object $o_1(l_1)$ is located very near object $o_2(l_2)$,

$o_1 r_2 o_2$ - object $o_1(l_1)$ is located near object $o_2(l_2)$,

$o_1 r_3 o_2$ - object $o_1(l_1)$ is located not near and not far from object $o_2(l_2)$,

$o_1 r_4 o_2$ - object $o_1(l_1)$ is located far from object $o_2(l_2)$,

$o_1 r_5 o_2$ - object $o_1(l_1)$ is located very far from object $o_2(l_2)$.

For example, "object A is near object B", or "Soncino is near Crema", or " The lake is far from the park".
Remark 4. All mentioned above proximity relations are described by membership functions.

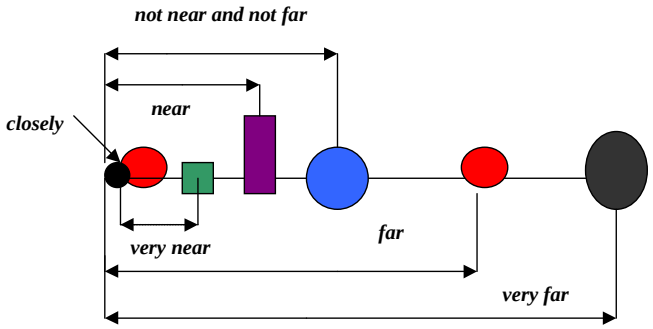


Figure 15–11. Proximity relations.

By using ordinary logical operations (usually conjunction) more complex relations can be described. For example, "Object A is far and in front of object B", "Object C is near and right of object B", etc.

I.2 Metric static spatial relations (crisp):

$o_1 R_{M0} x_k$ - position of the object $o_1(l_1)$ is described by point x_k ,where x_k is a point in a 3D space with a given coordinate system;

$o_1 R_{M1}(n) o_2$ - object $o_1(l_1)$ is located at the distance of *n* units from object $o_2(l_2)$.

For example, " Object C is in 5 miles from object B" or " Crema (town) is located in 50 km from Milan".

Metric static spatial relations (fuzzy):

$o_1 R_{F1}(N) o_2$ - object $o_1(l_1)$ is located at the distance about *N* units from object $o_2(l_2)$.

For example, " Object C is in 5 miles approximately from object B".

I.3 Orientation spatial relations (crisp or fuzzy) (see Fig.15-8):

$o_1 R_{M2} o_2$ - object $o_1(l_1)$ is located in a north direction from object $o_2(l_2)$;

$o_1 R_{M3} o_2$ - object $o_1(l_1)$ is located in a south direction from object $o_2(l_2)$;

$o_1 R_{M4} o_2$ - object $o_1(l_1)$ is located in an east direction from object $o_2(l_2)$;

$o_1 R_{M5} o_2$ - object $o_1(l_1)$ is located in a west direction from object $o_2(l_2)$;

$o_1 R_{M6} o_2$ - object $o_1(l_1)$ is located in a north-easterly direction from object $o_2(l_2)$;

$o_1 R_{M7} o_2$ - object o_1 is located in a north-westerly direction from object $o_2(l_2)$;

$o_1 R_{M8} o_2$ - object $o_1(l_1)$ is located in a south-easterly direction from object $o_2(l_2)$;

$o_1 R_{M9} o_2$ - object $o_1(l_1)$ is located in a south-westerly direction from object $o_2(l_2)$;

For example, "object A is about 5 miles from object B in a north-easterly direction" or "object A is in a west direction from the Bozeman city".
Remark 5. Spatial relations I.3 may be described as fuzzy sets shown in Figs.15-8,9.

I.4 *Special locations r*elations (Fig.15-12):

$o_1 R_{L1} o_2$ - object o_1 is located in the center of object $o_2(l_2)$;

$o_1 R_{L2} o_2$ - object o_1 is located in the left back angle of object $o_2(l_2)$;

$o_1 R_{L3} o_2$ - object o_1 is located in the right back angle of object $o_2(l_2)$;

$o_1 R_{L4} o_2$ - object o_1 is located in the left front angle of object $o_2(l_2)$;

$o_1 R_{L5} o_2$ - object o_1 is located in the left front angle of object $o_2(l_2)$.

For example, “A desk is in the center of the room”, “A chair is in the left front angle of the room”, “A church is located in the center of the city”.

Remark 6. The choice of this set of relations depends from a given problem area and task, for example may be also introduced such kind of relations as “ to be in the top (bottom) of the object ”, etc.

I.5 *Dynamic spatial relations* (*D*-relations) (Fig.15-13):

$o_1 R_{D1} o_2$ - object o_1 (l_1) is located right from a direction of motion of object o_2 ;

$o_1 R_{D2} o_2$ - object o_1 (l_1) is located left from a direction of motion of object o_2 ;

$o_1 R_{D3} o_2$ - object o_1 (l_1) is located in front of a direction of motion of object o_2 ;

$o_1 R_{D4} o_2$ - object o_1 (l_1) is located back to a direction of motion of object o_2 ;

$o_1 R_{D5} o_2$ - object o_1 (l_1) is located in the angle α right from the direction of motion of object o_2 ;

$o_1 R_{D6} o_2$ - object o_1 (l_1) is located in the angle α left from the direction of motion of object o_2 .

For example, “Walking down the road Ben saw a nice church on the left”.

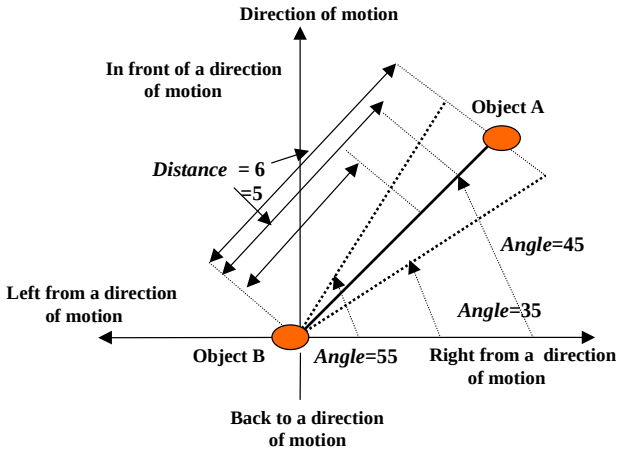


Figure 17–13. Dynamic spatial relations.

II. Size Relation and Size Linguistic Variables

For describing object’s sizes we will consider the following size relation:

$o_1 R_{size} z$ - object o_1 (l_1) has a size *z*,

and following size linguistic terms:

$o_1 R_{size} VS$ - object o_1 is very small;

$o_1 R_{size} S$ - object o_1 is small;

$o_1 R_{size} M$ - object o_1 has a middle size;

$o_1 R_{size} L$ - object o_1 is large;

$o_1 R_{size} VL$ - object o_1 is very large.

Remark 7. Linguistic variables *very small*, *small*, *middle size*, *large* and *very large* are described by corresponding fuzzy sets.

III. Properties description predicate

In order to describe that some property *p* is true in a given spatial scene *s* we introduce a relation $T(p,s)$, where *p* is some well formed formula (*WFF*) and $s \in S = \{s, s_1, ..., s_S\}$ the set of all spatial scenes of a given problem domain.

T(($o_1 R_2 o_2$), *s*) means that the property “object o_1 is located on object o_2 ”is true in particular spatial scene *s*.

Syntax of PLS Model

Introduce following syntactical rules for *WFF*.

8. Any term is *WFF*.
9. Any relation in the form I.1, I.2, I.3, I.4, I.5, II, III is *WFF*.
10. If α and β are *WFF*, then $(\alpha \wedge \beta)$, $(\alpha \vee \beta)$, $\neg\alpha$ are also *WFF*.
11. There are no another *WFF*.

Remark 8. Later, in the action model, instead of $(o_1 R o_2)$ expression we will use also $R(o_1, o_2)$ expression.

Remark 9. In *PLS* Model operations $(\alpha \wedge \beta)$, $(\alpha \vee \beta)$, $\neg\alpha$ are interpreted in the set-theoretical sense, that is, as follows:

- operation $(\alpha \wedge \beta)$ interpreted as simultaneous presence of facts α, β in a spatial scene.
- operation $(\alpha \vee \beta)$ interpreted as presence of one of facts α, β (or two) in a spatial scene, and
- operation $\neg\alpha$ interpreted as the absence of the fact α in a spatial scene.

To simulate a human spatial reasoning we must develop a set of axioms describing properties of spatial relations introduced above.

Human Spatial Reasoning Simulation

Pseudo-physical spatial model for the human spatial reasoning simulation is the aggregate of several components that reflect different characteristics of the spatial relations. *PLS* consists of two main parts: static spatial logic and dynamic logic

The static spatial logic describes properties of static spatial relations, the dynamic logic describes properties of dynamic spatial relations and a change of a world caused by performance of actions. The kernel of *PLS* consists of the set of axioms describing different properties (algebraic, semantic and human time perception) of basic spatial relations. Let's consider some fragments of axioms considered in *PLS*.
The axioms are given in the form of general schemes as follows.

I. Static Spatial axioms

I.1 Axiom Schemes for Relative positions *relations*

8. $o_1 R o_2 \Rightarrow \neg(o_2 R o_1), R \in \{R_1, R_2, ..., R_9\}$ (nonsymmetry property);
9. $o_1 R o_2, o_2 R o_3 \Rightarrow o_1 R o_3 \;\; R \in \{R_1, R_3, R_4, ..., R_8\}$ (transitivity property);
10. $o_1 R_2 o_2, o_2 R_2 o_3 \Rightarrow \neg(o_1 R_2 o_3)$ (nontransitivity property);
11. $o_1 R_2 o_2 \Rightarrow o_1 r_0 o_3$;
12. $o_1 R_3 o_2 \Rightarrow o_2 R_4 o_1$;
13. $o_1 R_5 o_2 \Rightarrow o_2 R_6 o_1$;
14. $o_1 R_7 o_2 \Rightarrow o_2 R_8 o_1$;
15. $o_1 R_9 (o_2, o_3) \Rightarrow (o_2 R_7 o_1) \wedge (o_3 R_8 o_1)$;
16. $o_1 R_1 o_2, o_2 R_2 o_3 \Rightarrow o_1 R_3 o_3$;
17. $o_1 R_3 o_2, o_2 R_2 o_3 \Rightarrow o_1 R_3 o_3$;
18. $o_1 R_2 o_2, o_3 R_4 o_2 \Rightarrow o_1 R_3 o_3$;
19. $o_1 R_1 l_2, o_2 R_2 o_1 \Rightarrow o_2 R_1 l_2$.

I.2 Axiom Schemes for Proximity *relations*

1. $o_1 r o_2 \Rightarrow o_2 r o_1, r \in \{r_0, r_1, r_2, ..., r_5\}$ (symmetry property);
2. $o_1 r o_2, o_2 r o_3 \Rightarrow \neg(o_1 r o_3), r \in \{r_0, r_1, r_2, ..., r_5\}$ (nontransitivity property);

In human spatial reasoning the evaluation of a distance between two objects, *x* and *y*, depends upon the sizes of these objects. Assume that objects *x, y, z* are located along one line. For this case the following axiom schemes can be used:

3. $(o_1 r_0 o_2) \wedge (o_2 r_0 o_3) \wedge (o_1 R_{size} VS) \wedge (o_2 R_{size} VS) \wedge (o_3 R_{size} VS) \Rightarrow (o_1 r_1 o_3)$;
4. $(o_1 r_1 o_2) \wedge (o_2 r_1 o_3) \wedge (o_1 R_{size} S) \wedge (o_2 R_{size} S) \wedge (o_3 R_{size} S) \Rightarrow (o_1 r_2 o_3)$;
5. $(o_1 r_2 o_2) \wedge (o_2 r_2 o_3) \wedge (o_1 R_{size} L) \wedge (o_2 R_{size} L) \wedge (o_3 R_{size} L) \Rightarrow (o_1 r_4 o_3)$; etc.

I.3 Axiom Schemes describing semantics of Proximity *and* Relative positions *relations*

1. $(o_1 R_1 o_2) \wedge (o_3 R_2 o_1) \Rightarrow (o_3 R_1 o_1)$;
2. $(o_1 r_2 o_2) \wedge (o_3 R_2 o_2) \Rightarrow (o_1 r_2 o_3)$.

I.4 Naive physics –based Axiom Schemes

We introduce also a special class of *naive physics-based* axioms describing "*semantic incorrectness*" of some spatial relations in static scenes from the standpoint of the physics.

For example:

1. $\Rightarrow \neg((o_1 R_1 o_2) \wedge (o_1 R_{size} L) \wedge (o_2 R_{size} M))$;

2. $\Rightarrow \neg((o_1 R_1 o_2) \wedge (o_1 R_{size} L) \wedge (o_2 R_{size} S))$;
3. $\Rightarrow \neg((o_1 R_1 o_2) \wedge (o_1 R_{size} M) \wedge (o_2 R_{size} S))$.

I.5 State constrains axioms

Following axioms (the so-named *state constrains axioms*) introduce also semantic incorrectness of static spatial scenes:

4. $\Rightarrow \neg((o_1 R_3 o_2) \wedge (o_1 R_4 o_2))$;
5. $\Rightarrow \neg((o_1 R_5 o_2) \wedge (o_1 R_6 o_2))$;
6. $\Rightarrow \neg((o_1 R_7 o_2) \wedge (o_1 R_8 o_2))$;
7. $\Rightarrow \neg((o_1 r_i o_2) \wedge (o_1 r_j o_2))\ r_i, r_j \in \{r_0, r_1, r_2, ..., r_5\}, r_i \neq r_j$.

I.6 The following axioms are introduced for relation T:

1. $\forall p \forall s\ \ T(\neg p, s) \Leftrightarrow \neg T(p, s)$;
2. $\forall p \forall q \forall s\ \ T(p \wedge q, s) \Leftrightarrow (T(p, s) \wedge T(q, s))$;
3. $\forall p \forall q \forall s\ \ T(p \vee q, s) \Leftrightarrow (T(p, s) \vee T(q, s))$;
4. $\forall p \forall q \forall s\ \ T(p \Rightarrow q, s) \Leftrightarrow (T(p, s) \Rightarrow T(q, s))$.

This class of axioms may be used for the task of analysis of spatial scenes description (see below example of application) or in some truth maintenance system for spatial scenes.

The *pseudo-physical logic of space* has been used in the simulation system for intelligent behavior of mobile service robot [15]. We will consider this application in following lectures.

3. Application aspects of spatial models

Spatial reasoning is a very general problem applicable to many different domains: human cognition, robot path and action planning, autonomous vehicle control, military tactical situation assessment, general purpose planning, animation and virtual reality systems; design and drafting automation; creative arts including parks and gardens design; artificial life of robots description, natural language (NL) understanding, etc.

Consider spatial models application for some of these tasks.

3.1 Visualization of Natural Language-based Descriptions of Scenes

Pseudo-physical spatial model (PLS) has been applied in the intelligent *Text-Picture* system developed for visualization of natural language-based descriptions of scenes [16]. In this system the mechanism of *scene understanding* was investigated.

Scene understanding is considered as the capability to analyze spatial relations given by a NL-text and to map a symbolic description of a scene into its visual representation. So, “Text-Picture” system replies by drawing on the screen an image mirroring its own “understanding” of the NL scene description.

In order to derive as many facts as it needs to draw a visual image *spatial reasoning* is used. Spatial reasoning was realized on the base of PSL where some spatial relations as normal defaults have been introduced.

Discuss a spatial scene understanding capability implemented in the system. Consider the following NL-description of a scene: "The robot is located in room 415 near a desk. A computer is located on the desk."

The spatial scene understanding capability includes correct answering on the following questions: “What spatial relation is between the robot and the computer?”, “Where is the computer located?”, ”Is the computer located in the room?”, “Is the computer near (far, etc.) the robot ?” and so on.

Let introduce the following names for objects: o_1 - the robot; o_2 - room 415, o_3 - the desk, o_4 - the computer. Designate spatial relation "to be in" as R_1, spatial relation "to be on" as R_2, and spatial relation "to be near" as r_3. Then the internal representation of initial scene will be as follows:

$$(o_1 R_1 o_2) \wedge (o_4 R_2 o_3) \wedge (o_3 r_3 o_1).$$

By using the spatial logic, the module of spatial analysis of a scene can infer and add to the initial spatial scene description the following two relations:

(1) $o_4 R_1 o_2$ ("the computer is in the room 415") obtained according to the axiom schemes (I.3(1)) ;

(2) $o_4 r_3 o_1$ ("the computer is near to the robot") obtained according to the axiom scheme (I.3(2)) .

Graphical Designer of the system is used for the construction of internal graphical representation (*GR*) of a current scene. For this task a special method for defining of object’s location described by spatial relations is introduced.

Let *pct* be a picture in which objects *A* and *B* are defined. Let $pct(A)$ and $pct(B)$ be 3D-images (sets of points in a given coordinate system) of objects *A* and *B* respectively.

In order to construct a picture where spatial relation *R* between objects *A* and *B* is depicted, we introduce special “visualization” semantics for our spatial relations. Consider a few examples:

A left B: means that in the picture *pct* the *x*-coordinate of every point in $pct(A)$ is less than *x*-coordinate of every point in $pct(B)$;

A right B: means that in the picture *pct* the *x*-coordinate of every point in $pct(A)$ is greater than the *x*-coordinate of every point in $pct(B)$;

A above B: means that in the picture *pct* the *y*-coordinate of every point in $pct(A)$ is greater than the *y*-coordinate of every point in $pct(B)$;

A behind B: means that in the picture *pct* the *z*-coordinate of every point in $pct(A)$ is greater than the *z*-coordinate of every point in $pct(B)$;

A inside B: means that in the picture *pct* $pct(A) \subseteq pct(B)$;

A outside B: means that in the picture *pct* $pct(A) \bigcap pct(B) = 0$, i.e. objects *A* and *B* do not have any common points.

The distance L_{AB} between two objects *A* and *B* connected with fuzzy spatial relation *R* is the function of following parameters:

$L_{AB} = F(R, L_A, L_B, L_{scene})$, where L_A, L_B and L_{scene} are the sizes of object *A*, object *B* and a given scene respectively.

For spatial proximity relations special rules of computing L_{AB} are introduced as follows:

(Copia modificata per una migliore consultazione on-line)

$$L_{AB} = \begin{cases} K_{closeness}(L_{scene} - L_A - L_B), \ L_A \approx L_B \\ (K_{closeness} + L_B / L_A)(L_{scene} - L_A - L_B), L_A >> L_B \end{cases},$$

where $K_{closeness}$ is the maximal value of a membership function of fuzzy relation *R*: $K_{closeness} = \max_u \mu_R(u)$, where *u* is the universal linguistic scale of distance.

The output from the *Graphical Designer* is the set of object's coordinates in the coordinate system connected with a given spatial scene. This set of object's coordinates is called the *cognitive map* of a given spatial scene.

Vizualizer converts the cognitive map of the given scene into the set of physical coordinates of a display and represents 3D images of spatial scenes on the display screen.

In Fig.15-14, the graphical representation of the following initial scene description is shown: "In a room there is a table, wardrobe, a chair and a sofa. The table is located in the center of the room. The chair is right and near from the table. The sofa is behind and near the table. The wardrobe is right from the table and not far it. A lamp and a note are located on the table".

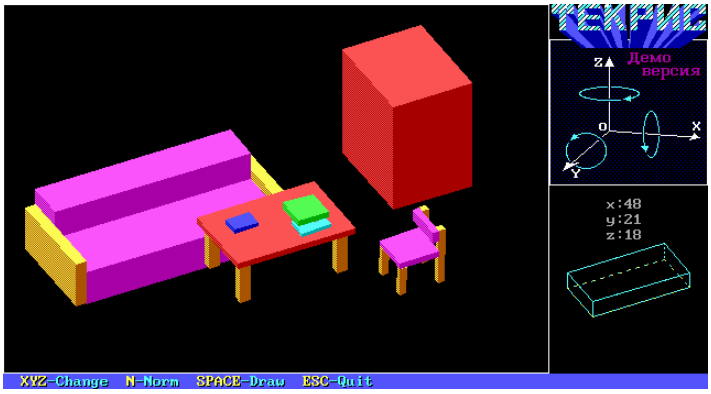


Figure 15–14. 3D graphical representation of a spatial scene described by the NL text.

Graphic Editor realizes the functions of construction and modification of graphical objects. The example of the graphical representation of an object is shown in the Fig.15-15.

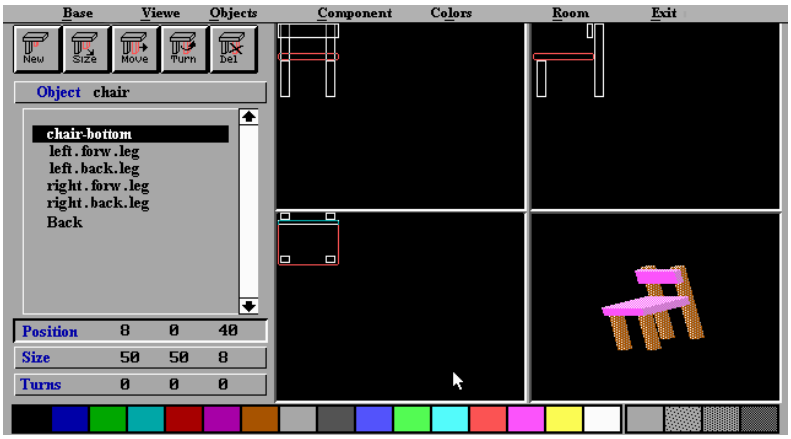


Figure 15-15.Example of the graphical representation of the chair.

Remark 10. As shown in Fig.15-15, an object can graphically be represented by two ways: in the form of three projections of the object (in front view, view from left, and view from above); and in the form of isometric projection called as 3D representation of the object.

Another examples of spatial scene understanding and visualizing are shown in Fig. 15-16,17,18 for the following examples.

Remark 11. The "Text-Pictures" system was developed to understand and visualize texts given in Russian language. In examples below we give English translation of input texts.

Example 1. Consider the following NL scene description (Text 1):

"A small bird is sitting on the top of a house. Four trees (firs) are growing to the left side of the house and two limes are located right. A car is moving in front of firs. There is a bench in front of the limes. Two peoples are sitting down on the bench. One of them is reading a book."

In Fig.15-16 the picture representing a situation described by the Text 1 is shown.

(Copia modificata per una migliore consultazione on-line)

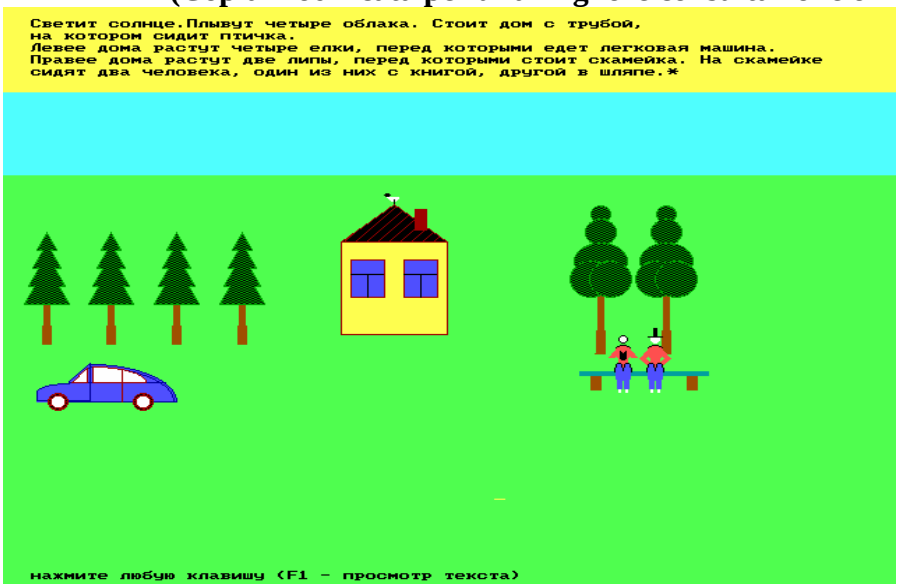


Figure 15-6.

Example 2 (Text 2):
“ There is a house with a fence and a wicket-gate. A fir is growing behind the house. Left from the house two peoples are bringing a box. A bird is sit down on the box. Two trees (limes) are left from the peoples bringing the box. Two students on bikes are moving along the house. A truck is moving in front of the students.”
In Fig.15-17 the picture representing a situation described by the Text 2 is shown.

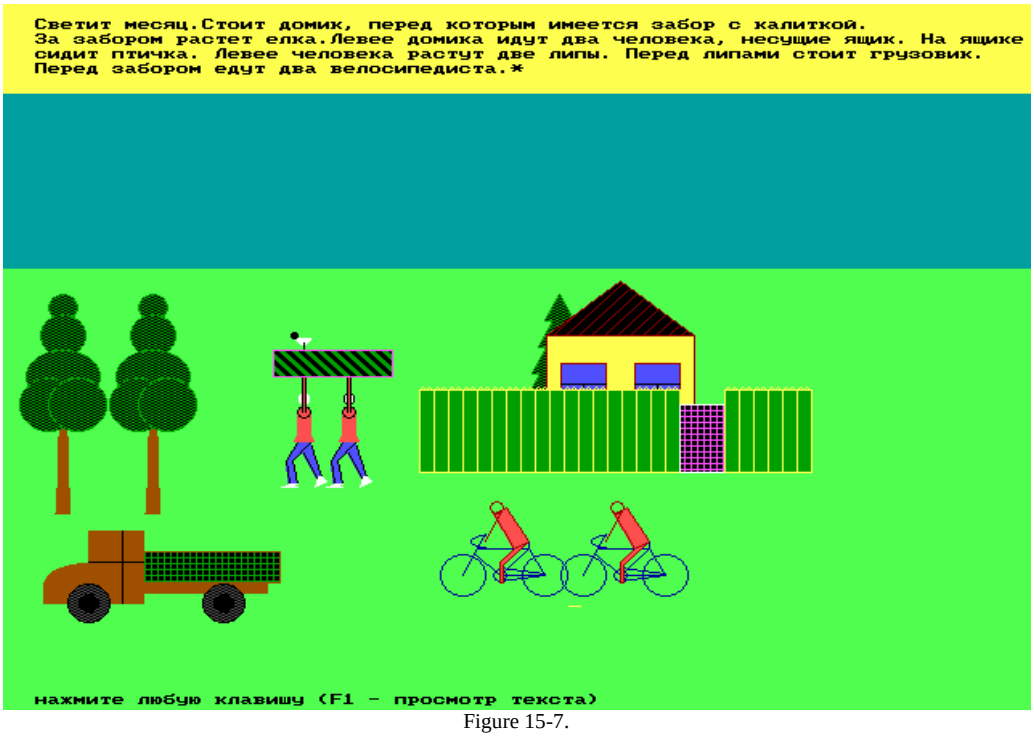


Figure 15-7.

Example 3 (Text 3):
“A man with a shovel is standing in the body of a truck with a gravel. Near the truck there is a pile of gravel, and another man with a shovel is standing near the pile. Two trees (limes) are located right from the truck. Near the limes there is a house. A smoke goes from a tube of the house. A man is sleeping on a bench located in front of the limes. Two cars are moving along the house, and three peoples on bikes are moving in front of the cars.” In Fig.15-18 the picture representing a situation described by the Text 3 is shown.

(Copia modificata per una migliore consultazione on-line)

Figure 15-8. Graphical representation ofText 3.

Remark 12. Similar research is developed in [17] (Clay & Wilhelms, 1996) where an object-placement system called *Put* was implemented. The system allows direct manipulation of an object by using a combination of linguistic commands. For this task a spatial model based on spatial relations like “on”, “in”, “at”, “in front of” etc. is also developed. The *Put* language allows make a specification of objects geometry and properties as well as spatial relationships. The interactive environment, including viewing and manipulation, control panels and linguistic interface have been constructed.

3.2 Spatial qualitative modeling for intelligent mobile robots

In robotics one of the basic aims is to build robots capable to automatically planning motions with or without exact quantitative information about their environment. Prior knowledge about environment, for example maps, typically contains some metric information and information about spatial relations between objects.
It will be useful to have a *motion planner* that can perform spatial reasoning and construct a motion path, for example from initial point 1 to target point 2 in a given environment. In this case spatial model must contain the formalism for describing distances, angles and locations of objects and inference rules for deriving a collision-free path for robot moving among objects in environment. This formalism is, for example, the *pseudo-physical spatial model* described above.
A lot of path planner algorithms are developed in robotics. *Path planner* performs spatial reasoning and finds a global near-optimal route given only a qualitative geometric description of the spatial map (see, for example, [18]). Within such route the planner can further generate local path plan based on some sensory feedback. For this task a *fuzzy component of a spatial model* is used.

Locomotion control for path tracking task

Locomotion control for path tracking determined by a planned path and fuzzy inference mechanism based on set of spatial fuzzy rules. The planned path can be obtained, for example, by using genetic algorithm and spatial hierarchical node map (NP) method [19]. For this task a special fuzzy component of PLT *was developed and a simple Mamdani's fuzzy inference method is used.*

The following three kinds of parameters are taken as *input of fuzzy inference for path tracking control*:
D_p – a distance between planned path and robot; D_T – a distance between target and robot; A_T – an angle between a target and a robot (4 fuzzy sets are chosen for D_p , and 7 fuzzy sets for D_T and A_T).
The following two parameters are taken as *output of fuzzy inference:*
S_T – a steering angle; V_T – a robot's movement speed (7 fuzzy sets are chosen for S_T , and 4 fuzzy sets for V_T).
The following two types of control rules are considered:
1) control rules for steering: the input is D_p and A_T , the output is S_T (the number of rules is 49);
2) control rules for movement speed: the input is D_T and A_T, the output is V_T (the number of rules is 28).

Obstacle avoidance task.

The robot avoids obstacles according to information about distance between the robot and human beings or other obstacle from environment recognition part. We consider 11 types of spatial relations for this task:

- “to be in front of “ (FC0);
- “ to be in left”(SL0);
- “to be in right”(SR0);

four different types for relation

- “to be in the left in the angle interval α ”, (FL0, FL1, FL2, FL3);

and four different types for relation

- “to be in the right in the angle interval α ” (FR0, FR1, FR2, FR3) (see Fig.15-19).

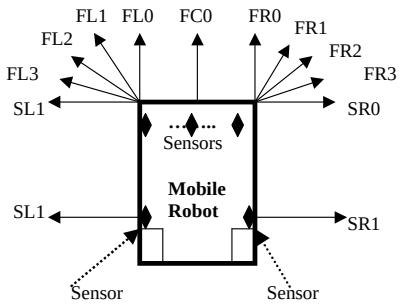


Figure 15-19. Dispositions of robot's sensors.

(Copia modificata per una migliore consultazione on-line)

Infrared (IR) and ultrasonic (US) sensors are disposed on the robot according the given spatial directions as it is shown in the Fig.9. By joint use of infrared (IR) and ultrasonic (US) sensors the robot can detect human beings and obstacles in building which are located around the robot according to considered spatial relations. So, the eleven input parameters for fuzzy control are calculated on the base of information from sensors.

For the task of *obstacle avoidance control* three parameters are used as output of fuzzy inference: S_T - steering angle; V_T - movement speed, and *Ratio* - decision coefficient connected with safety.

Consider example of robots behavior called as *reactive navigation*: following a given path avoid unforeseen obstacles in real time. For this task a *fuzzy component of the spatial model* is used. The fuzzy component include a set of fuzzy rules, for example, as follows:

1) IF (*obstacle is left from a robot and close to him*), THEN *turn right quickly*;
IF (*obstacle is left from a robot and robot is far*), THEN *go straight with normal speed*;
IF (*obstacle is in front of the robot and close to him*), THEN *stop*;
Or
2) IF (*obstacle-right-distance* is *close*) AND (*obstacle-left-distance* is *far*) THEN {*turn left*}
IF (*right-distance* is *close*) AND ¬ (*obstacle -left*) THEN {*turn left*}
IF ¬ (*obstacle* is *close*) THEN {*go to target*},

where "*obstacle-right-distance*"," *obstacle-left-distance*", "*obstacle-left*" are linguistic variables whose values are obtained from sensors, and *close, far, left* are spatial fuzzy relations described by corresponding fuzzy sets.

Remark 13. Fuzzy reasoning technique is described in the first part of our Lectures Notes.

Lecture N 16 Actions Representation Models

In this lecture we will talk about actions that change world states, and discuss how to formalize a change in the world.

The notion of a *state* is central in most models of physical world. This notion allows us to describe changing in a world. We will consider a *state*, or *situation*, as a *snapshot of the world at a given point in time*. At different points of time, the world can be in different states.

We will consider that a *world persists in one state until an action is performed that changes it to a new state*.

In order to formalize actions and changing world we include states and actions as objects in the universe of discourse. Then we may use names of states, for example, as $s_1, s_2, ..., s_n$, and names of actions, for example, as $a_1, a_2, ..., a_k$.

The problems connected with a formal description of actions and a change including the following:

- description of *action results*;
- description of *necessary conditions* for actions;
- *frame problem*, that is, the problem of characterizing the aspects of a state that are not changed by an action,
- description of simultaneous actions,
- description of plans, i.e. actions ordering in a time.

Different action models have different solutions of mentioned above problems.

1. Brief history of action models design

1.1 Situational calculus of McCarty and Hayes

One of the first action models was a *situational calculus* developed by McCarty and Hayes in1969 [20]. They proposed the formalization of *reasoning about time* and *action* constructed as an extension of the classical predicate logic. They introduced a situation variable (or world state) into predicates. The change of world states from one state to another are based on action and described by mapping function. For example, the action of "*painting*" is described by the following mapping function:

$$color(house1, blue, s) \rightarrow color(house1, red, s_1),$$

where *s* is the old situation, and s_1 = *paint* (*house*1,*blue, s*)) is the new situation.

One problem should be mentioned when discussing the issue of a world change description. This is so called *frame problem*:

What should we keep unchanged in the representation when the situation has changed?

How can the system know what has changed without checking all the data? Consider the following task.

Imagine that we have a computer system, which maintains data about a house and monitors whether there is a possibility for a dangerous situation to occur (for example, fire hearth). In order to monitor the house, the system keeps information about location and many other properties and parameters of all objects in the house. A video camera takes pictures and the information about current situation in the computer system is updated regularly. The system analyses the information in order to predict an event which may happen.

For example, a glass drops on the floor in the kitchen. The global situation is changed. How to describe this change? Which data should be updated? To update all the data in the system computer may work for days so that it is unable to react to new changes. Obviously, relationships between objects and some default rules must be introduced.

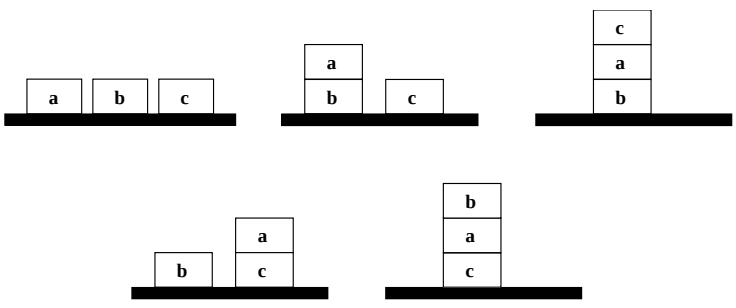
McCarty and Hayes have illustrated the simplest way of an action model design. This action model had a number of deficiencies: changes are discrete, there are no tools for expressing time, simultaneous actions and complex dynamic situations of the real world. To avoid these disadvantages more complicated action models have been developed.

1.2 *State-based action model*

The next simple model of description of states, actions and change is a so-called *state-action model* [21]. The idea of such state- based action logic is nicely illustrated in the context of the Blocks World.

Consider the world consisting of three blocks located on a table. Each block can be somewhere on the table or on a top of exactly one other block. So, different states of this world correspond to different configurations of blocks.

For this world there are 13 possible states (Fig.16-1).



(Copia modificata per una migliore consultazione on-line)

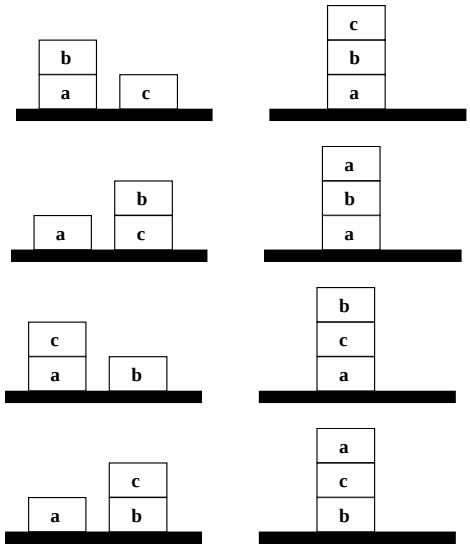


Figure 16-1. State space for the Blocks World.

How to describe changing in this world?

Firstly, we must introduce the description of current scenes of our world. We will consider that each world state represents one static scene of the world. The static scene may be described by using some spatial and other relations. We will use the following relations:

- On (A, B, s_1) means that block *A* is located directly on block *B* in a given state s_1 ;
- Clear (C, s_1) means that block *C* has no blocks on it top in a given state s_1;
- Table (B, s_1) means that block *B* is located on the top of the table in a given state s_1;

In order to describe that some property *p* is true in a given situation *s* we introduce a relation T(p, s). For example, T(On $(A, B), s_1)$ means that the property On(*A*,*B*) is true in particular state s_1.

The following axioms are introduced for relation T:

1. $\forall p \forall s \;\; T(\neg p, s) \Leftrightarrow \neg T(p, s)$
2. $\forall p \forall q \forall s \;\; T(p \wedge q, s) \Leftrightarrow (T(p, s) \wedge T(q, s))$
3. $\forall p \forall q \forall s \;\; T(p \vee q, s) \Leftrightarrow (T(p, s) \vee T(q, s))$
4. $\forall p \forall q \forall s \;\; T(p \Rightarrow q, s) \Leftrightarrow (T(p, s) \Rightarrow T(q, s))$

We introduce also the so-named *state constrains axioms*:

5. $\forall x \forall s \;\; T(Table(x), s) \Leftrightarrow \neg (\exists y \, T(On(x, y), s))$
6. $\forall y \forall s \;\; T(Clear(y), s) \Leftrightarrow \neg (\exists x \, T(On(x, y), s))$
7. $\forall x \forall y \forall z \forall s \;\; T(On(x, y), s) \Rightarrow \; T((On(x, z) \Leftrightarrow y = z), s))$

Axiom 5 asserts that an object is on the table if and only if it is not on some other object.
Axiom 6 asserts that an object is clear if and only if there is nothing on top of it. Axiom 7 states that an object can be on at most one other object.

In order to describe a change in our world we introduce the following actions:

$M(c, a, b)$ - *moving* block *c* from block *a* to block *b* (Fig.16-2);

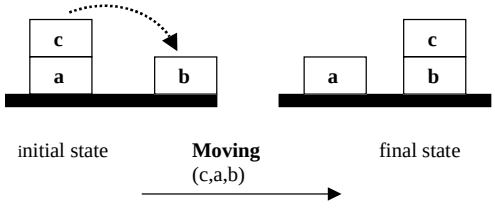


Figure 16-2. Graphical illustration of moving action.

$U(c, a)$ - *unstacking* block *c* from block *a* and placing it on the table (Fig.16-3);

(Copia modificata per una migliore consultazione on-line)

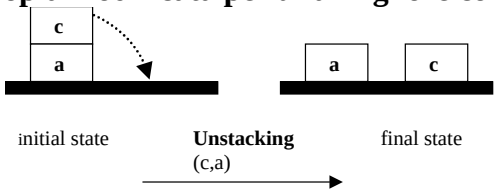


Figure 16-3. Graphical illustration of unstacking action.

$S(b,c)$ - picking up block *b* and *stacking* it on block *c* (Fig.16-4).

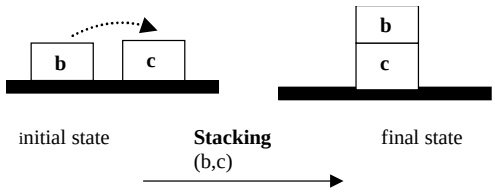


Figure 16-4. Graphical illustration of stacking action.

Finally, there is the action of doing nothing - No.

We can conceptualize the effects of actions in the form of a function *Do* that maps an action (from a set *A* of all actions) and a state (from a set *S* of all states) into the state that results from the execution of the specified action in the specified state, that is,

$$Do: \quad A \times S \rightarrow S .$$

We will write, for example, $Do(M(c,a,b),s_{15})$ which describes results of action $M(c,a,b)$ in state s_{15}.

In the Fig.15-5 effects of actions in the Blocks world are shown.

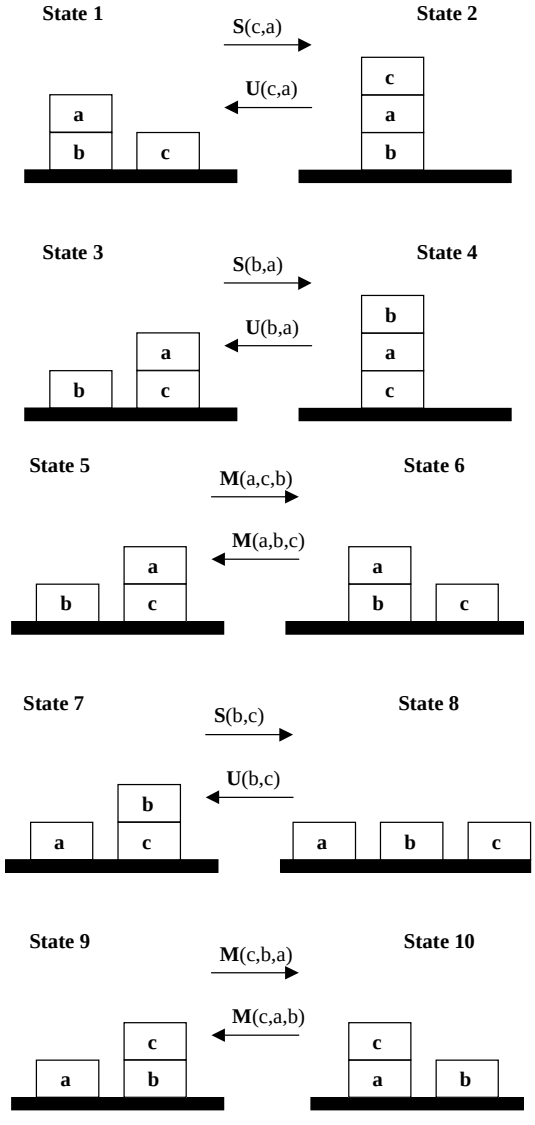


Figure 15-5. Effects of actions in the Blocks World.

(Copia modificata per una migliore consultazione on-line)

The following axioms describe results of our actions.

8. $$T(On(x,y),s)) \wedge T(Clear(x),s)) \wedge T(Clear(z),s) \wedge (x \neq z) \Rightarrow$$ $$T(On(x,z), Do(M(x,y,z),s)) \wedge T(Clear(y), Do(M(x,y,z),s)$$,

where $Do(M(x,y,z),s) = s_1$ is a new state.

This axiom affirms:

if in a state *s* block *x* is on block *y*, both *x* and *z* are clear, and *x* is not the same as *z*, then the action $M(x,y,z)$ will have the effects indicated by the consequent of the implication, that is, in state s_1 (equal $Do(M(x,y,z),s)$) block *x* is on block *z*, and block *y* is clear.

9. $$T(On(x,y),s)) \wedge T(Clear(x),s) \Rightarrow$$ $$T(Table(x), Do(U(x,y),s)) \wedge T(Clear(y), Do(U(x,y),s)$$

This axiom says that

if in a state *s* block *x* is on block *y*, and *x* is clear, then after the action $U(x,y)$, *x* is on the table and *y* is clear.

10. $$T(Table(x),s)) \wedge T(Clear(x),s)) \wedge T(Clear(y),s) \wedge (x \neq y) \Rightarrow$$ $$T(On(x,y), Do(S(x,y),s))$$

The axiom affirms:

if in a state *s* block *x* is on the table, both *x* and *y* are clear, and *x* is not the same as *y*, then after the action $S(x,y)$ *x* will be on *y*.

11. $T(p,s) \Rightarrow T(p, Do(No,s))$

The axiom says that if there is the property *p* in a state *s* then after the action *No* (i.e. do nothing) the property is not changed.

To solve the frame problem for our world we introduce the following *frame axioms* which indicate the *properties that remain unchanged* after each action:

12. $T(Clear(u),s) \wedge (u \neq x) \Rightarrow T(Clear(u), Do(U(x,y),s))$

13. $T(Table(u),s) \wedge (u \neq x) \Rightarrow T(Table(u), Do(U(x,y),s))$

14. $T(On(u,v),s) \wedge (u \neq x) \wedge (v \neq y) \Rightarrow T(On(u,v), Do(U(x,y),s))$.

Axiom 12 says that a block is clear after the *U*-action if it is clear before the action; Axiom 13 states that a block is on the table after the *U*-action if it was on the table before this action.

Axiom 14 affirms if blocks *u* and *v* are not arguments of the *U*-action and block *u* is on the block *v* then after this action the relation $On(u,v)$ is conserved.

Analogous axioms can be written for other actions.

Remark 1. So far we consider only individual actions. *How to describe multiple actions*?

For this case we can introduce a finite set of actions as an *actions chain*. The overall result of executing the actions chain (beginning from some initial state) is the state obtained by executing actions in order. The first action is performed in the initial scene, the second action is performed in the state resulting from the first action, and so on.

You can see that by using this simple model we can describe different states of our world and changing in it. But we consider very simple world.

Causal reasoning model

In this model developed by L.A. Stein and L.Morgenstern (1994) [22] reasoning about how things change over time (so-called *causal reasoning*) is explored. The model is based on the extension of the predicate theory, where predicate notation is used to express states with *non-boolean* value. For example, the following expression

$$Holds(t, colour(house, red))$$

means that *colour* (*house*) has the value *red* in the world state with index *t*.

In this model a *point in time* defines a particular *world state*, and the state of the world is changed by *actions*.

Notation $Occurs(t,act)$ means that action *act* occurs in the world with index *t*, and resulting world state is $t+1$.

By the definition

$$Time(Occurs(t,act)) = t\ ,\ Act(Occurs(t,act)) = act\ .$$

To connect world states and actions, the following notation is introduced:

$Causes$ (*preconditions, excluded-actions, actions, effect*).

This means that if *preconditions* hold in a world when *actions* (but not *excluded-actions*) occur, then *effect* will hold (or occur) in the resulting world state. Preconditions can be a set (that is a conjunction), so we may have a multiple preconditions. Multiple consequences (effects) can be represented using several $Causes$.

The example of Causes for block's world is as follows:

$$Causes((Clear(a), Clear(b), (), (Move(a,b), On(a,b))$$

In considered models of actions exactly one action occurs in each world state and that action is known. But there may be more complex situation, when a few actions occur concurrently, or when an action is unknown, or when no change is performed (as in the case of *wait* action).

Further improvement of actions models results in development more complicated models for description actions and changing in complex worlds. A particular attention in action models design is given to the problem of interrelation between spatio-temporal and action logics. A lot of researchers investigated this problem [7,15,23,24,25,26]. Consider main ideas of some advanced action models.

2. Advanced Action Models

2.1 Dynamic logic approach to reasoning about action and change

Dynamic logic approach to reasoning about action and change [23] is based on Propositional Dynamic Logic *(PDL) developed by V. Pratt [24]. PDL is a modal logic of actions and computer programs.*

(Copia modificata per una migliore consultazione on-line)

Basic concepts of PDL

It is assumed that given a set $S = \{s, t, ..., s', s'', ...\}$ of possible total states of the world. A *proposition* can be identified with the set of states in which it is true. An action α is a binary relation R_α on *S*, that is, a set of ordered pairs $\langle s, t \rangle$, where *s* is the initial state of some execution of α and *t* is the final state. Of course, the final state need not be uniquely determined. Thus the modeling of actions is *semantically non-deterministic*.
Remark 2. Semantical non-determinism concerns the fact that a (halting9 execution of an action v may end up in different states. One reason for this is that there exists always a variety of different ways to perform an action.

An operator [α] is associated with each action α. The expression $[\alpha]A$ means "after every terminating execution of α, A *is true.*

PDL provides a powerful language for describing different compounds actions:
$\alpha ; \beta$ - a compound action consisting of doing first α and then β;
$\alpha + \beta$ - an action consisting of doing α or β non-deterministically;
α^* - the action consisting of doing α some finite number of times;
A? - the action consisting of verifying that *A* holds;
λ - doing nothing (stationary waiting).

PDL Language

Let $\mathrm{P}_0 = \{p_1, p_2, ..., p, q, r, ...\}$ and $\mathrm{A}_0 = \{a_1, a_2, ..., a, b, c, ...\}$ be infinite sets of propositional variables and action variables, respectively. We will use $A, A_1, B...$ to denote formulas and $\alpha, \alpha_1, \beta, ...$ to denote arbitrary terms (denoting actions). So, the following sets are introduced:

Propositional variables: $p \in \mathrm{P}_0$;

Action variables: $a \in \mathrm{A}_0$;

Formulas: $A \in Lang(PDL)$, where *Lang*(*PDL*) is a language of PDL;

Action terms: $\alpha \in \mathrm{A}$;

$A ::= p \mid \neg A \mid A_1 \vee A_2 \mid [\alpha]A$;

$\alpha ::= a \mid \alpha_1; \alpha_2 \mid \alpha_1 + \alpha_2 \mid \alpha^* \mid A?$

PDL Semantics

A *Model M* is a structure consisting of the following triple: $\langle S, \{R_\alpha : \alpha \in \mathrm{A}\}, v \rangle$, where v is a function mapping $\mathrm{P}_0 \to Pow(S)$, where $Pow(S)$ is the powerset of *S*.
A formula *A* is said to be *valid* in a model *M* iff (if and only if) *A* is true at all states *s* in *M*.

PDL Application

PDL-based approach has been used for the task of planning: synthesis of a plan of actions to achieve some specified goal or goals.

Remark 3. Constructing a logical system, we must keep in mind two important properties: *correctness* and *completeness*. A logic system *L* is correct with respect to a class of interpretation models *M* iff all theorems of *L* are valid in all models in *M*. Logic *L* is (weakly) complete with respect to a class of models *M* iff all formulae which are valid in *M* are theorems of *L*.
It is shown that PDL is correct and weakly complete with respect to the class *M* of standard models [23].

2.2 Pseudo-Physical Model of Actions

The *Pseudo-Physical Logical Model of Actions* is the logical deductive system that allows to imitate human reasoning about actions and spatial scenes and their change in a real environment space [7].
This model has been developed for an intelligent mobile robot decision making and robot's behavior planning [15]. Consider main ideas of the model and example of application.
In order to formalize human reasoning about actions and spatial situations let us include objects, actions and spatial scenes as objects in the universe of discourse. Then we may use their names in statements describing some real world situations.
Consider following sorts of terms:

- a set of world states $W = \{w_1, w_2, ..., w_n, ...\}$;
- a finite set of objects names $O = \{o_1, o_2, ..., o_i\}$;
- a finite set of actors (or subjects, or agents) names $\hat{S} = \{S, S_1, S_2, ..., S_i\}$;
- a set of spatial points $\mathrm{P}^{3D} = \{p_1, p_2, ..., p_k, ...\}$, where each p_k is a point in a 3D space with a given coordinate system;
- a set of localizations (places) names $L = \{l_1, l_2, ..., l_m\}$;
- a set of actions modificators $M = \{m_1, m_2, ..., m_j\}$;
- a set of real numbers $N = \{n, n_1, ..., n_k, ...\}$;
- a set of fuzzy numbers $N = \{(\tilde{n}, \mu_{\tilde{n}}), ..., (\tilde{n}_k, \mu_{\tilde{n}_k}), ...\}$, where $\tilde{n}_k$ is a fuzzy number described by membership function $\mu_{\tilde{n}_k}$.

Predicates

(Copia modificata per una migliore consultazione on-line)

Consider predicates describing a set of basic robot's actions.
Basic Actions
Basic actions consists of the following classes:

$$A = \{a_1, a_2, ..., a_n\} = A_{moving} \cup A_{manipulation} \cup A_{special},$$

where A_{moving} is the set of *Motion Actions* , $A_{manipulation}$ is the set *Manipulation Actions*, and $A_{special}$ is the set of *Special operations.*

Generally speaking, any action a_i may be represented by *n*-ary predicate

$a_i(\arg_1, \arg_2, ..., \arg_n)$, where $\arg_i, i = 1,...,n$ are arguments. Arguments variables are used for defining semantics of a given action. They are called *semantic cases* of the action.
There are following *semantic cases*:

- an actor (or subject, or agent) of an action (for example,"Robot1 is moving in a corridor") ;
- a direct object of an action, i.e. the action is directed toward the object (for example,"Robot 1 pushes a knob of a door ");
- an instrument of an action, i..e. the action is realized by means of the instrument (for example," Robot1 push a door's knob by a manipulator" , here the instrument is the robot's manipulator) ;
- a recipient of an action, i.e. the action is directed to the recipient (for example, " Robot1 gives the keys to Robot 2", here Robot 2 is the recipient);
- a modificator of an action (for example, it may be different types of moving (slowly, quickly, etc.) or frequency of the action (often, some times, etc.)) ;
- a place where an action is performed (for example, "Robot 1 puts a book on a table", here the place is a room);
- a *time* of action execution.

Remark 4. In the action description some semantic cases may be lost.
Consider now basic actions as follows.

Motion Actions:

$Move_1(S)$: *S* (agent of action) *is moving* (according with a current direction of moving and velocity);

$Move_2(S, p_1, p_2)$: *S moves from point* p_1 (or *from a place* or *from an object*) *to point* p_2 (or *to a place* or (*nearly*) *to an object*);

$Move_3(S, l_k)$: *S moves inside place* l_k ;

$Move_4(S, l_k)$: *S moves outside place* l_k , $l_k \in L$;

$Move_5(S, m_1)$: *S turns right* ($m_1 \in M$) (or *left, forward, back,* according with a current direction of moving);

$Move_6(S, n_1)$: *S moves on* some (fuzzy) *distance* $n_1 \in N$ or $\tilde{n}_1 \in \tilde{N}$;

$Stop(S)$: *S* is *stopped.*

Manipulation Actions:

$Grasp(S, o_k)$: *S grasps object* o_k (including different types of grasping: strongly grasp, weakly grasp and so on);

$Hold(S, o_k)$: *S holds object* o_k ;

$Move\_obj(S, o_k)$: *S moves object* o_k (may be with different types of moving velocity);

$Put_1(S, o_1, A)$: *S puts object* o_1 *at point A* (or *place* l_k);

The following set of actions connected with location of two objects according to some spatial relation:

$Put_2(S, o_1, o_2)$: *S puts one object* o_1 *on another object* o_2 ;

Put_2 - *put into* (Put_3 - *under,* Put_4 - *on,* Put_5 - *right from* , Put_6 - *left from,* Put_7 - *in front of,* Put_8 - *behind of,* Put_9 - *in between, far from,* Put_{10} -*near from,* and so on) ;

$Take(S, o_1)$: *S takes object* o_1

$Push(S, o_1)$: *S pushes object* o_1 ;

$Throw(S, o_1)$: *S throws object* o_1 , etc.

A set *of special operations* consists of the following actions:

$Open(s, o_k)$: *S opens* a door;

$Close(S, o_k)$: *S closes* a door;

$Bring(S, o_k), o_k \in O$;

$Clean(S, o_k)$: *S cleans* (some surface, and etc.).

Special Passive Action:

$Wait$ - *wait* (conditions of waiting).

Remark 5. We may consider also a set of more complex action called *scripts*. They consist of an actions chain and describe more complex action in terms of basic actions, for example, the action "bring a correspondence " can be described as the following actions chain: "move to office room", "take a correspondence from the box", "move to the laboratory".

(Copia modificata per una migliore consultazione on-line)

Causal Predicates

Introduce following three types of causal predicates:

- $Cause_1(p, a_k)$: if some property *p*(called *preconditions*) hold in a world then *action* a_k can be performed. Preconditions can be a set (that is a conjunction) of actions or a *WFF* (well-formed formula); For example, in order to perform the action "take a book on a table", the necessary action " move to the table" must be performed;
- $Cause_2(a_k, p)$: if the action a_k occurred in a world then *p*(called *effects*) will hold (or occur) in the resulting world state. For example, the effect (result) of the action "take a book " is described as " hold the book in the hand".
- $Cause(a_1, a_2)$: the realization of action a_1 causes the realization of the action a_2 ; this is not necessary demand, the action a_2 can be or can not be performed. For example "Robot took the key"(action a_1) causes the action "Robot opens a room "(action a_2), which may be not performed;
- $Phys - Cause(a_1, a_2)$: action a_2 follows the action a_1. For example, "a glass cap is fall down and destroyed".

Motivation Predicates

- $Goal(S, a_k, p)$, which means that the actor *S* performs the action a_k in order to achieve some property *p* (called a *goal*) described by some action or *WFF*; For example, "Robot 1 took the key" in order "to open the room 2";
- $reason(S, a_k, p)$, which means that the actor *S* performs the action a_k because *S* wants some property *p* (called a *reason* (or *intent*)); *p* may be described by some action or *WFF*. For example, "Robot 1 goes to a shop (action) because he wants to buy a toy (intent) for Robot 2".

Description of properties and actions execution

In order to describe that some property *p* (described as a well-formed formula *WFF*) is true in a given situation *w* we introduce a relation $T(p, w)$.

For example, $T((o_1 R o_2), w_1)$, where $R \in \{set - of - spatial - relations\}$ means that the property $(o_1 R o_2)$ is true in particular state w_1.

We will describe the effects of actions in the form of a function *Do* that maps an action (from set *A* of all actions) and a state (from set *W* of all states) into the state that results from the execution of the specified action in the specified state, that is,

$$Do: \quad A \times W \to W .$$

We will write, for example, $w_2 = Do(a_k, w_1)$ which describes a resulting state w_2 of a world after the performance of the action a_k in the state w_1.

Each state w_i of a world can be described by temporal point t_i in a given temporal scale, and each process of an action's execution can be considered as some event. So, in accordance with a temporal model we may consider interval or instant events (or actions) and different temporal relationships between them.

The pseudo-physical actions model is the sorted first order predicate theory where the set of predicates is represented by actions and relations mentioned above, and the set of axioms, describing semantic properties of actions in time and space, are developed.

Consider examples of axioms schemes applied for the intelligent robot's behavior simulation.

Remark 6. In actions axiom schemes, we will use temporal and spatial relations written in the form of predicate with given number of arguments. For example, instead of ($o_1 R o_2$) expression (used in the spatial model) we will use $R(o_1, o_2)$ expression.

Axiom schemes

I.

1. $\forall a Do(a, w) \Rightarrow R_{begin}(a, t_1) \wedge R_{end}(a, t_2)$, $a \in A, w \in W$,

where $a \in A, w \in W$, R_{begin}, R_{end} are temporal relations defined earlier, t_1, t_2 are points in a given temporal scale with a given measure unit.

The axiom affirms that any action's execution has beginning and ending points in a time.

II. Frame axioms

1. $T(R(u,v), w) \wedge (u \neq x) \wedge (v \neq y) \Rightarrow T(R(u,v), Do(a(S,x,y), w))$,

where $u, v, x, y \in O$, and $R \in \{set - of - spatial - relations\}$ defined earlier.

This axiom says that if objects *u* and *v* are not arguments of the *a*-action and object *u* is connected with object *v* by some spatial relation *R* in the spatial scene *w*, then after this action the relation $R(u,v)$ is conserved in new spatial scene $w_2 = Do(a, w)$.

2. $T((R(u,v) \wedge (u = x), w) \Rightarrow T(\neg R(u,v), Do((Put_i(S,x,y), w))$,

where $u, v, x, y \in O$, $R \in \{set - of - spatial - relations\}$, and

Put_i one of $\{Put_2, Put_3, ..., Put_{10}\}$ actions.

Axiom 2 says that if the object *u* is one of the arguments of *Put*-action and object *u* is connected with the object *v* by some spatial relation *R* in some spatial scene *w*, then after the *Put*-action the relation $R(u,v)$ is not conserved in a new spatial scene $w_1 = Do(Put_i, w)$.

Consider some partial cases of the axiom 2.

3. $T((R_2(u,v) \wedge (u = x), w) \Rightarrow T(Free(u), Do((Put_i(S,x,y), w))$,

where $R_2(u,v)$ is the spatial relation "to be on".

4. $T((R_2(u,v), w) \Rightarrow T(Free(u), Do((Put_{10}(S,u,v), w))$,

(Copia modificata per una migliore consultazione on-line)

5. $T((R(S,o_1),w) \Rightarrow T(\neg R(S,o_1), Do((Move_i(S,x,y),w))$,
where $Move_i$ one of $\{Move_1, Move_2,..., Move_5\}$ actions.

Axiom 5 says that if the agent *S* of the *Move*-action and the object o_1 are connected by some spatial relation *R* in the spatial scene *w*, then after the *Move*-action the relation $R(S,o_1)$ is not conserved in a new spatial scene $w_1 = Do(Move_i, w)$.

Pseudo-Physical Action Model Application

This model has been used for the design of intelligent robot's behavior simulation system including natural language understanding, spatial scene analysis, actions planning and execution (see next Lecture 17).

Soundness and *completeness properties of spatio-temporal and action models*

We considered main problems of external world modeling based on spatio-temporal and actions logics. Finally, let us talk about their deductive properties.

Let *F* be a finite set of relationships. We say that *F* is *consistent* if there exists a spatial scene *s* that satisfies all the relationships in *F*. For example, the set { $(o_1 R_{left} o_2), (o_2 R_{left} o_3)$ } is consistent, while set { $(o_1 R_{on} o_2), (o_1 R_{left} o_2)$ }is inconsistent. We say that a relationship *R* is implied by *F* if every spatial scene that satisfies all the relationships in *F* also satisfies the relationship *R*. For example, the set of relations { $(o_1 R_{left} o_2), (o_2 R_{left} o_3)$ } implies the relation $(o_1 R_{left} o_3)$.

Consider a set of axiom schemes of a spatial logic in the following form:

$(o_1 R_1 o_2), (o_2 R_2 o_3) \Rightarrow (o_1 R o_3)$.

We say that relation *R* is *deducible in one step* from a set of relationships *F* using axiom schemes, if *R* contains in the right part of some axiom and each relation in the left part of the axiom is contained in *F*.

We say that relation *R* is *deducible in k steps* from the set of relationships *F* using axiom schemes, if there exists a finite sequence of relations $R_1, R_2, ...R_k$ ending with *R*, i.e. $R_k = R$, such that R_1 is deducible in one step from *F* using one of the axiom schemes, and for each $i = 2,3,...,k$ R_i is deducible in one step from the set $F \bigcup \{R_1, ...R_{i-1}\}$.

We say that *R* is deducible from *F* in zero steps if $R \in F$.

Definition 1.
Relation *R* is *deducible* from *F* using the axiom schemes if *R* is deducible from *F* in zero or more steps.
Consider a set of axiom schemes *A* and define *soundness* and *completeness* of *A*.

Definition 2.
An axiom scheme in *A* is said to be *sound* if every spatial scene that satisfies all the relationships in the left part of the axiom scheme also satisfies the relationship given by the right part of the axiom scheme.
Definition 3.
The set of axiom schemes is *sound* if every axiom scheme in *A* is sound.
Definition 4.
The set of axiom schemes *A* is *complete* if it satisfies the following property: for every consistent set *F* of relationships, every relation implied by *F* is deducible from *F* using the axiom schemes in *A*.

The main questions of any developed logical system are the *soundness* and *completeness* questions. Proofs of soundness and completeness theorems for the case of temporal logic axioms can be found in [7] and for the case of three dimensional objects locations description in [27]. In the case of action logical models, we may say that the set of axioms is complete only for some simple world models, and may be modified depending on a problem area and a given task of modeling.

Final Discussions

We considered the problems of simulation of human spatio-temporal and action reasoning by using logical approach and discussed physical and logical aspects of human space-time perception .
For the complete logical description of a space-time continuum representing real environments, we introduced the pseudo-physical spatio-temporal-action logics (PSTAL). Moreover, we considered them as the physical model representation of an external real world and used them for intelligent agent's (like robots) behavior simulation.

The main result of our investigation follows. We confirm that the use of PSTAL as *the model representation of an external world* is the necessary and sufficient conditions for the correct description of robot's artificial life in space-time continuums. Another words it is possible to describe by the means of a logical model complex behavior of intelligent robot.

Using pseudo-physical spatio-temporal logics we can introduce the logical model of physical space-time continuum for robot's artificial life. Using action logic we can introduce the logical description of different technological operations in a physical space. Using different components of PSTAL we can simulate human qualitative reasoning and rational intelligent behavior. Thus, we can develop intelligent robots with intelligent capabilities including such components as: NL-based communication, task-level planning, goal-oriented behavior, creation of cognitive maps of environments scenes, navigation in an environment, etc.

Lecture N 17 Goal-oriented Behavior and Planning Problems

Example: Behavior Simulation of Intelligent Service Robot

The *ability to plan ahead* is a key aspect of intelligent behavior. By knowing effects of our actions and exploiting that knowledge, we can achieve goals while avoiding danger and economizing on resources. For example, we can save time and energy in shopping if we think in advance about what we need to buy and plan an appropriate route. People make a lot of plans, sometimes for themselves, sometimes for other people, sometimes for machines.

Let us return to the first lecture and refresh one of five laws of intelligent behavior – *the principle of the rationality*. This law says follows: if an agent has knowledge that one of its actions lead to one of its goals, then the agent will select that action.

The given formulation of the principle of rationality poses the question of how to characterize more precisely this rational behavior. We may say that *the rational behavior of an intelligent agent is a goal-oriented or planned behavior*.
In a broad sense, *planning* is *designing the behavior of some being that acts, either an individual, a group of individuals, or an organization*. The output of planning process is some kind of steps for behavior, which we call a *plan*.

(Copia modificata per una migliore consultazione on-line)

Consider classical example of rational, goal-oriented behavior known as “a monkey and bananas”. The problem itself is the following. A monkey is situated in a room containing a couch, a ladder, and a bunch of bananas (Fig.17-1).

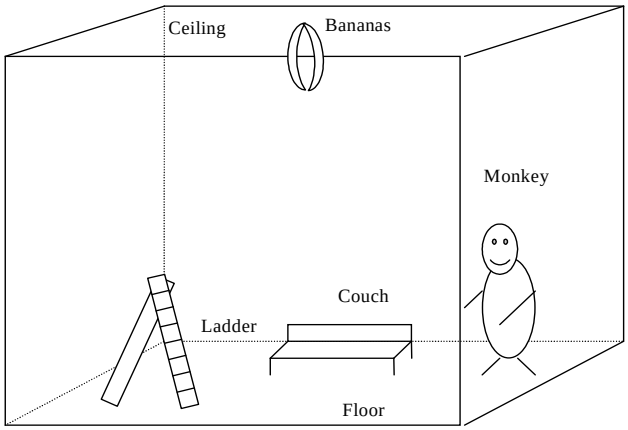


Figure 17-1. The “monkey and bananas” situation.

The objects are distributed on a floor and bananas hang from the ceiling. Each object has its coordinates. The height of monkey is too small in order to grasp bananas directly by hand. What must to do the monkey in order to eat bananas? He goes to the ladder, brings it under a bunch of bananas, climbs up and takes bananas.
So, the monkey shows us his rational behavior, that is, the ability to find a plan of actions in order to achieve a goal (“to eat bananas”) in the current situation.
The “Monkey and bananas” decision making process can be, for example, consist from the following steps:
In order to eat bananas, I must take bananas;
In order to take bananas, I must be near bananas;
I cannot directly come to a place near bananas, then I try to find some instrument to do so;
Oh, I see a ladder. I’ll take a ladder and climb up to the bananas.

We may write a general heuristic rule for planning as follows:

IF (<*a given situation*>) and (<*a defined goal*>), THEN (<*a sequence of actions*>).

For our example with the monkey we may write the following rules:

1) IF (monkey is at position *x* on the floor) and (he does not hold anything) and (bananas are at position *y* on the ceiling) and (a ladder is at position *z*) and (*z* is different from *x* and *y*), THEN (monkey moves to position *z* in order to take the ladder) and
2) IF (monkey is at position *z* on the floor) and (he holds the ladder), THEN (he moves with the ladder to position *y* in order to climb onto the ladder and take bananas).

The plan of monkey’s actions is following:
Go to the ladder; grasp the ladder; go to the place under bananas holding ladder; drop the ladder; climb onto the ladder; grasp bananas and eat.

How can we automate planning?

In planning we start with a set of desired properties (a goal) and try to construct a plan that results in a state with the desired properties (a goal state).

There are a wide variety of planning problems, differentiated by the types of their inputs and outputs. For example, the input to the behavior designer (or planner) can be a description of an initial block’s world state as follows: “there are three blocks on the table, one red, one white and one blue”, and a goal state description: “the blue block is on the red block”. The output of planner is an order of actions (like “move”, “take” and “put”) that can be performed in order to transform the initial world state to the goal state. Or, for example, the input to the behavior designer can be a set of actions that must be carried out, and the output is an order in which to carry them out. This is a typical transportation scheduling problem.

Typically, planning problems get more and more difficult as more flexible inputs are allowed and fewer constraints on the output are required. It is very difficult to solve these problems very quickly (in a real time), and most planning algorithms are NP-complete (that is, the complexity of algorithms is evaluated by a non-polynomial function of time).

Classical AI planning

Planning research started in the 1960s was mainly based on an application of two standard symbolic AI techniques: search and theorem proving.

The classical AI approach to planning problems is to start from specifications of the goal, initial state of a problem and effects of actions, then try to infer a string of actions that transform initial problem state to the goal state (Fig.17- 2).

The planner in this case takes as input data initial state (of a given problem) and goal descriptions, and uses a knowledge base containing information about a set of available actions. The output of the planner is a plan of actions that when executed in a state i satisfying the initial state description produces a state g satisfying the goal description.

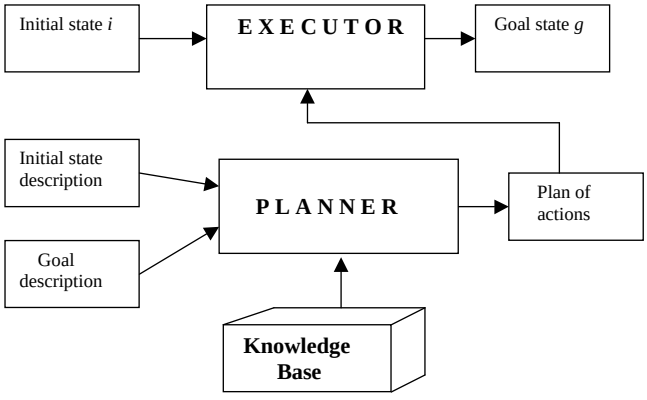


Figure 17–2. Planning process scheme.

(Copia modificata per una migliore consultazione on-line)

So, in AI, planning procedure is considered as *a searching of a solution in a problem state space for the transformation an initial problem state to the goal state.*

Remark 1. Many problems can be solved by applying a sequence of transformations starting from an initial null solution. At each step, there is a choice of which transformation to apply in order to find a solution satisfying a given goal. This is typical *problem solving* procedure developed in AI. If we think of transformations as agent actions, then problem solving is a planning procedure.

Let us formalize planning problem as follow [21].

Planning Problem

A planning problem consists of:

- initial state description i ,
- goal state description g ,
- the set of available actions *A*,
- a set of actions chains $\gamma \in \Gamma$, where each action $\in A$,
- database Ω that includes sentences describing the initial state, and
- the knowledge base K including actions axioms.

In order to describe a goal, we introduce special unary predicate $Goal(t)$.

Definition of a Plan:

An action chain γ is a plan for a planning problem if it satisfies the following conditions:

(1) $\gamma \in \Gamma$;

(2) we must be able to prove from Ω that γ achieves a state satisfying $Goal$ when executed in state i , that is $\Omega \Rightarrow Goal(g) = Goal(Do(\gamma, i))$.

Consider a few examples of classical planning.

Example 1. Let we have the following initial state of the Blocks World:
"block *c* is on block *a* and blocks *a* and *b* are both on the table" (see Fig.17-3).

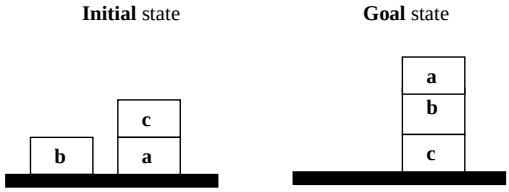


Figure 17-3. Initial and goal world state representation.

If we call this state as S_1 , then we can consider the following initial state description:

$$\Omega_1 = T(Clear(a), s_1) \wedge T(On(c,a), s_1) \wedge T(Table(a), s_1) \wedge T(Clear(b), s_1) \wedge T(Table(b), s_1) .$$

Remark 2. There is no requirement that the initial state description be complete. If some information is left out, it simply means that more than one state may satisfy the description and the planner must be sure to find a plan that is guaranteed to work in any of these states.

As we said earlier, a goal is any desired state. For example, we may desire the Blocks World state in which all the blocks are on the table, or we may desire that there be a stack of three blocks: "block *a* is on block *b* and block *b* is on block *c*".

In our case, the desired state mentioned above can be expressed as follows:

$$T(On(a,b), g) \wedge T(On(b,c), g) \Leftrightarrow Goal(g) .$$

Let us repeat here actions axioms needed for planning in our example.

$$T(On(x,y), s)) \wedge T(Clear(x), s) \Rightarrow$$
$$T(Table(x), Do(U(x,y), s)) \wedge T(Clear(y), Do(U(x,y), s) ;$$
$$T(Table(x), s)) \wedge T(Clear(x), s)) \wedge T(Clear(y), s) \wedge (x \neq y) \Rightarrow$$
$$T(On(x,y), Do(S(x,y), s)) .$$

We introduce also sentences for expressing actions chain $[b,c]; b,c \in A$ as follows:

$$Do([b,c], s) = Do(c, Do(b, s))$$

The actions chain $\gamma = [U(c,a), S(b,c), S(a,b)]$ is a plan to solve the problem given in this example. It is clearly a member of Γ . Using the information in Ω_1 , we can easily prove that the plan works, i.e., we can derive the following sentence:

$$Goal(Do[U(c,a), S(b,c), S(a,b)], s_1) = T(On(a,b), g) \wedge T(On(b,c), g) .$$

Planning Direction

We have seen that planning procedure is a searching (or inference) in a state space defined by initial and goal state descriptions. In previous lectures we discussed two main directions of such kind of searching: *forward* (from initial facts to the goal) and *backward* (from the goal to the initial facts).

One of the key determinants of planning efficiency is a planning direction. In some cases, it is better to plan forward from initial state. In other cases, it is better to work backward from the goal. In yet other cases, a mixed strategy is best.

Example 2. Consider task 2: find a plan of actions for the following *Block World* transformation:

The initial *Block World* state: "block a is on the block b and block b is on block c ;

(Copia modificata per una migliore consultazione on-line)

The goal state: "block a is on the Table" (Fig.17-4).

If we call initial state as S_1, then we can consider the following initial state description:

$$\Omega_1 = T(Clear(a), s_1) \wedge T(On(a,b), s_1) \wedge T(On(b,c), s_1) \wedge T(Table(c), s_1).$$

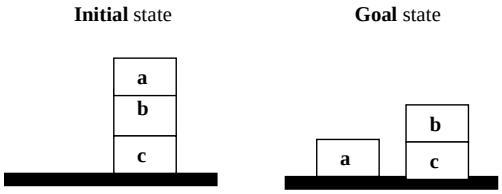


Figure 17-4. Initial and goal state representation in example 2.

We define our goal by the following goal predicate:

$$T(Table(a), g) \Leftrightarrow Goal(g).$$

Backward planning
Consider *backward planning* procedure ideas.
We begin to construct a plan in backward direction, from a goal description. We will find in the knowledge base an action axiom, where right part of the axiom contains properties given in the goal, in our case $T(Table(a), g)$. This is the following axiom:

$$T(On(x, y), s)) \wedge T(Clear(x), s) \Rightarrow$$
$$T(Table(x), Do(U(x, y), s)) \wedge T(Clear(y), Do(U(x, y), s)$$

Variable *x* in the description of the axiom takes the value *aa* and we have the following expression:

$$T(On(a, y), s)) \wedge T(Clear(a), s) \Rightarrow$$
$$T(Table(a), Do(U(a, y), s)) \wedge T(Clear(y), Do(U(a, y), s)$$

We will use this description for the *U* operator to reduce the goal to the subgoal as follows:

$$T(On(a, y), s)) \wedge T(Clear(a), s)? \qquad (17\text{-}1)$$

If we can find a block *y* in state S_1 such that $On(a, y)\ \ and\ \ Clear(a)$, then executing $U(a, y)$ in that state will achieve the goal.

By matching facts in Ω_1 and condition (17-1) you can see that condition (17-1) is true for $y = b$. Hence, the following action $U(a,b)$ is the plan for the task 2.

Finally, we can formulate backward planning as follows: start with the goal statement, reduce this goal to subgoals, and so on until we get to conditions on the initial state.
Remark 3.You can see that the key mechanism in the planning procedure is the proving of the existence of a correct plan based on the actions axioms. We discussed main ideas of the backward planning. The general algorithm for backward planning was developed by Green and based on the resolution method of inference [21]. *Green's method* is sound in that it produces only correct plans. It is also complete in that it is guaranteed to produce a correct plan whenever one exists. Unfortunately, Green's method like all planning procedures can be extremely inefficient.

Forward planning
Consider *forward planning* procedure ideas.
Example 3. Consider following task 3: find a plan of actions for the following *Block World* transformation:
the initial *Block World* state: "block a is on the block b and block b is on block c ;
the goal state: "block b is on the Table" (Fig.17-5).

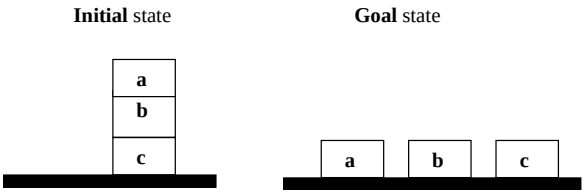


Figure 17-5. Initial and goal state representation in example 2.

We write the initial state description as the following DataBase:

1. $T(Clear(a), s_1)$,
2. $T(On(b,c), s_1)$,
3. $T(On(a,b), s_1)$,
4. $T(Table(c), s_1)$.

We write our goal as $T(Table(b), g) \Leftrightarrow Goal(g)$.

We can formulate ideas of *forward planning* as follows: start with the facts in initials state description, and by using actions axiom infer a set of new facts. Check contains the goal statement in the inferred set of facts. If no, continue the inference process with the new facts as input until the solution is found or a work time is over.

So, let us begin to apply axioms connected with one of chosen action, for example $U(x, y)$:

$$T(On(x, y), s)) \wedge T(Clear(x), s) \Rightarrow$$
$$T(Table(x), Do(U(x, y), s)) \wedge T(Clear(y), Do(U(x, y), s)$$

and

$T(On(u,v),s) \wedge (u \neq x) \wedge (v \neq y) \Rightarrow T(On(u,v),Do(U(x,y),s))$

(frame axiom for $U(x,y)$) to a given input facts containing in a DataBase.

Beginning from facts 1-4 and using the operator description and frame axiom for the $U(a,b)$ action, we can derive the following new facts:

6. $T(Clear(a),Do(U(a,b),s_1))$,
7. $T(Table(a),Do(U(a,b),s_1))$;
8. $T(Clear(b),Do(U(a,b),s_1))$;
9. $T(On(b,c),Do(U(a,b),s_1))$;
10. $T(Table(c),Do(U(a,b),s_1))$.

Beginning from facts 6-10 and using the operator description and frame axiom for the $U(b,c)$ action, we can derive the following new facts:

11. $T(Clear(a),Do(U(b,c),Do(U(a,b),s_1)))$;
12. $T(Table(a),Do(U(b,c),Do(U(a,b),s_1)))$;
13. $T(Clear(b),Do(U(b,c),Do(U(a,b),s_1)))$;
14. $T(Table(b),Do(U(b,c),Do(U(a,b),s_1)))$;
15. $T(Clear(c),Do(U(b,c),Do(U(a,b),s_1)))$;
16. $T(Table(c),Do(U(b,c),Do(U(a,b),s_1)))$.

By matching the goal statement $T(Table(b),g)$ with the inferred set of facts, we find that the goal statement is contained in the inferred fact 14, hence the solution of planning task 3 is as follows:

$g = Do(U(b,c),Do(U(a,b),s_1))$ and $\gamma = [U(b,c),U(a,b)]$.

Remark 4. A planning problem can be represented functionally as a graph where nodes represent situations (states in the state space of all possible situations) and the arcs represent actions that cause changes in the situations (Fig.17-6).

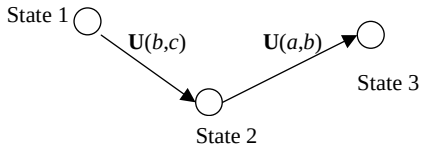


Figure 17-6. Graph-fragment of a state space representation in planning.

First planning systems in Artificial Intelligence research were GPS (the General Problem Solving) developed by A. Newell in 1969, and STRIPS (STanford Research Institute Problem Solver) developed by R.Fikes and N.Nilsson in 1972 [28].
GPS designers try to develop planning system capable in principle to solve any problem. Planning was considered as searching (or inference) from any initial state to final state (goal) applying a sequence of transformations starting from an initial state. Transformations usually called operators. But the description of a search problem has been quite abstract.
STRIPS problem solver will be applied to plan the Shakey robot behavior in a simple block's world.
First planning systems were based on the idea of *keeping the search space close to situation space*. But the field of planning shifted away from this idea.

Example 4. Consider the following planning task for robot's behavior simulation based on the pseudo-physical action model. Let we have the following initial state and goal:
Initial state: " In the room there is a desk located in the center. A note is located on the desk, and a pen is on the note."
Goal : "take the note".
We write the initial state description as the following DataBase:

1. $T(R_1(desk,room),w_0)$;
2. $T(R_2(note,desk),w_0)$;
3. $T(R_2(pen,note),w_0)$, where R_1, R_2 are spatial relations "to be in" and "to be on";
4. $T(R_1(Robot,room),w_0)$.

We write the goal as follows: Goal $(Take(Robot,note))$.

The design of a plan of robot's behavior is begun from the goal like in backward planning procedure. We try to infer states transformations in according with *preconditions* of the goal action.
Let find an axiom describing *preconditions* for the action "take":

1. $\Rightarrow Cause_1((r_1(S,o_1) \wedge Free(o_1)),Take(S,o_1))$;

According to this axiom we write *preconditions* as follows:

$(r_1(Robot,note) \wedge Free(note))$?

Let us check is it possible to do the action $Take(Robot,note)$ in the state w_0?

We try to find facts $(r_1(Robot,note) \wedge Free(note))$ in the Data Base. We check:

(Copia modificata per una migliore consultazione on-line)

1) is the fact $(r_1(Robot, note)$ in w_0 ? Answer is: No;

2) is the fact $Free(note))$ in w_0 ? Answer is: $\neg Free(note))$ (according to a Data Base containing a set of facts in initial state and the following axiom:

2. $T(\neg Free(note), w_0) \Leftrightarrow T(R_2(pen, note), w_0)$, where R_2 is spatial relation "to be on").

So, the facts $(r_1(Robot, note) \wedge Free(note))$ are absent in our initial state.

In this case we will find actions execution of which results in $(r_1(Robot, note) \wedge Free(note))$. Let us try to find an action's effect axiom describing $(r_1(Robot, note)$ as a result. It is as follows:

3. $\Rightarrow Cause_2(Move_2(S, o_2), (r_1(S, o_2))$.

We will use also the following planning axiom:

4. $Cause_2(a, p) \Rightarrow T(p, (Do(a, w))$.

By using axioms 3 and 4 we can generate the following new action:

$$Cause_2(Move_2(Robot, note), (r_1(Robot, note)) \Rightarrow$$
$$T((r_1(Robot, note)), w_1), \ where \ w_1 = Do(Move_2(Robot, note), w_0)$$.

Let us finally check is it possible to do the action $Move_2(Robot, note)$ in the state w_0 ?

There are no *preconditions* for this action and it can be performed in the state w_0.

Describe now the new state w_1 after execution of $Move_2(Robot, note)$.

Applying frame axioms, we can write the following description of w_1:

1) $T(R_1(desk, room), w_1)$;

2) $T(R_2(note, desk), w_1)$;

3) $T(R_2(pen, note), w_1)$, where R_1, R_2 are spatial relations "to be in" and "to be on";

4) $T(Free(pen), w_1)$, (applying axiom 2 in this example);

5) $T(r_1(Robot, note), w_1)$), (the result of new action);

6) $T(r_1(Robot, pen), w_1)$.

The last fact is inferred by using the following spatial axiom:

$r_1(Robot, note) \wedge R_2(pen, note) \Rightarrow r_1(Robot, pen)$.

Now let us find an axiom describing $Free(note))$ as a result. It is the following frame axiom for *Put*-actions:

5. $T((R_2(u, v), w) \Rightarrow T(Free(v), Do((Put_{10}(S, u, v), w))$.

By using this axiom we can generate the following new action:

$$T((R_2(pen, note)), w_1) \Rightarrow$$
$$T(Free(note), w_2), where \ w_2 = Do((Put_{10}(Robot, pen, note), w_1))$$.

Let us check is it possible to do the action $Put_{10}(Robot, pen, note)$ in the state w_1 ?

By using axiom 6

6. $Do(a_1, w) \wedge Cause_1(a_2, a_1) \Rightarrow Do(a_1, Do(a_2, w))$

we can write:

$$Do(Put_{10}(Robot, pen, note), w_1) \wedge Cause_1(Take(Robot, pen), Put_{10}(Robot, pen, note))$$
$$\Rightarrow Do(Put_{10}(Robot, pen, note), Do(Take(Robot, pen), w_1))$$.

Axiom 6 says that before the action $Put_{10}(Robot, pen, note)$, the action $Take(Robot, pen)$ must be executed.

Let us check is it possible to do the action $Take(Robot, pen)$ in the state w_1 ? The answer is "Yes", because *preconditions* for this action $(r_1(Robot, pen) \wedge Free(pen))$ are true in the state w_1.

So, finally we have the following plan of robot's actions which can be executed in the given initial state:

$w_1 = Do(Move_2(Robot, note), w)$;

$w_{2=} Do(Take(Robot, pen), w_1))$;

$w_3 = Do(Put_{10}(Robot, pen, note), w_2)$;

$w_4 = Do(Take(Robot, note), w_3)$.

Remark 5. Unfortunately, the possibility of expressing simple planning problems this classical way has led to the idea that this is the only way to think about planning problems. But in reality this approach to planning is applicable to a small subset of these problems.

Main assumptions of classical planning are:

- a world is passive;
- plans are sequences of actions;
- all actions are well known;

- the outcome of every action is perfectly predictable.

Dynamic World Planning

Many researches dissatisfied with the classical assumptions about planning. One often-cited argument was that real environment is dynamic, i.e., it is changing and imperfectly known. How to find plans, for example, for robots in dynamic environments. Dynamic planning approach attempts to create agents that could react by direct coupling of sensing (stimulus) to effecting (response). Trends which are recently emerged are in the use of multi-level planners to integrate strength of classical planning with those of reaction-based systems.

Now a lot of special-purpose planners are developed [28]. There are a class of a partial-order planning algorithms. In partial-order planners, the plan steps did not kept in a linear order, and thus they have often been referred to as "nonlinear". One special case of planning is a *motion planning*: finding a path from one point to another through a complex region in a metric space. Motion planning includes techniques from computational geometry.

Another special case is *scheduling*: finding a good order to perform a series of known tasks. This problem is very important in many fields of industry. Scheduling includes techniques of combinatorial optimization.

One of the biggest problems in planning is *search control*. Inefficiency in planning can be controlled by making use of stored solutions to previous planning problems. This approach is called *memory-based* or *case-based* planning. But it is hard to show formally that old planning cases will help.

Consider now simulation problems of intelligent service robot behavior.

Intelligent Service Robot Behavior Simulation

Let us discuss main problems of the design of intelligent robot's behavior simulation system including natural language understanding, spatial scene analysis, actions planning and execution.

We will consider the simulation environment like an office building and investigate behavior algorithms in the following artificial life of our robot. The robot works in a complex office building world with many rooms, floors, corridors, elevators, other robots and human beings. In each room there are many objects with different spatial relations. The robot performs some actions and changes spatial relations between objects.

A human operator represents a natural language-based (NL) description of robot's artificial life conditions, which includes the following:

- an initial environment scene description (for example, the human-operator may describe a room in the building, objects in this room, fuzzy spatial relations between them and so on);
- instructions for the robot's work (for example, instructions like "go to the post boxes and bring a correspondence"; or "clean the room during night time"; or "bring the book located on a table in the room 113" and so on).

This description (on NL) is the input to the simulation system. The output of the simulation system consists of two things: 1) a plan of robot's actions according with the instruction given by the human-operator, and 2) visualization of the robot's behavior. We consider the visualization as a temporal chain of 3D graphical representations (GR) of spatial scenes representing results of robot's actions in the given environment. The first scene represents the GR of the initial spatial scene (before the robot's activity), the last one represents the GR of the final scene corresponding to the result of the final action execution (i.e. after robot's activity).

The structure of the service robot's behavior simulation system is represented in the Fig.17-7.

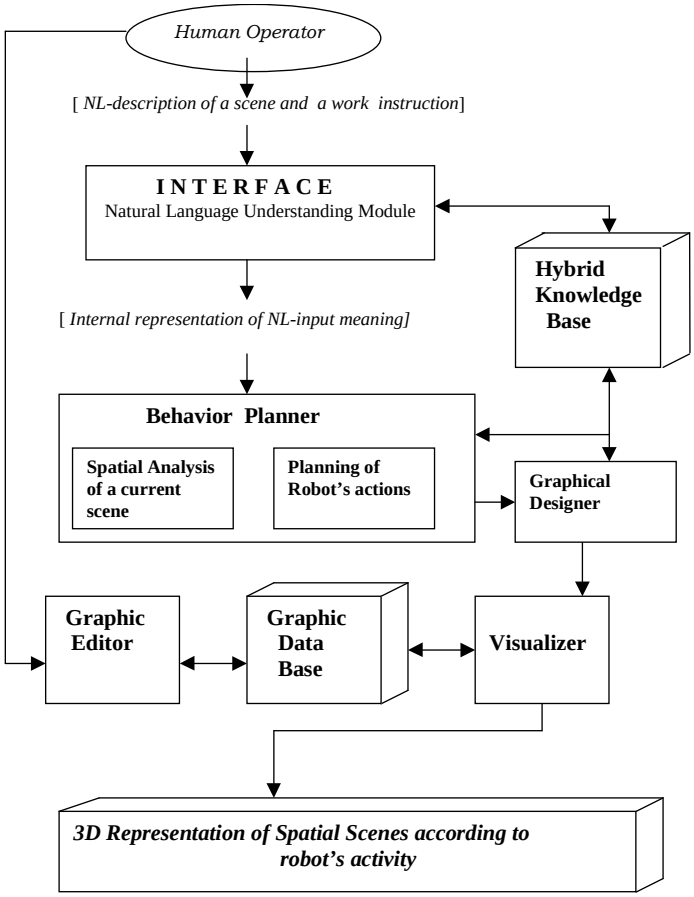


Figure 17-7. Structure of the robot's behavior simulation system.

The system consists of the following modules: 1) NL-interface with NL understanding module (linguistic processor); 2) *Behavior Planner* performing reasoning and planning of robot's actions in the environment scenes; 3) *Graphical Designer* for the constructing graphical representations (GR) of scenes; 4) the tools for mapping and change of simulation environment scenes including *Graphic Editor* and *Visualizer*; 5) *Knowledge* and *Graphic Objects Bases*.

The *Hybrid Knowledge Base* (*KB*) consists of a few knowledge classes. There are *a priori knowledge* and knowledge acquired in the process of problem solving.

All of a priori knowledge can be divided into three classes: 1) Class A contains syntactical knowledge about objects, subjects, actions, relations; 2) Class B contains knowledge about pseudo-physical spatial logic and action logic, which are logical deductive systems describing geometrical and physical properties of a space and actions. These logics are used for the simulation of spatial scenes and dynamic situation; 3) Class C contains knowledge about semantic and pragmatic properties of actions, of objects and spatial relations between them.

(Copia modificata per una migliore consultazione on-line)

Mixed approach based on frames and productions formalisms is used for the knowledge representation. The frame-based part of *KB* describes objects and their properties, relations and actions. The production-based part of *KB* describes spatio-temporal and action logics axioms.

Discuss briefly *the structure of the simulation system.*

NL-input describes a 3D spatial scene (for examples, a room) with objects and spatial relations between them (*initial spatial scene*) and contains work instructions for the robot. The NL-text (in our case, it is on English) inputs to the linguistic processor realizing its transformation to a formal internal representation (*IR*). In the NL-text description, two parts are defined: the description of the initial static spatial scene and the description of robot's actions contained in the work instruction.

The *IR* consists of a set of the frame prototypes representatives of objects, localizations, spatial relations, actions, spatial scenes and scripts. For the NL processing the modification of the Wood's augmented transition networks method is used. (We will talk about NL-processing in the next lecture.)

Behavior Planner performs the following five functions: 1) analysis of the consistency of the current scene from the standpoint of the spatial logic; 2) checking whether it is possible to realize the current action from the qualitative (naïve, or common sense) physics model; 3) construction the robot action's plan based on the spatio-temporal/action logics and the script model; 4) construction the causal outcome of robot's actions; 5) construction the set of spatial scenes, ordered in a time and representing the environment change connected with the robot's actions.

Graphical Designer of the system is used for the construction of internal graphical representation (*GR*) of a current scene. For this task a special method for defining of object's location described by spatial relations is introduced. The method is based on the simulation of human spatial reasoning developed in pseudo-physical spatial model considered earlier. The output from the *Graphical Designer* is the set of object's coordinates in the coordinate system connected with a given spatial scene. This set of object's coordinates is called the *cognitive map* of a given spatial scene.

Vizualizer converts the cognitive map of the given scene into the set of physical coordinates of a display and represents 3D images of spatial scenes on the display screen.

Example 5.Consider the following situation. By the direct human-robot communication line our robot receives from the human-operator the following NL-input including the initial scene and the work instruction description:

" Room 217 is located on the second floor of the building. The desk is in the center of the room. The lamp is on the desk. The wardrobe is located in the left back angle of the room. The sofa is right and not far from the wardrobe. The chair is behind of the desk and close to it. Take the lamp and put it on the wardrobe."

Remark 6. We propose that the initial location of the robot is known.

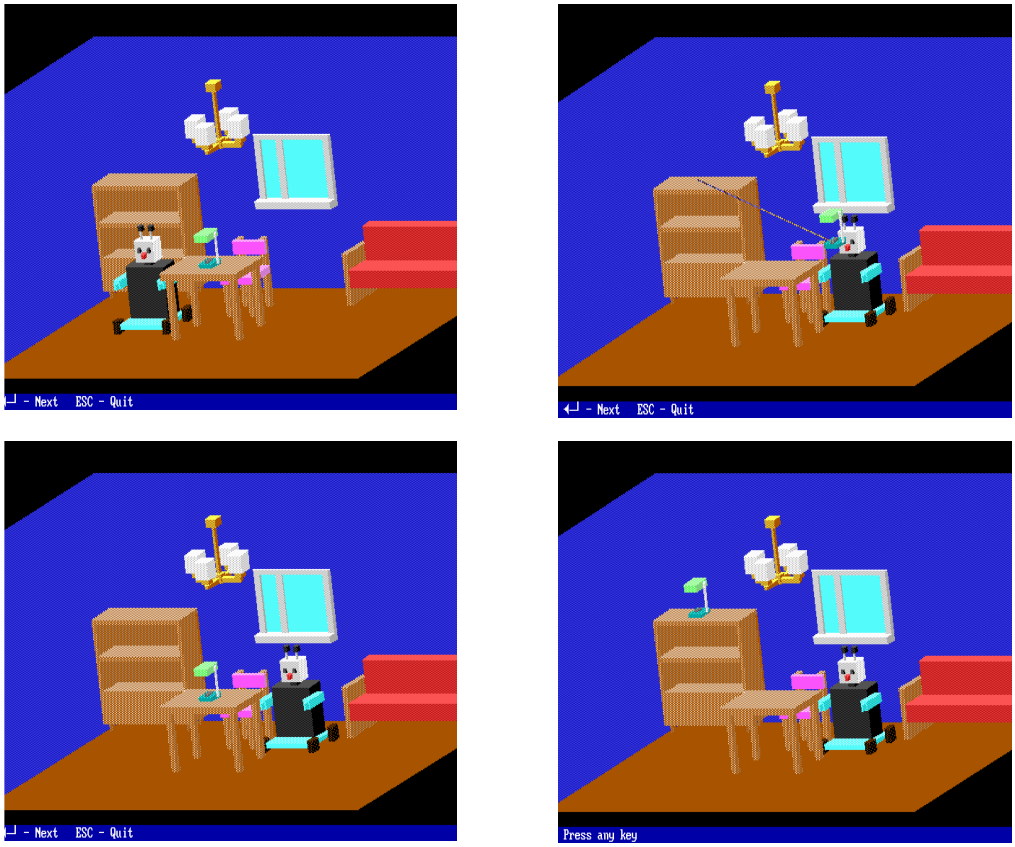


FIGURE 17-8. The computer simulation of the robot`s behavior.

The behavior simulation system understands the input NL text, constructs the plan of actions, and visualizes the set of scenes representing the results of robot's actions. The system solves following six tasks:

1) translation the NL-input to the IR;
2) analysis of the consistency of the initial scene from the standpoint of the spatial logic;
3) construction of the plan (as the set of actions) for the given work instruction;
4) construction of the set of scenes for the visualization of actions results;
5) construction the GR for each scene;
6) scene's visualization (represented as temporal chain of scenes on the display screen).

Results of computer simulation of our example are shown in Fig.17-8. The first scene in Fig.17-8 is the GR of the initial scene described by the NL text including the robot location. In the second scene robot is located near the desk and right from it. In the third scene the robot took the lamp and threw it on the top of the wardrobe (in this scene a trajectory of throwing is shown by a line) and in the final, fourth, scene the lamp is located at the top of wardrobe.

The output of the intelligent mobile service robot's behavior simulation system includes the set of robot's actions, the cognitive map of the initial scene, and the GR's of scenes which are constructed by using pseudo-physical spatio-temporal and action models. This output is used then in the managing system for the control of the robot behavior in the real environment including navigation and obstacle avoidance.

(Copia modificata per una migliore consultazione on-line)

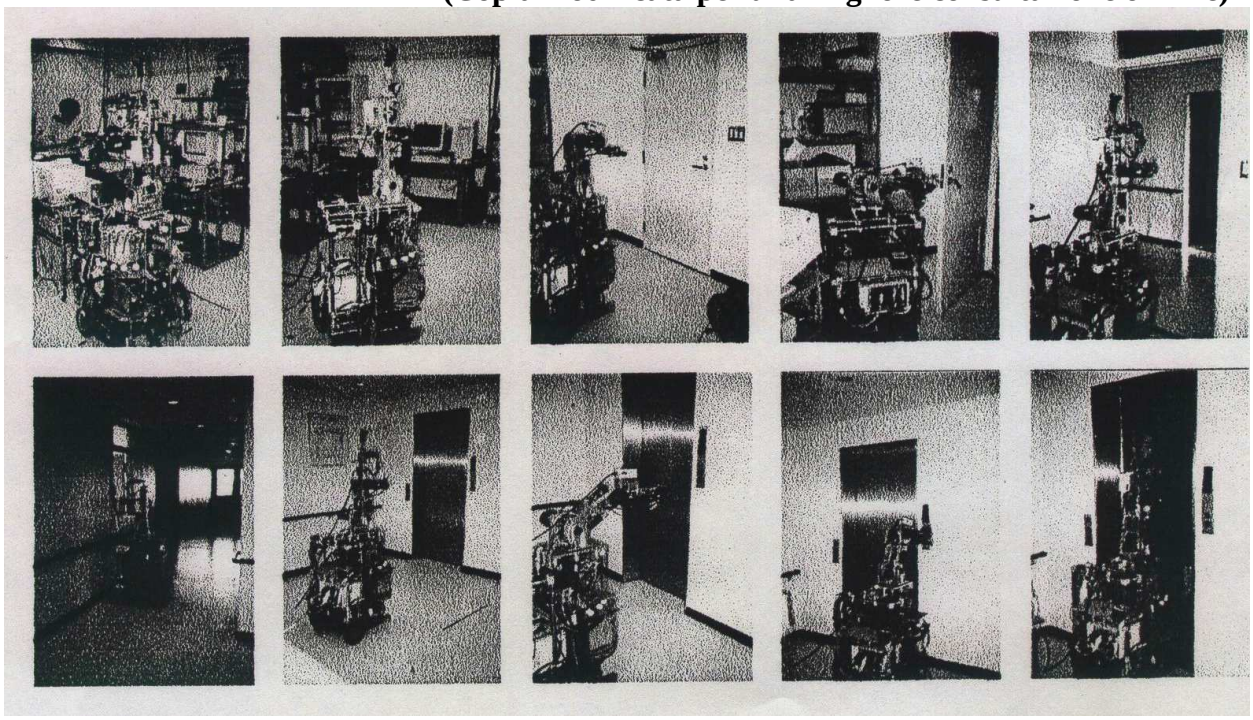


Figure 17-9. the experiment of real behavior of intelligent mobile service robot.

The real experiment was conducted by using the direct human-robot communication line where a human operator introduced the following NL- based instruction for the robot: “Go to the office room and bring correspondence for our laboratory”.
Behavior Planner constructs the plan of actions including the following operations: “go out from the room”(consisting of the set of actions as “turn in front of the door of the room”, “open the door“, “move to an elevator”); “push a knob of the elevator”; “wait”; “come into the elevator”; “move to second floor (office room); “move from the elevator”; “move to the office room”, and so on.
The experimental behavior of the robot performing this task is shown in the Fig.17-9.

Lecture N 18 Intelligent Human-Computer Interaction. Natural Language Processing

In this lecture we will talk about problems of a Human-Computer interaction beginning from a simple dialog and going through Intelligent Multi-Media to Virtual Reality Systems.

Communication is a very important part of evolution and life of any human being. During this process we can interchange by our thoughts, ideas, opinions, associations, dreams, we can evaluate our behavior, acquire new knowledge and data.
How to describe a mechanism of communication and how it can be simulated in computer systems?

For the support a human-computer communication too much interactive computer systems are developed today. They are evolved from simple dialog systems - through intelligent interfaces - to virtual reality systems.
We will consider *intelligent interaction as a capability to deep understanding of semantic content of input message given by a partner of communication*.
Let us consider the general scheme of any communication and define *understanding* capability and its levels as follows.

Model of communication

In Fig.18-1 the general scheme of communication is shown.

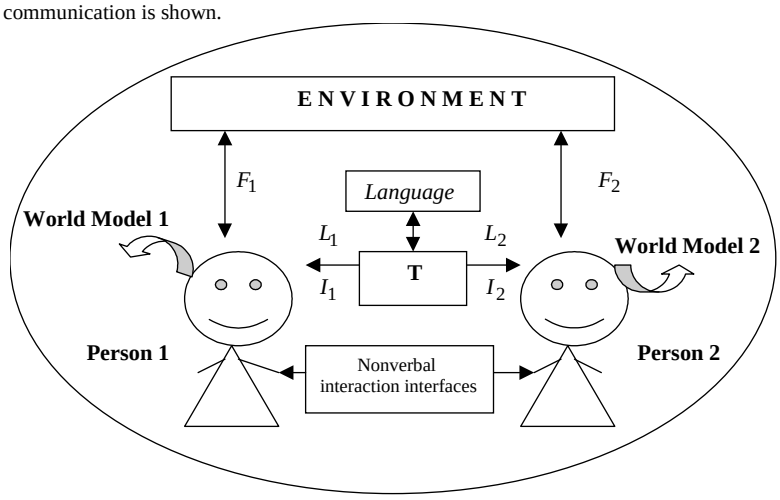


Figure 18–1. The general scheme of communication.

Agents (or partners) of communication process, P_1 and P_2, are immersed into one (the same for the both agents) *environment E*. It will be real or virtual. Agents have *World Models*, M_1 and M_2, *Language* Systems, L_1 and L_2, *Brains*, B_1 and B_2, *Sensor Systems* (generally, vision, ear, taste, tactile and smell organs) and *Effectors* (such as foots, hands, or others).

(Copia modificata per una migliore consultazione on-line)

Each agent has own *mapping function* $F_1(F_2): E \to M_1(M_2)$ which allows to construct the own World Model of a given environment *E* and represent in a memory different things of this reality (for example, objects, events, relations between them, ideas, feelings, associations, representations about the communication partner, etc.).

Each agent has also own *interpretive function* (I_1, I_2) used for the interpretation of an input information from a partner in terms of his world model.

Partners are interchanged by messages. A message may be expressed on a language (*verbal component of communication*) or another ways (*nonverbal components of communication*), for example, by gestures, voice, pictures, etc.

If language system L_1 is not equal to the L_2, then interface *T* is needed. We may consider the following types of communication:

- natural or "*human-to-human*" communication when P_1 and P_2 are human beings;
- "*human-computer*", (or "human-agent", or "human-robot" communication) , when P_1 is a human being, P_2 is a interactive computer system ;
- "*computer system - computer system"* (or "agent-agent", or "robot-robot") when P_1, P_2 are interactive artificial agents, or robots.

For "human-to-human" communication the role of *T* may perform a human translator, or some form of electronic translator, if a partner don't know the language of his opponent.
For "human-computer/robot/agent" communication interface *T* for verbal interaction and nonverbal interaction interfaces must be developed.
Factually, *the history of human-computer interaction is the evolution of computers and their interfaces*. The following scheme reflects that:

machine code translator → assembler languages translators → high level programming languages translators, → dialog systems → intelligent interfaces and multimedia systems → and finally *→ intelligent agents and virtual reality systems.*

We may define *communication as a dialog process for achievement of some goal or intent*.
During a dialog process a partner understands a sense of received message (step1) and reacts on input information by some way (verbal or nonverbal) (step2).
A dialog (and communication) process is successful, if partners of communication can understand each other.
Consider the following necessary for understanding conditions:

1. If L_1 is not equal to the L_2, then interface *T* is needed;
2. *World Models of partners* M_1 and M_2 *must be adequate each other*. The adequacy means that mutual messages of partners can be interpreted in the same terms. For example, consider the following case. Partner P_1 is a passionate football onlooker. His model contains knowledge and data about football games, winners, etc. Partner P_2 knows all Hollywood's movies only, and his world model does not contain information about football games at all. In this case P_2 may not understand messages from P_1 about football games;
3. *Mapping functions* F_1 and F_2 *must be noncontradictory each other*. It means that there is no situation like following: one partner P_1 says, for example, "an orange is yellow", while his opponent, P_2 , affirms that "the orange is black". In this case there are two different contradictory mappings of reality.

Remark 1. Of course, we may especially introduce such kind of contradictions in a dialog, for example, like Alice did it in her magical world (well known story written by Luis Carroll).

4. *Interpreting functions* I_1 and I_2 *must be non contradictory each other*. It means that, if some input *S* is interpreted by the partner P_1 as $I_1(S)$ and by the partner P_2 as $I_2(S)$, then these interpretations must be non contradictory each other. For example, consider the following interpretations of the message *S* = " I saw a penguin in a street": $I_1(S)$ = "the penguin is a bird " and $I_2(S)$ = "the penguin is an animal ". You can see that first interpretation contradicts with second.

The violation of the mentioned above conditions leads to misunderstanding of partner's messages and results in an unsuccessful dialog.
Now we can answer on the following question: why we can understand each other? If four mentioned above conditions are satisfied during a dialog, then it is means that we can understand each other.

Different Levels of Comprehension

How to define *the comprehension skill of a computer system* during a human-computer interaction?

There is no strict definition of that. First designers of human-computer dialog systems define it as follows:
If you having some purpose interact with a computer system and achieve your purpose during the interaction, then it is considered that the computer system understands you.
Of course, this is a very narrow pragmatic definition of understanding and it does not reflect complex deep processes connected with human like understanding.
We will consider the comprehension skill as a *complex process of matching and finding a correlation between input information and knowledge contained in the World Model, reasoning and generation reaction on the input in some form (verbal, or nonverbal*).

Let us discuss different levels of the comprehension (or understanding) skill appearing in a communication process.
Introduce following notations:
T – input information in a textual form (natural language-based or formalized) describing some situation in a problem area;
E – the extended text *T* including the knowledge about given problem area;
P – the extended text *T* including the knowledge about a person and a process of communication;
W - information including nonverbal components of communication (gestures, voice, images, associations, etc.);
TR – rules of text extending based on the structure of the text *T*;
ER – rules of text extending based on the problem area knowledge (Knowledge base and Facts Base);
PR – rules of text extending based on the knowledge about communication process and psychology of a person of communication;
WR – rules of text extending based on the nonverbal knowledge;
A – the answer of a human-computer interaction system;
KB – Knowledge Base;
FB – Facts Base.

Definition:
We will consider that a human-computer interaction system *understands input text T, if it can give right answers* (*from the standpoint of a human expert*) *on all possible questions about situation of real world described by T.*

(Copia modificata per una migliore consultazione on-line)

Based on this definition of the comprehension (understanding) skill, the following levels of understanding in human-computer interaction may be introduced.

Zero level of comprehension is characterized by the following scheme of understanding:

$$T \Longrightarrow A.$$

It means that all answers are given using only a syntactical structure and explicit information in *T*.
For example, consider the following human-computer interaction:
Human input in the form of the following text *T*: "Shakespeare wrote "Hamlet".
System: OK.
Human question: "Who did write "Hamlet"?
System: "Shakespeare ".
Human question: "Who is the author of "Hamlet"?"
System: I don't know.

You see that *zero level of understanding is based only on a simple matching procedure* based on the explicit information given in *T*.
This model of human-computer interaction is realized in some information retrieval systems, which are not intelligent systems.

First level of comprehension is characterized by the following scheme understanding:

$$E \Longrightarrow \text{A.}$$

It means that all answers are given by using procedure $\Phi(T, TR, ER)$ extending the input text *T* on the base of rules *TR* and *ER*,

where $\Phi(T, TR, ER)$ includes searching and inference mechanisms.

So, the *first level of understanding is based on searching and inference procedures*. In this case the system can give right answer on second question mentioned above because it can use the following knowledge: "an author of some thing" is defined as "a person who write this thing".

Second level of comprehension is characterized by the following scheme of understanding:

$$P \Longrightarrow A.$$

It means that all answers are given using procedure $\Phi(T, TR, ER, PR)$ extending input text *T* on the base of *TR, ER* and *PR*-rules.

In this case the system can answer on the questions like "what intent (or goal) has a partner when he is saying *T* " or "what does a partner want when he said *T*?".
Second level of understanding is based on searching/ inference procedures and *includes psychological model of a person of communication*.

Third level of comprehension capability is characterized by the following scheme of understanding:

$$P \Longrightarrow A.$$

It means that all answers are given using procedure $\Phi(T, TR, ER, PR, WR)$ extending input text *T* on the base of *TR, ER, PR* and *WR*-rules.

In this case, the system can answer on questions like "what image has the object (in *T*)?" or "what does a partner show?" or "what does it mean for the system when *T* is the input?" and others.
Third level of understanding is based on searching/inference procedures and *includes psychological and nonverbal models of communication*.

Turing test for AI system

The importance of language understanding in communication between humans and computers was the essence of the *Turing test for AI system.*

Alan Turing, British mathematician had introduced the definition of Artificial Intelligence (AI) system by the following way. Consider a communication process between two persons P_1 and P_2. P_1 is a human, and P_2 is unknown for the P_1 partner of communication. P_1 can talk with P_2, but he can not see him. If after some period of a dialog between the human P_1 and his partner, the P_1 can not recognize whether his partner is an AI-system or a human, then we may consider that the AI-system possesses an artificial intelligence.

First human-computer interaction systems appeared zero-one levels of comprehension, modern intelligent multi-modal interfaces and virtual reality systems appear high levels of comprehension skill.

Human-Computer Dialog Systems Design History

One of the first designers of natural language-based dialog systems, E. Weizenbaum, developed a system called "Elize"[29]. The typical dialog with Elize was, for example, as follows:
User: People are similar each other.
Elize: How they are similar?
User: They some times make crazy things.
Elize: May you give some example?
User: Yes, my friend invites me to talk with you.
Elize: Your friend does invite you to talk with me?
User: He said that I am today sad.
Elize: Are you today sad?
User: …. Etc.

Such a kind of a dialog may be continued infinitely. Let us discuss main ideas of "Elize" dialog.You can see that "Elize" generates answers by using *paraphrases*. Paraphrasing procedure converts user's sentence into questions, or replaces some words in input by another. The mechanism of paraphrasing uses a set of templates which are compared with the input. If some template is matched with the input, then an answer is generated according the given template.
For example, consider an answer design on the following input phrase: " It seems that you hate me".
The phrase is analyzed from the left side to the right, word by word, and *key words* are found. Key words have a priority. In this sentence there are two key words: "*y*ou" and "*me*".
Then template *D* of this input sentence is as follows:

D: (0 *you* 1 *me*).

"0" means that here may be any number of words; "1" means that between "you" and "me" must be 1 word. The phrase is represented as the following parts:

(1) it seems that (2) you (3) hate (4) me.

"Elize" has a database of key structures as follows:

$$(K_i(D_{i_k})(R_{i_1}, R_{i_2}, ..., R_{i_m})),$$

where K_i is a key word, D_{i_k} is a template of input phrase, and $(R_{i_1}, R_{i_2}, ..., R_{i_m})$ is a rule of construction an answer.

Input template *D* is compared with D_{i_k} for a given key word. If matching is successful, then answer is based on the scheme $(R_{i_1}, R_{i_2}, ..., R_{i_m})$. If there is no matching, then general answer is constructed, for example, by the following way:

<*template*> + (2)(3)(4),

where <*template*> is " why do you ask me that".
So, finally the following answer is generated:

Why do you ask me that I hate you?

In the case with "Elize", the system can not understand anything, but it can imitate understanding. Such kind of a dialog gives an *illusion of a dialog*.

Remark 2. As you many times observed, some times in real situation human beings also use this kind of dialog in communication. For example, if you don't want to deep in the essence of what your partner is saying, yours answers will be simple "Yes-yes", and so on. This form of dialog is called a *"fatigued" dialog*.

There was a curious case in a dialog systems history. One time a vice-president of a company, where programmer Dan Bobrow was working, calls to him and the following dialog was occurred:

Vice-president: I would like to talk about new computer.
"*Bobrow"*: Why do you ask me about this?
Vice-president: May be, I can obtain a contract with…
"*Bobrow*": Why do you sure in that?
Vice-president: My partners are never seen our computer system.
"*Bobrow*": Does it have a great meaning for you?
Vice-president: It seems to me that you are in doubt. Call me later.

But Bobrow didn't call him because the vice-president was talking with a computer program like "Elize", not with Bobrow. This is a legend from first AI researches.

Another interesting example among first dialog systems is the program called "robot" developed by Terry Vinograd (MIT, USA) in 1973 [30].
The "robot" maintains a natural language-based dialog with a human operator and performs some actions in a simple world containing several cubes, cylinders and pyramids. It is assumed that robot has video camera and can recognize objects in the world (Fig.18-2).

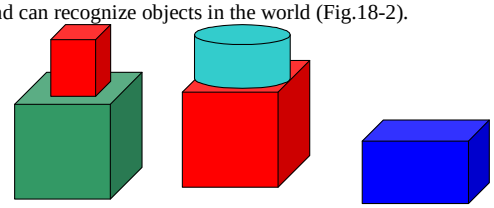


Figure 18-2 .A Block's World scene.

Typical dialog with the Vinograd "robot" is following:
Human Operator: "grasp the cube"
Robot: "I don't understand which cube you mean"

(*Remark* 3. There are four cubes in the scene, and the program knows that phrases beginning with "the" are intended to refer to a specific object the speaker has in mind.)
Human Operator: "grasp the big red cube"
Robot: OK.

Consider human-computer interaction, based on some subset of natural language (NL). Usually it is restricted and simplified NL for given problem area NL interaction is an extremely complex phenomenon. It involves recognition of sounds, words, phrases, and combining forms, comprehension and usage.

Natural Language Processing Problems

The methodology of NL-processing is developed in three main directions:

1) theoretical linguistics based approaches applied for the task of machine translation;
2) pragmatic approaches applied for the task of natural language access to databases and knowledge bases ;
3) hybrid approaches applied for the task of intelligent agents design.

First direction of research tries to develop general computational models of natural language, second one tries to design problem oriented interactive systems. In this case some restricted subset of NL are considered, and their simple model are designed. Third direction is a combination of first and second approaches.

There are three kinds of NL-processing models: *syntax-based processing*; *semantic-based processing*, *hybrid models*.
First type models are characteristic for first direction of research, second one for pragmatic direction, and last one is used in modern interactive systems.

Syntax-based processing

In syntax-based NL-processing a syntax plays the main role in input processing which consists of the following levels: morphological, syntactical, and semantic (Fig.18-3). Consider main ideas of syntax-based NL- processing based on a given NL grammar. Let we have the following simple English phrase grammar represented as a set of following rules:

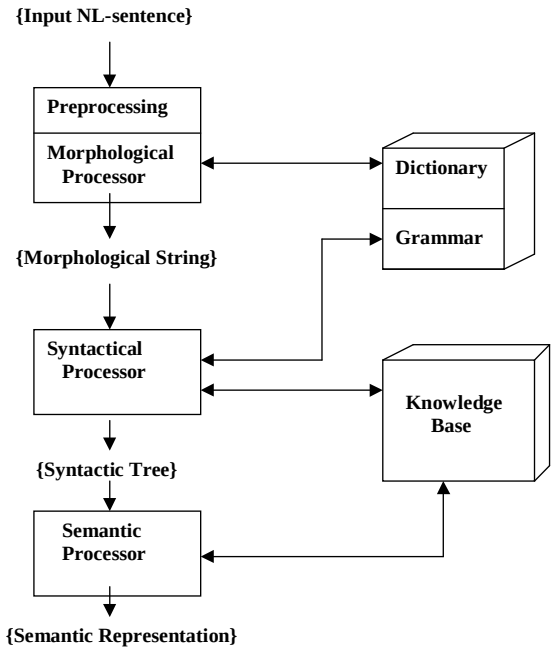


Figure 18–3. The structure of Syntax-based NL- processing.

(Copia modificata per una migliore consultazione on-line)

$$S \rightarrow NG + VG + NG$$
$$NG \rightarrow DET + NUM + ADJ^{*} + NOUN$$
$$VG \rightarrow VERB$$
$$DET \rightarrow THE / A,$$

where $S, NG, VG, DET, NUM, ADJ, NOUN, VERB$ are nonterminal symbols of the grammar. *S* means a sentence, *NG* means a noun group, *VG* is a verb group, *DET* is a determinant, *NUM* is a number, *ADJ* is an adjective. The asterisk "*" is used to indicate that one or more repetitions of the option are allowed. For example, the following sentence may be described by this grammar:
" The three big brown dogs ate a raw steak".
Let this sentence be the input to the NL-processing scheme given in Fig.18-3. In preprocessing and morphological stage we define the number of words in the input and their morphological characteristics. Output from this stage is a morphological string, for example, as follows:

$(Mstring:$
$(w_1 : entry = the, syntax - group = DET, property = definite)$
$(w_2 : entry = three, syntax - group = NUM)$
$(w_3 : entry = big, syntax - group = ADJ)$
$(w_4 : entry = brown, syntax - group = ADJ)$
$(w_5 : entry = dog, syntax - group = NOUN, property = plural)$
$(w_6 : entry = ate, syntax - group = VERB)$
$(w_7 : entry = a, syntax - group = DET, property = undefinite)$
$(w_8 : entry = raw, syntax - group = ADJ)$
$(w_9 : entry = dog, syntax - group = NOUN, property = single)$
$(w_{10} : entry = ., property = end - of - phrase)$)

The *Mstring* inputs to the Syntactical Processor, or Parser, which finds a syntactical structure (called also as *syntactical tree*) of the given input. In our example, the output from the Parser is shown in Fig.18-4.
Syntactic tree inputs to the Semantic Processor, which finally generates some semantic representation (*SemR*) of the input sentence, for example, based on a semantic network. Semantic representation describes *meaning* of the input phrase.
The *notion of meaning has no exact definition*. " A given phrase has meaning" means that most people with a given native language consider that it is so. "A given phrase is nonsense" means that most people with the given native language consider that it is so.
When we introduce semantic representation we define meaning exactly.

Semantic Processor performs two main functions: construction of semantic meaning of the input and *conceptual processing* of the information given in the input. Conceptual processing is one of the main mechanisms realizing *the comprehension skill* of a system. This is *a complex process of matching and finding a correlation between input information and knowledge contained in the World Model and reasoning.*
For example, consider the sentence "John ate a frog" word by word. The concept underlying the word "John" is basically a set of features some of which are known. "John" is a human. Humans have sex, and "John" is "male". Humans have names, and "John" is the name.
Notice that already much of what we said could be wrong. For example, "John" may be a dog. So, we *make assumptions when we understand NL*.

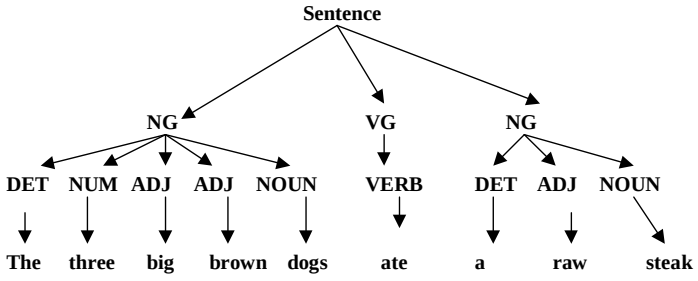


Figure 18-4. A syntactical tree structure.

NL-meaning representation

Generally speaking, meaning representation consists of concepts and relations between them. The following question is very important: *How much concepts and how much relations are needed in order to describe meaning of any NL-sentence?*

A lot of NL-meaning representation models have been developed. Consider one of them – *conceptual dependency representation* of R. Schank [31] and discuss main ideas of the model.

Conceptual dependency representation for NL-meaning

For NL-meaning representation, R.Schank developed so-named *conceptual categories and rules* in order to describe all concept's types and their combinations. We will now make a complete list of these concepts.

Conceptual Categories

There are following *conceptual categories* (called also as conceptual roles, or types of concepts):
Conceptualization: the basic unit of the conceptual level out of which thoughts are constructed.
A conceptualization is made up of the following:
Actor: the notion of the performer of an *ACT*.
ACT: an action is what can be done by an actor to an object. There are only eleven of these ACTs (details will be given below).
Object: a thing that is acted upon.
LOC: locations – every physical ACT has a location that modifies the place of occurrence of the conceptualization that included it. Locations are considered to be coordinates in space. LOCs can modify conceptualizations

as well as serve in the role of direction.

Direction: the location that an ACT is directed towards.

Recipient: the receiver of an object is the result of an ACT. Included within recipient is the *donor* of the object.

T: times – most conceptualizations have a time. The time is considered to be a point or a segment on a time line. This point or segment may be absolute (for example, 2 p.m. on 8/3/2001) or relative (for example, before yesterday).

State: the state that an object is in.

PP: only physical objects are *PP*s. *PP*s may serve in various roles in a conceptualization. *PP*s that are animate, or have animate qualities (like "machines") may be *actors*. (Natural forces, a special kind of *PP* (like "wind"), may also be actors). Any PP may serve in the role of object. A PP may be in the role of direction, in which case it is considered to indicate the location of that PP is what intended. Animate PPs may also serve as recipients.

AA: action aiders – these are modifications of features of an ACT, for example, a speed factor. There are very few AAs.

PA: attributes of an object. PAs take the form: State (Value). That is, a PA is an attribute characteristic (like color or size) plus a value for that characteristic (red or 10 cm). Every physical object can be defined by a set of attribute States with specific values.

Conceptual Rules

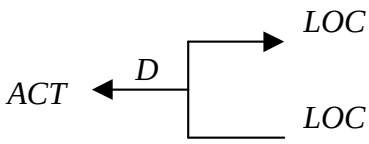
The written above conceptual categories combine in certain specified way called the *conceptual syntax rules*. Schank introduced special graphical structures (Schank's diagrams) for description of conceptual categories combinations as follows:

1) $PP \Leftrightarrow ACT$: It means that certain PPs can ACT.

2) $PP <\equiv> PA$: PPs (and some conceptualizations) can be described by an attribute.

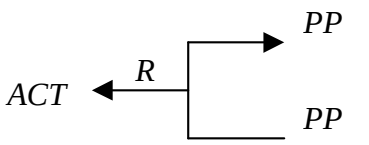
3) $ACT \overset{o}{\leftarrow} PP$: ACTs have objects.

4)



ACTs have direction.

5)



ACTs have recipients.

6) $ACT \overset{o}{\leftarrow} \Updownarrow$: ACT requires conceptualizations or combinations of conceptualizations as objects.

7) $ACT \overset{I}{\leftarrow} \Updownarrow$: ACTs have conceptualizations as instruments.

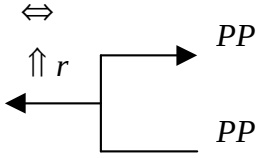
8) $\overset{PP}{\underset{\Leftrightarrow}{\updownarrow}}$: PPs can be described by conceptualizations in which they occur.

9) $\overset{T}{\underset{\Leftrightarrow}{\updownarrow}}$: Conceptualizations have times.

10) $\overset{PP}{\underset{\Leftrightarrow}{\updownarrow}}$: Conceptualizations have locations.

11)

$\Leftrightarrow$



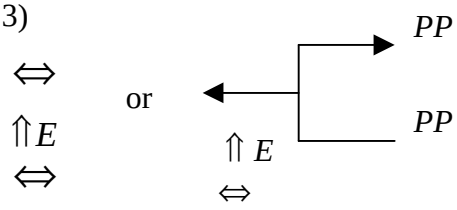
: Conceptualizations can result in state changes for PPs.

$\Leftrightarrow$

12) $\overset{\Leftrightarrow}{\Uparrow R}$: Conceptualizations involving mental ACTs can serve as reasons for Conceptualizations.

(Copia modificata per una migliore consultazione on-line)

13)



: States or state changes can enable conceptualizations to occur.

14) ACT / AA : ACTs can be varied along certain dimensions (e.g., speed for motion ACTs).

Just as lexically words are put into sentences, conceptually concepts are put into what we call *conceptualization*. This notion is central in the conceptual dependency model. The conceptualization (depicted as " ⇔ ") consists of an actor and an action plus a specific set of conceptual cases. There are two main categories of actions: *physical actions*, or actions performed on physical objects; and *mental actions* performed on ideas.
There are restrictions on what kinds of objects can be used with any given ACT, which constitutes what we have called the *conceptual semantics* for each ACT. Each ACT also has a specific set of conceptual cases that it takes.
Remark 4. For each action it is important to have in mind that the notion of the meaning of an ACT can only really exist with respect to the effect of a given ACT on the system that employs it. Thus the real meaning of each ACT is the set of inferences that are possible true when that ACT is present.

Primitive actions
There are eleven primitive actions considered in the *conceptual dependency representation* model: *PROPEL*, *MOVE*, *INGEST*, *EXPEL*, *GRASP*, *PTRANS*, *ATRANS*, *SPEAK*, *ATTEND*, *MTRANS*, *MBUILD*.
Consider conceptual semantics for each listed above action.

Physical ACTs
PROPEL: means "apply a force to". It takes objective and directional case. Its object must be a physical object, and there is a rule in memory that tests to see if the size of the object is small enough with respect to the force being exerted on it in order to establish if the object will now be in a new location. The Directive case for PROPEL indicates the direction of the force being exerted. Unlike other ACTs, PROPEL permits inanimate actors (for example, machines).
MOVE: means "move a body part". The only possible objects for MOVE are body parts. MOVE requires Directive case which is used to describe the path followed by the body part.
INGEST: means "take something to the inside of an animate object". The object of INGEST must be smaller then the particular body opening of the actor that is entering. If the object is bigger then it can be inferred that it was divided up somehow into smaller bits previous to INGESTing. The Directive case for INGEST is always to a body opening and from the original position of the object.
EXPEL: means "take something from inside an animate object and force out".
GRASP: means "to physically grasp an object". The object must be within certain size limits, and the Directive case is always to the body part doing the GRASPing.
PTRANS: means "to change the location of something". Any physical object may be PTRANSed. PTRANS requires Objective, Directive and Instrumental cases. The Directive case for PTRANS contains the old location of the object and the intended new location. The concept of Instrument is very important for PTRANS since the actual ACT that was performed is specified in it.
ATRANS: means "to change some abstract relationship with respect to an object". The object is a combination of a physical object and an abstract relationship that that physical object has with some animate object. The animate object is indicated in the Recipient case.
SPEAK: means "to produce a sound". Its object is always some type of sound. Since sounds can be directed, Directive case is always present. The "from" part being the source of the sound and the "to" part being the direction towards are described by the Directive case.
ATTEND: means "to direct a sense organ or focus organ towards a particular stimulus". The object of ATTEND must be a sense organ (nose, eyes, ears, tactile receptors). ATTEND takes Directive case. The "to" part of the Directive case is always the location of the stimulus being focused upon.

Mental ACTs:
MTRANS: means "to transfer information". Therefore objects of MTRANS are always conceptualizations. MTRANS takes recipient case, where the possible receiver are parts of people's heads (for example, long-term memory) and the possible senders are sense organs.
MBUILD: means "to create or combine thoughts". It requires a different kind of object arrow than the other ACTs. MBUILD takes conceptualizations as objects and creates new conceptualizations from them.

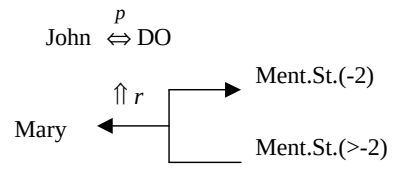
States
Many states of objects can be described by scales which have numerical values. In so doing, we do not claim that humans represent states in this way, but rather that they can detect differences between adjectives that these scales suggest. That is, they can tell you that "angry" is just a little less of the same thing than "furious".
We have chosen to treat these things by scales.
The following scales are used: *HEALTH*, *FEAR*, *ANGER*, *MENTAL-STATE*, *PHYSICAL-STATE*, *CONSCIOUSNESS*, *HUNGER*, *DISGUST*, *SURPRISE*.
Each numerical value of a scale corresponds to some linguistic value. For example, the scale HEALTH goes from –10 to +10 and has numerical-linguistic correspondence shown in Table 18-1.

Table 18-1. The *HEALTH* scale.

Linguistic	*Numerical*
dead	-10
gravelly – ill	-9
sick	$[-9,-1]$
under – the – weather	-2
all – right	0
tip – top	$+7$
perfect – health	$+10$

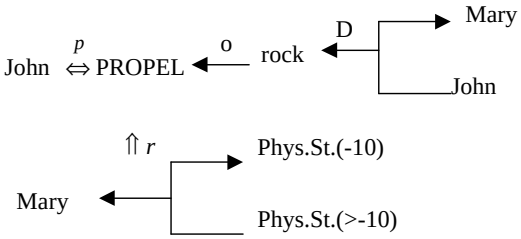
Consider now a few examples of NL-meaning representation based on Schank's diagrams.

(1) John annoyed Mary.

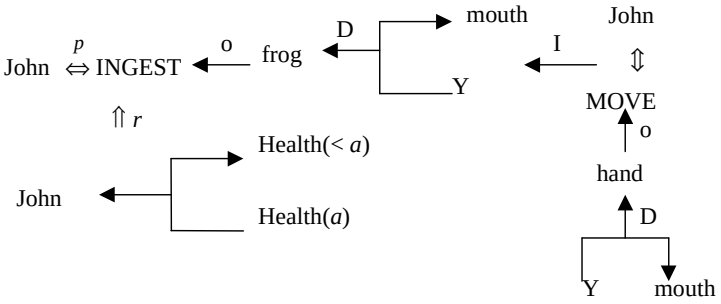


(Here index "*p*" means a past time, and DO is an unknown action).

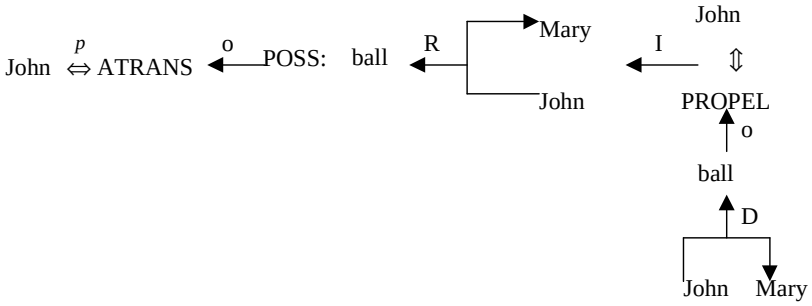
(2) John killed Mary by throwing a rock at her.



(3) John ate a frog.



(4) John threw a ball to Mary.



Semantic Networks-based NL-meaning representation

Another popular scheme of NL meaning representation is based on semantic networks. Figure 18-5 shows a meaning representation of the following NL-sentence
" A big brown dog is eating a raw steak".

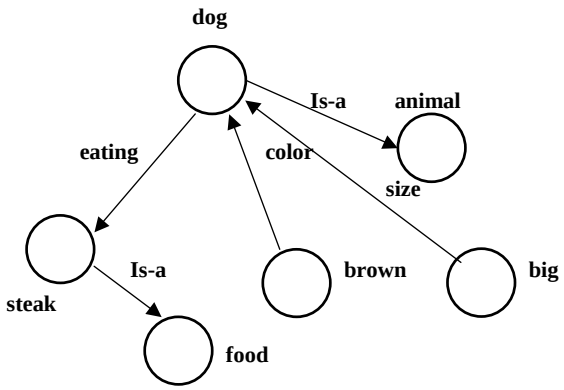


Figure 18-5. Semantic network-based NL-meaning representation.

Remark 5. The choice of NL meaning representation depends upon a problem task. For example, for NL understanding and machine translation tasks more deep representations like Schank's model are needed, for information retrieval tasks more simple models are used.

(Copia modificata per una migliore consultazione on-line)

NL-processing based on Augmented Finite State Transition network

A NL-string can be transformed into semantic structure (like a semantic network) with the aid of a program (called *Linguistic Processor*) that consults a dictionary (or lexicon) and a grammar.

One of typical method of NL-processing is based on *Augmented Finite State Transition network* (AFSTN) developed by Woods [32].

AFSTN represents a NL interpretation process and a grammar of NL in the form of transition network shown in Fig.-6.

Consider the following simple phrase structure grammar:

S → NP + [AUX] + VP

S → AUX + NP + VP

NP → [ART] + [ADJ*] + N + [PP*]

PP → PREP + NP

VP → V + [NP] + [PP*],

where S, NP, PP, VP are nonterminal symbols of the grammar. S means a sentence, NP means a group of a noun, VP is a group of a verb, AUX is an auxiliary word, PP is a group consisting of a noun and a preposition, ART is an article, ADJ is an adjective.

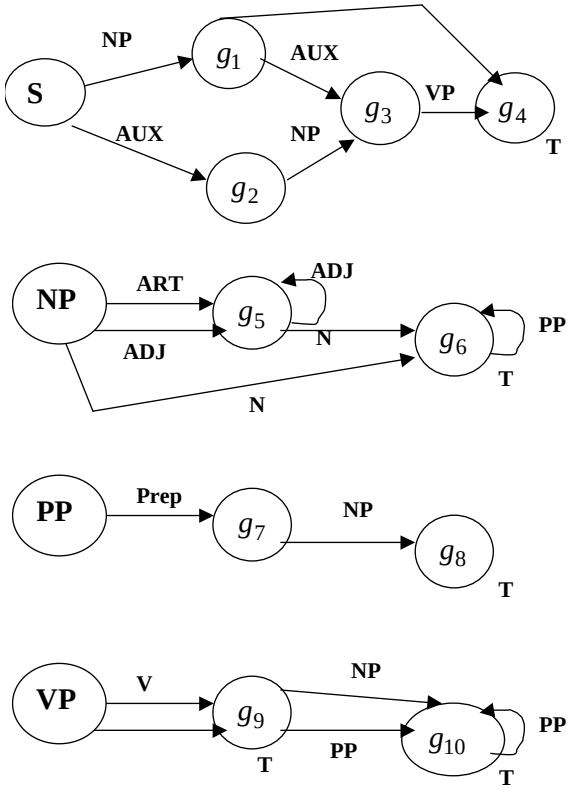


Figure 18–6. The Augment Finite State Transition Network for a simple grammar.

This grammar is a context-free, it uses the parentheses symbols "[]" for indication of options which may absent and asterisk "*" to indicate that one or more repetitions of the option are allowed.

In Fig.18-6 the AFSTN that represents this grammar is shown. The figure shows only syntactic category information associated with the arcs. But each arc may in fact have an associated set of conditions to be met and operations to be performed as control passes from state to state.

You see that the nodes of AFSTN are labeled by nonterminal symbols (S,NP,PP,VP) of the grammar and also by some states g1, g2,…g10.

Some states – g4, g6, g8, g10 – are specially marked by the symbol "T" which means that a phrase or sentence can end at the node.

The arcs of AFSTN are also labeled by nonterminal grammar symbols and represent condition that must be satisfied in order to go along this arc.

The grammar can be recursive (with asterisks). In this case AFSTN contains loops in their structure as it is shown in Fig.18-6.

The AFSTN network consists of a set of subnetworks describing NL-processing of each fragment of a grammar.

In Fig.18-6 "S"-net and "NP"-net,"PP"-net,"VP"-net are shown.

Main part of linguistic processor is a monitor which like a scanner looks at each word of a string, examines a current word, matches it with AFSTN , defines next transition in AFSTN, and construct a fragment of future syntactical structure of input.

Let us look at the work of the monitor for the following input string:

"The merry widow danced a jig".

It takes the first word – "the" and goes to the first node (S) of the "S"-net. The first arc is one labeled NP, which causes a transfer to the "NP"-net.

The first word in a string ("the") corresponds to the name of the first arc, ART (article), and allows transition to state g5, constructing a fragment of syntactic tree (as, for example, (S((NP(ART- the)…) and scanning the next word in a string.

The next word "merry" has category Adjective which allows transition along ADJ-arc (a loop) to state g5, constructing a fragment of syntactic tree (as, for example, (S((NP(ART- the)(ADJ- merry)…) and scanning the next word.

The word "widow" allows transition along N-arc to state g6 labeled T that means that the noun phrase has been satisfied. This allows to finishing construction of the fragment (S((NP(ART- the)(ADJ- merry)(N-widow)) …) , then return to the "S"-net and achieve transition along NP-arc in net S to state g1 and scanning next word "danced".

The word "danced" causes transition along VP-arc that causes a transfer to the VP-net.

Then transition along V-arc to state g9 and scanning next word "a" are performed.

NP-arc causes a transfer to NP-net, then transition along ART-arc to state g5 and scanning next word "jig".

The word "jig" allows transition along N-arc to state g6 labeled T that means that the noun phrase has been satisfied.

This allows to return to the "S"-net, achieve transition along VP-arc in net S to state g4 and construct the following fragment of syntactic tree:

(S(NP((ART- the)(ADJ- merry)(N-widow)) VP (V-danced) NP((ART-a)(N-jig))).

(Copia modificata per una migliore consultazione on-line)

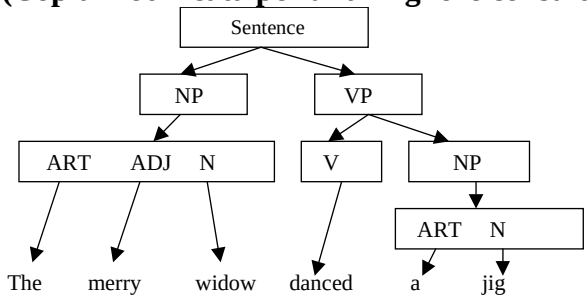


Figure 18–7. A Syntactic Tree of the given phrase.

The state g4 is labeled T which means that sentence is satisfied. In this case a next word in the string is taken and analyzed.
If it is "." (a point, the end of sentence), then NL-processing is successful.
If one or more words lasted in a string, then it means that given AFSTN (grammar) is not enough for the analysis of the given input.
Graphically the *syntactic structure* of the given sentence is represented in Fig.18-7.

The next step in NL-processing is a construction of *semantic meaning of NL sentence*. For meaning representation we will use semantic networks model. In this case the meaning of our NL-sentence can be described, for example by semantic network shown in Fig.18-8.

Finally, consider example of syntactic and semantic representations (shown in Fig.-9 and Fig.-10) for the following Italian sentence: *Ieri la ragazzina dai capelli rossi e mangiato una mela.* {Yesterday the red-haired girl ate an apple.}

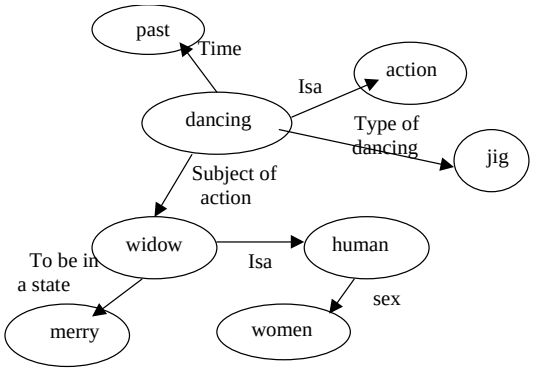


Figure 18–8. Semantic Representation of the given phrase.

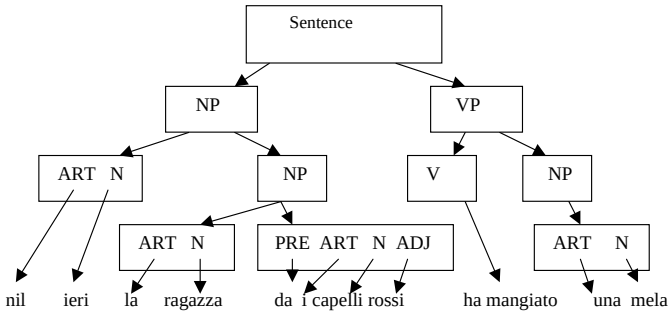


Figure 18–9. A Syntactic Tree of the given phrase.

Semantic-based processing

This approach usually applied for the task of natural language access to databases and knowledge bases in a given problem area. In semantic-based NL processing a syntax plays an additional role, and the process of meaning extraction is mainly based on a knowledge base of a given problem area. Linguistic Processor transforms a NL- string into an internal meaning representation by using a dictionary (or lexicon) and a knowledge base.
Discuss main ideas of NL-processing in the semantic-based approach.

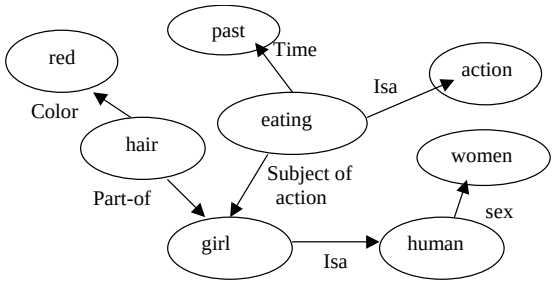


Figure 18–10. Semantic Representation of the given phrase.

The structure of semantic-based NL- processing is shown in Fig.18-11.
Usually, the internal meaning representation coincides with a knowledge representation. For example, it can be a frame-based representation. In this case, an input NL-phrase is translated into frames representatives.

(Copia modificata per una migliore consultazione on-line)

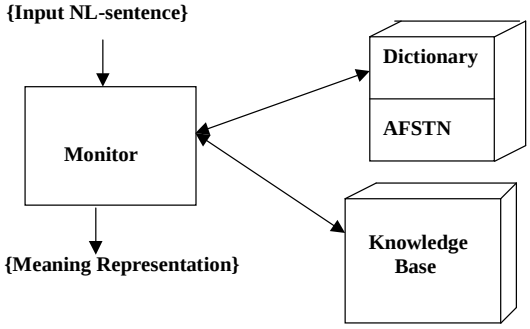


Figure –11. The structure of semantics-based NL-processing.

Consider, for example, the following problem area and problem task. We want to design a NL-interface to a Data Base containing information about all persons of some university, including their names, ages, occupation, addresses, participation in international conferences, grants, etc.
Begin from the dictionary design. The dictionary consists of a set of *entries* describing all NL-words from the pragmatic standpoint, that is, semantics of NL-words is described in terms of Knowledge Base (KB) elements, for example in terms of frame-structure elements. Let us write a general form of a frame structure as follows:
(<*name of frame*>:
<*name of slot*> < *description of value*> > <*value of slot*>).
All NL-lexicon is divided on two classes: *meaningful* (*m*-type) words and *functional* (*f*-type) words.
The meaningful lexicon has analogs in the knowledge base, while the functional lexicon has no analogs in the knowledge base (for example "." (the point)).
Each word can be described in a system's dictionary as follows:
(<*entry*>:
type = *m*-type;

meaning = (< w_1 >,< w_2 >,…,< w_i >)),

where < w_1 > is one simple description of the entry.

In the case of homonym, one NL-word may have a few, < w_1 >,)< w_2 >,…,< w_i >, meaning descriptions. Each simple description of entry's meaning < w_1 > is represented by two designations:

< w_1 > = (<$\in$ - name of frame>, <type of element in a given frame>),

The first designator, <$\in$ - name of frame>, shows to what frame in KB belongs the *entry*. The second designator shows what role plays the *entry* in the given frame (is it a <*name of frame*> or <*name of slot*>, < *description of value*> >, <*value of slot*>).
Functional entries have no meaning.
For semantic NL processing we will also use an *Augmented Finite State Transition network* (AFSTN). Call it a semantic AFSTN. Each state of the semantic AFSTN is defined by a word type, has a number of preconditions and post-actions.
The analysis of an input NL-phrase is considered to be successful if beginning from a first left word in the phrase and initial state of the AFSTN and coming through all phrase word by word, we are coming in the final state of the AFSTN.
Let us describe AFSTN formally as follows.

AFSTN $\Rightarrow$ (<AFSTN-description> <the set of states>
<AFSTN-description> $\Rightarrow$ (<the name of initial state> <the name of final state>
<the list of registers> <the list of available types>
<the name of the file with procedures>)
<state> $\Rightarrow$ (<the name of state> <the list of transitions>)
<transition> $\Rightarrow$ (*m*-type/ *f*-type/rest <transition body>)
<transition body> $\Rightarrow$ <the return point>/ <PROG>
<the return point> $\Rightarrow$ JUMP/ MOVE/FINISH/UPSET
JUMP $\Rightarrow$ "take the following word in a phrase"
MOVE $\Rightarrow$ "change AFSTN state without changing the word"
FINISH $\Rightarrow$ "finish processing"
UPSET $\Rightarrow$ "return to processing of alternative. If there is no alternative, finish processing".

Monitor Functions
The monitor performs the following functions:
1) preprocessing of an NL-input including extraction information from a dictionary;
2) control of processing by AFSTN the NL-input including:
3) extraction a current word from the input string, analysis of meaning of this word, choice one meaning (in the case of homonym) and performing actions connected with processing by the AFSTN, and finally generation of a fragment of internal meaning representation.

Example of NL-based interactive system. One interesting example of this type NL-based interactive system is a dialogue system ALFRESCO [33] that allows a user to interact with a database of Italian medieval frescoes, asking about represented persons, subjects, authors, dates, details, etc. For example, he may ask:
"*Which other authors made frescoes with Leonardo da Vinci during the 15th century*?"

ALFRESCO takes this NL-phrase, extract the meaning represented in frame language, for example, as follows:

(fresco1:
isa = fresco
author = Leonardo da Vinchi;
name = "Last Supper";
place = Milano;
date = 15th century)
	(fresco2:
	author = <find name>;
	place = Milano;
	date = 15th century)

(Copia modificata per una migliore consultazione on-line)

Intelligent Multi-Modal Interfaces

Our social ways to exchange information have changed. As example, you can see growing popularity of the Internet. More and more people understand new capabilities and skills hidden in the Internet. Internet allows easy access and manipulation of complete information units. This aspect enables consumers to reuse creatively materials for their own purposes. Design of intelligent interfaces to Multi-Media like Internet and design systems for selecting and presenting media content becomes now a central problem. These systems have to be based on AI technology. What does this mean for the future of media computing?

1) We must provide intelligent tools for easy navigation and interpretation of audio-visual, and textual content in growing information spaces.
2) We need to access to the content hidden in information structures. For example, raw video data must be processed in order to extract its content in more usable form.

All these problems require special semantic description languages for content representation. (Examples of the languages can be found in [48]). So, the future of media computing lies in transforming search engine relying on keywords to intelligent interfaces offering "find what I want".

The general structure of an intelligent multi-modal human-computer interface is shown in Fig.18-12. Consider main modules of this interface.

Speech recognition module is trained on a speech corpus and uses a linguistic knowledge. It recognizes spoken words, phrases, and sentences from some dictionary. The input speech signals are first digitized, transformed into feature vectors, and then used in the speech recognition module to match with already known speech patterns and to recognize small speech units, for example, phonemes.

Language modeling module takes recognized speech units and combines them into relevant words and sentences.

Reasoning module does approximate reasoning over user's query and allows for vague, fuzzy queries to be used. The module performs two levels of matching a query to a database. The first one is exact matching, when the query matches exactly the information in the database. The second one is similarity-based matching which involves interaction between the user and the system. The module finds conceptual
similarities based on previous knowledge represented, for example, in terms of fuzzy rules.

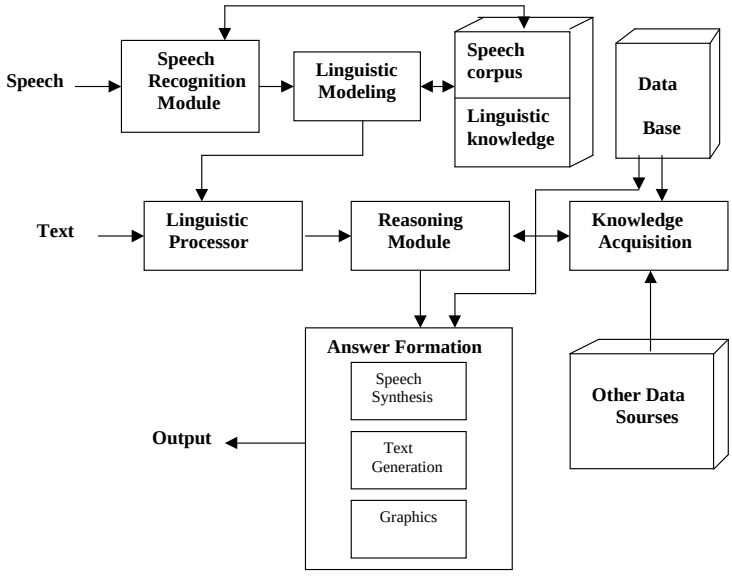


Figure 18 –12. Intelligent multi-modal human-computer interface structure.

Knowledge acquisition module is used to extract knowledge (content) from different sources of information: numerical, visual, symbolic, textual, topological (geometrical), and sensory including sound, touch, smell, taste. The idea is to aggregate all these information using natural language just as human beings do. This is a so-called *information fusion* problem, which is very important issue in intelligent interactive systems design.

Answer formation module produces the answer to the user and performs a dialogue at any phase of the information retrieval. It has speech synthesis, text generation and graphics submodules.

Lecture N 19 Virtual Reality Systems and Intelligent Interaction Agents - New Forms of Human-Computer Interaction

Virtual reality systems (VR-systems) as a qualitatively new form of human-computer interaction became now the focus of a large number research activity in many different areas [34]. They provide the new tools in professional training, education, medicine, robotics, designing of intelligent systems.

Traditional intelligent interactive systems are evolved in the context of *verbal paradigm* when for the communication process different language-based approaches were developed.

Next step was the change of the verbal paradigm to – *multi-modal paradigm*. In this context different approaches based on the use of visual, auditory, and verbal information are developed.

The appearance of VR-systems gives the possibility to change the paradigm of evolution of intelligent interactive system design from the multi-modal paradigm – to *virtual reality paradigm*. In VR-context approaches including different psychological aspects of behavior (emotions, instinct and intuition mechanisms) may be included in the process of communication and problem solving.

In robotics the problem of design new intelligent control systems including biological mechanisms such as intuition, instinct, emotions needs the development of new approaches and tools. Based on VR-system a designer of a robot's intelligent control system can simulate different virtual worlds and examine different behavior models in artificial life of robot. This allows to evaluate control algorithms of a real time behavior of the robot and reduce difficulties connected with such troubles as robot collisions with real objects and robot hardware damages. VR-systems are qualitatively new tools in the task of remote control of robot. In this case VR-system creates the environment in which robot works and operator can feel this environment. It is very important in many tasks, for example in remote surgery by medical robot.

Define the VR-system as *an advanced human-computer interface which immerses a user in a three dimensional world of his problem model for direct perception of this world and direct manipulation with objects in a real time. At this time the special technology of mapping of 3D-world intended for all forms of human perception is realized.*

VR- technology provides the interfaces to all perceptual and motor systems of human. (We will call these interfaces as *VR- interfaces*).

Why VR-systems are qualitatively new tools?

Consider, for example, a computer aided design (CAD) of a new type of an aeroplane. This is very complex and laborious process. Using a traditional CAD system a designer can construct the detail schemes of all parts of design and by the means of her *vision* (*only*) examine and modify this parts. VR-system allows to immerse the designer inside the aeroplane's model. So, the designer can see and feel the internal environment of the aeroplane. He may touch to all parts of design, by the direct

manipulation quickly change locations of some parts and find more optimal solution. So, the VR-system decreases essentially the time design of new prototypes and increases the quality of design. Therefore these systems became the indispensable tools in many areas.

Based on a VR-system a designer can use not only own analytical capabilities and knowledge but other experience connected with tactile, auditory, visual and so on sensations.

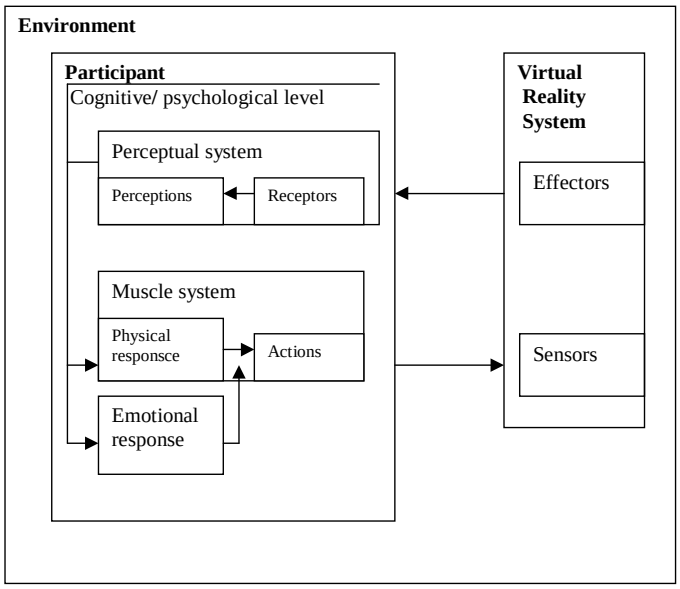
For the development of VR-interfaces (for example, for a direct manipulation in VR) many ideas and methods from robotics have been used. Therefore researches in VR-technology and advanced robotics technology can be very useful to each other.

There is another reason why VR-systems have been so widely used. *The virtual world created by a VR-system satisfies a number of human desires or even instincts, for example amusement, curiosity, investigation, creation and self-manifestation*.

Discuss now the fundamental problems in developing and realizing VR technologies, main components of VR-systems, its essential properties and differences from traditional human-computer systems and problem solving tools.

The structure of interaction and main components of VR-systems

A VR-system supports the creation and experience of environment. Main goal of the system is to place a user in an environment and to establish relationships between him and created environment. These relationships are supported by VR-interfaces. A human-computer interaction process realized by a VR-system can be described from the two positions [35]: from the human point of view (see Fig.19-1,a) and from the VR-system view (see Fig.19-1,b).

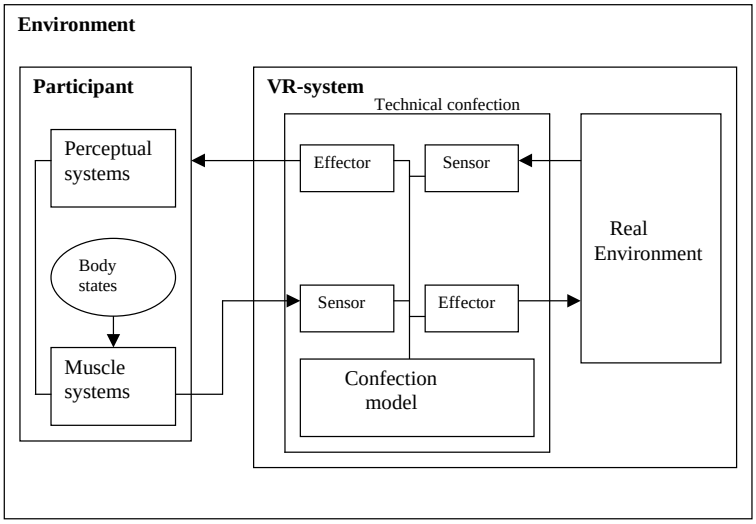


(a)

Figure 19 –1. Human-VR-system interaction : (a) the human view on the interaction.

The participant within an environment interacts with the VR system through effectors and sensors. The VR system uses effectors to stimulate the receptors of the human perceptual system. These effectors can include, for example, a helmet-mounted display, a force-feedback device, or even an odor emitter. The system detects the participant's actions, as expressed by the muscle systems, with sensors. This human-centered view characterizes the participant in the center of the environment.

In Fig.19-1,(b) a human-computer interaction process realized by a VR-system is shown from the VR system view (technical view).



(b)

Figure –1. Human-VR-system interaction: (b) the VR-system view on the interaction.

Figure 19–1,b shows a new component of VR system called the *technical confection.* Confection is the process of preparing or making, especially by combining. VR-system combines the artificial environment, the participant and the real environment into one *participatory environment*. It establishes the relationship between the sensors and effectors for the participant and the environment.

Figure 19-2 presents functional components of the confection model which is the central in determining the participatory environment.

There are two control functions: one for interfaces and one for environments. Environment control implements the combination of real and artificial environments. The artificial environment comprises two elements:

1. *Space definition*, including definitions of volume, spatial relations, objects, time, physical laws and material flow;
2. *Environment management*, including dynamics, elimination, creation, scaling, sampling, and time control.

Remark 1. The definition of the artificial environment includes all space elements, however, it might also include the rules for environment management. The *flight simulator industry* has developed technology to define *artificial worlds* as databases.

Mediation module is applied when VR-system operates as an interface between the participant and the real environment. *Representation* module refers to different forms the participant can take within the artificial or real environment. It includes agents, robots, or personifications with such design parameters as autonomy, intelligence, and quantity.

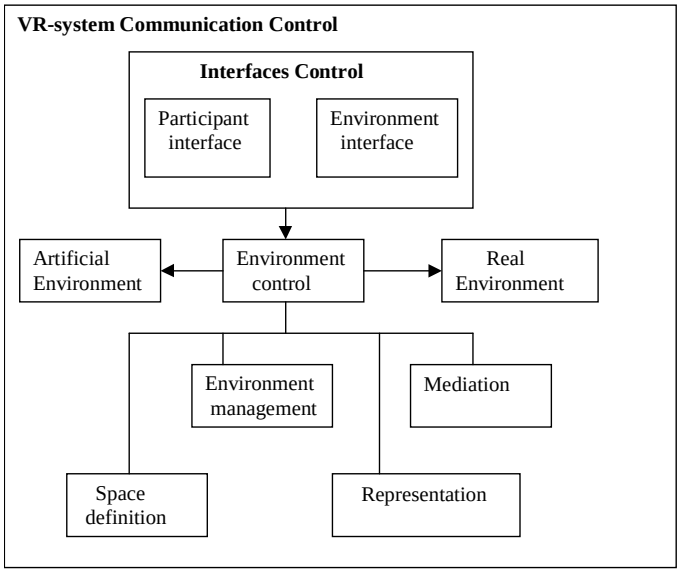


Figure 19 –2. Functional components of the VR-system control module.

The confection model can support independent models for each perceptual system. It also supports independent models for each muscle system, but the participant's detection of action is usually correlated with perceptual systems.
Consider the following parameters that define VR system properties:

- interface mapping;
- world model source;
- time and space definition.

Figure 19-3 represents example interface mappings for participant perceptual and muscle systems. We do not intend that this list to be limiting, rather, it illustrates types of VR interfaces.

A world model source describes the space changing. There are three main types of world model sources: dynamic, constructed, and recorded. A dynamic world model source describes the space changes based on the participant's actions, while a constructed model is defined a priori as a fixed space and objects. It is a static model database. Recorded models are based on time recording of the space. They capture a specific sequence of actions for later replay by the participant.

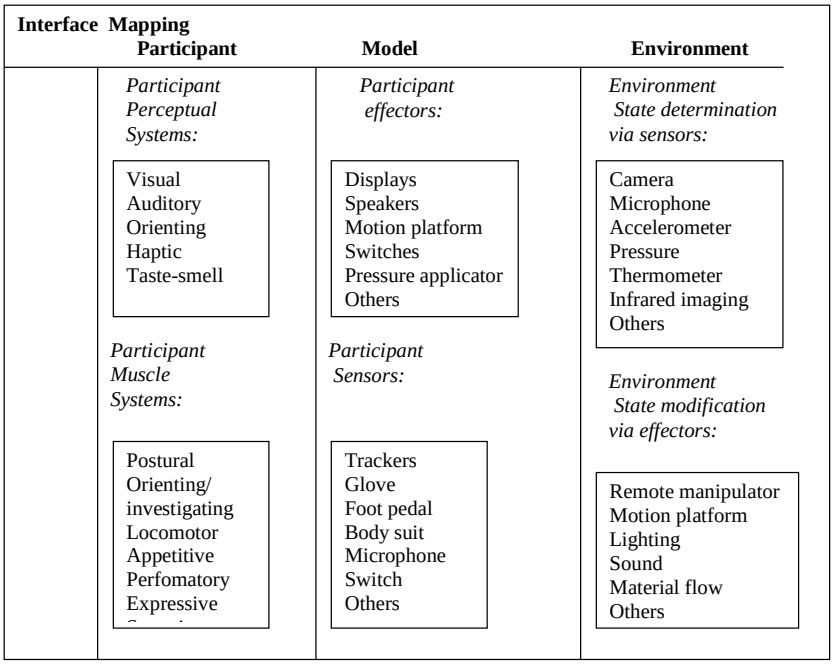


Figure 19- 3. Interface mapping example.

Figure 19–4 lists the parameters that allow the confection model to change time and space properties of real or artificial environments. An example of time alteration is a flight simulator that lets participants "jump ahead" to a desired point without flying the whole way there.
The direct parameters reflect the participant's natural experience of space and time. Manipulation of the space parameter is illustrated by microteleoperation, where the participant's perspective and operations are reduced to the size of molecules.

VR-engineering problems

Both the quality of the experience in VR and the participant's ability to function effectively are determined by how well the individual perceives the participatory environment. The participatory environment is created through a spectrum of stimulations to perceptual systems and actions by muscle system.

(Copia modificata per una migliore consultazione on-line)

How to create the sensation of a full immersion in VR? - This is a central problem in the VR-technology. Therefore, the main task of VR-engineering is the design of VR-interfaces to perceptual and muscle systems of a human being, which support human sensation of the VR and human action in the VR.

Each VR-interface consists of a sensor system (which receives and processes an information from environment including information from human effectors) and effector system (like feedback system) which generates a reaction of the VR-system.

According to the set of VR- interfaces a participant feels an immersion to the VR. For the design of such a kind of interfaces the understanding of similar human systems are needed.

Time	**Space**
Direct 1:1 correlation between time in the environment and the participant involvement	**Direct** (*x*,*y*) matching of the participant space and the environment
Multiple (*nt*) time modification between participant space and the environment	**Distance** (*mz*) distance scaling of the participant space and the environment
Fixed (*T*) fixed time between participant space and the environment	**Scaled** *f*(*x*,y,*z*) or (*nx*,*my*,*kz*) scaling of distance for the spatial dimensions between of the participant and the environment
Remapped *f*(*t*) functional remapping of time between participant space and the environment	**Functional** *f*(*t*) functional remapping of distance for the spatial dimensions between participant and the environment

Figure 19– 4. Time and space parameters.

A system view on a human perception was introduced by J. Gibson [36]. This approach helps to understand and simulate a human interaction with a VR-environment.

Human beings possess the six *perceptual (sensor) systems*: *visual, orienting, auditory, haptic, taste and smell* (Table 19-1) and seven *muscle (effector) systems* (Table 19-2):

postural (for maintaining of body equilibrium); *orienting* (for movements of body parts to obtain external stimulus); *locomotive* (for movements of body) , *performatory* (for performing of actions), *expressive* (for displaying of emotions) and *sematic* (movements for a voice expression).

The task of VR-technology is to develop interface technology, which integrates both perceptual and muscle systems of a human. Of course, it is impossible to attain a complete perceptual equivalence but there is a subjective threshold at which interaction within VR feel natural and VR-environment seems real.

Consider a contemporary development of VR interfaces and problems connected with their design.

For the direct manipulation with VR objects are developed a systems including *electronic gloves* (so named *Data Glove*) and *Hand-tracking devices*.

There are a few design of Data Glove, for example VPL Data Glove (produced by the VPL Co., USA). Physically this Data Glove consists of a lightweight Lycra glove fitted with specially treated optical fibers along the backs of the finger [37]. Finger flexion bends the fibers, attenuating the light they transmit. The signal strength for each of the fiber is sent to a processor that determines joint angles between adjacent finger.

Table 19-1. Human's perceptual (sensor) systems.

System	*Mode – of – activity*	*Receptive Units*	*Organ anatomy*	*Organ activity*	*Stimuli*	*External information*
Orienting	*Posturing and orienting*	*Mechanical and gravity receptors*	*Vestibular organs*	*Body equilibrium and balance*	*Forces of gravity and acceleration*	*Direction of gravity*
Auditory	*Listening*	*Mechanical receptors*	*Cochlear organs*	*Orienting to sounds*	*Vibrations in the air*	*nature and location of vibratory events*
Haptic	*Touching*	*Mechanical, thermal, etc. receptors*	*Skin, joints, muscles, tendons*	*Exploration*	*Deformation of tissues, etc.*	*contact with object surfaces and shapes, solidity, cold viscosity, heat*
taste – smell	*Tasting* *Smelling*	*Chemical and mecha - nical receptors*	*Mouth* *Nose*	*Savoring* *Snifting*	*chemistry of ingested objects* *Chemistry of vapors*	*Biochemical values* *Nature of odors*
visual	*Seeing*	*photo receptos*	*Ocular mechanism and eyes*	*Accomodation adjustment, scanning*	*light*	*size, shape dis tan ce, location, color, texture and movements*

A 3D-space magnetic tracker attached to the back of the hand determines position and orientation of palm. Hand-tracking systems design has now developed technologies borrowed from robotics [37]. The above mentioned Data Gloves have very restricted capabilities, because can support only some simple kind of hand motion like *grasping and moving of VR objects or gestures*. The problem of support complex tasks of object's manipulation in VR is now under research.

The next step in design of VR interfaces is development of *Data Suit* devices, which can reproduce the sensation from motion of a human body in VR. The design of the Data Suit devices may be based on the using of *exosceleton* (developed in robotics).

For reproducing of tactile sensations by direct manipulation in VR some kind of *Touch (or Tactile) Displays* are developed [38].

Tactile display can be for example, a force feedback system reproducing the object contact strength sensations. Note that such display used also in robotics for dexterous teleoperation tasks. So, the problem of touch sensations modeling is very important both in VR and in advanced robotics technologies.

Table 19-2. Human's muscle (effector) system.

System	*Purpose*	*Application*	*Other systems*
Postural	*Orientation with gravity and acceleration forces*	*Maintain body equilibrium*	*Vestibular organs*
Orienting - investigating	*Movement of body parts to obtain external stimulus*	*Sense information or explore*	*All other senses*
Locomotor	*Movements of body or body parts*	*Go from one location to another*	*Orienting - investigating and postural*
Appetitive	*Movement of body parts to take from or give to*	*Ingest or relieve*	*Taste, ingestion, and other body functions*
perfomatory	*movements beneficial to the individual*	*Take, move, protect self*	*Locomotor and others*
Expressive	*Movements to express self, display emotion*	*Make postural, facial and vocal movements*	*Voice, hearing facial muscles*
Sematic	*Movements to signal actions, state, or expression*	*Voice expression*	*Any other system based on signal intents*

The development of robot hands now is at the stage when a design like a human arm is appeared but capability like a human arm is not. Robots systems have not yet reproduced the tactile sensing abilities of the human hand, and research in this direction is needed.

There is the physiological model of human tactile sensing based on a set of human receptors and their basic mechanisms [39]. This model can be the conceptual base for development of tactile displays for VR-systems and advanced robotics. Touch Display can consist of a few receptors and its control systems.

Each receptor is a specialized cell that transmits signals about its environment state or change of environment state to its control system. For example, vibration, contact, heat can provide the necessary stimuli to excite a receptor. Each receptor responds to only one of stimulus and works like a filter. That is, a pain receptor is excited only when intensity of a stimulus exceeds a threshold.

The development of systems reproducing different tactile sensations (perception of contact, temperature, pressure, pain, texture and so on) are the important problem in VR engineering. It is also interesting to simulate sensations arised during walking or running (perceptions of body's movements). This is very complex task and its solution requires the activation of many kinds of researches.

For reproducing of auditory sensations different *acoustic displays* [40] are developed which generate different kinds of acoustic signals in real time. The main problem here is *the synchronization of auditory and other kind of information (for example, visual*).

For reproducing of taste and smell sensations special devices are needed, for example like *odor emitter*. This problem is now a little developed and well ideas are needed.

For reproducing of visual sensations the visual system is used. It is a very important perceptual system for a human. Therefore the design of visual VR interfaces is a primary task in VR-technology.

Mainly according to visual perception system the feelings of immersion in VR are formed.

How to create the visual display similar to human vision system?

Here we encounter with the problem of understanding what is a human vision system.

Many researches including *painters and psychologists* are examined this problem.

In the VR-technology this problem is solved be the means of *special visual display design* and *realistic computer graphics techniques* which are used for visual perception and generation of VR-environment.

There are a few kind of visual displays for VR: *Large Screen Monitors*; *Helmet-Mounted Displays* - HMD; *Stereoscopic Displays like glasses*, *Virtual Retinal Displays* [41].

Realistic computer graphics is a computer graphic, which supports the following aspects:

1) *human visual perception properties including the perception of space;*
2) *orientation and position (of body and others objects) in space;*
3) *perception of moving in space, including the direction of moving and others.*

Discuss now what are human visual perception properties.

For human perception of a space the following five factors are important:

- perception of third dimension (or perception of space depth);
- binocular parallax (or binocular disparity);
- accomodation (eye focus change);
- perception of environment scenes according to the direction of human attention ;
- perception of environment illumination and light propagation (including shadows).

The first and second factors are supported by a special stereoscopic and virtual retinal displays, another are supported by a realistic computer graphics system. Stereoscopic and retinal displays enable the left and right eyes of a human to see two separate views of the same object at the same time. The human brain reconstructs these two images in a 3D image.

Why the binocular vision system is needed in VR technology?

There are two reasons: 1) it provides a metric for environmental scaling and a participant can define spatial relations existing in VR scene; 2) binocular vision is needed for precision of manipulative actions [42].

Very important problem in design of VR control system is a problem of defining of the participant's attention in the environment scene.

Eye movement is *one capability to actively control or obtain external information*. For this task the design of active attention control system and full-field of view visual displays is needed.

Remark 2. Note that models of VR-interfaces supporting a natural way communication with a participant in VR can be also used for the development of new types of robots in advanced robotics like Humanoid robots.

Let us return to the general structure of VR system and discuss main component of VR-systems software.

Software of VR-system consists of the following main components:

- *Interfaces Control System*,
- *VR-Environment Simulation and Generation Systems*,
- *Cooperative Interaction* (between different kinds of interfaces) *Control System*,
- *Environment Change Control System*; and
- *Knowledge Base*.

The environment simulation component is an important part of any VR-systems. At each step defined by the passing of a time in the VR this module processes the information from the environment change control system and creates new representation of the environment based on the model of world change (for example, by using spatio-temporal logics, qualitative physics, some laws in the VR like day and night change and so on).

The *environment generation component* supports a *realistic generation of VR in real time.*

The *system of cooperative control* supports the *synchronization and integration of different kind of information* in processes of generation or perception. For example, the object movements must be conciliated with the level of sound signal from this object; or verbal information (for example, some natural language message) must be conciliate with visual image which a participant is watching.

The knowledge base contains all necessary information for VR-environment simulation including spatio -temporal, physical, action models and others.

Essential features of VR-technology

Finally, let us write essential features of VR-technology which are:

1) The immersion (or presence) effect;
2) A direct manipulation with VR objects in real time ;
3) The support of perception based on human perception criteria;
4) The support of human motor capabilities in VR including locomotion in space;
5) The support of exploration and engagement in VR.

These features make a VR-system qualitatively different from the traditional human-computer system. The possibility to immerse in a VR allows to a human - designer (or a human-operator and so on) to use natural (and more simple methods) way of investigation (or control) based on the direct observation and action in the VR.

So, *instead of a complex mathematical model of description of some tasks or behavior, a human can use own experience based on the direct perception and activity in the modeling world.*

Remark 3. There is one psychological problem connected with VR technology. We create a new technology, which in the next future may change human living style or cultures.

This implies that researchers must seriously think about neither large potentiality of VR technology nor its negative effects. VR is not indifferent for human mentality. VR creates a special mental state of a participant which in real situation is not often (or never) occur. The base of such states is the sensation that you can do all what you want in VR, you are Got. The participant begin to think that he has huge capabilities and can overcome all difficulty. This creates the feelings that you have superiority with respect to VR, and your behavior infallible and you can not be punishable.

Some human (especially, children) may continue such behavior in real life. Or may be cases when human beings immersed in VR don't want to return from VR into reality. All mentioned above are the negative effects of VR technology. Therefore serious psychological researches connected with VR influence on human mentality are needed. Note that first results in this area are alerted: psychologists are shown that some of the high level programmers named hakers have nonnormative behavior.

VR -technology application problems in robotics

Many companies now are interested in VR technology as a new problem solving tool. In robotics the problem of design new intelligent control systems including biological mechanisms such as intuition, instinct, emotions needs the development of new approaches and tools. VR-technology allows to a designer of intelligent control system for robot to simulate different virtual worlds and examine different algorithms and behavior model of robot. There are some attempts of VR technology application in design of remote control system for robot [34]. The system based on robot's camera images and computer graphics synthesizes VR-environment for an operator. The operator wears stereoscopic glasses and immersed into VR. This allows to him to feel a real control site conditions. So operator can control the robot located in very far distance from him much more better than by the traditional way.

The use of VR technology for development and investigation of different model of robot's behavior in their artificial life is also became very important. In this direction first steps are made and a new interesting solution are needed.

Finally, we would like to say that VR technology and its application now is a new branch of research where experimental prototypes are developed. This technology is rather expensive and its application must be justified.

Example of VR system application: Modeling Evacuation in VR.

It's late Thursday afternoon in one big shopping center in London. A fire breaks out. Some people run directly for exits, parents try to gather their children, and still others try to investigate the fire to see what can be done. At the next exit, the crowd pushes to get through a doorway. Their progress slows as they try to squeeze through. But a click on a mouse button and a drag of the doorjamb widens the doorway. Twice as many people now flow through.

This is example of VR-system (called *Vegas*) application for real time evacuation scenarios [43]. *Vegas* creates virtual shopping environment and shows how quickly fire and calm action can be in an emergency. Each character has been assigned a random generator. When a fire breaks out, the number generated at that moment will dictate whether the character heads for a familiar exit, stops to gather family members, or goes to investigate the trouble.

Vegas is based on *Superscape*'s application for developing VR-environment. In Superscape a high-level C-type language SCL (Superscape Control Language) is used to assign behavioral or physical characteristics to objects. Superscape's editor includes a library of attributes to which you can assign values, or you can script of your own using SCL. SCL is used also to assign constraints and behavior to objects – specifically, people, fires, and smoke – in emergency situation. The fire's parameters are limited its size and spreading speed. *Vegas* is used to investigate different models of people behavior in emergency situation (a crowd trying to squeeze through a doorway, and so on) in order to find optimal evacuation scenario in real situations.

(Copia modificata per una migliore consultazione on-line)

Embodied Conversational Interface Agents

Animals and humans all manifest social qualities and skills. Dogs recognize dominance and submission, strand corrected by consistent personalities, and so forth. On the other hand, only humans communicate through language and carry on conversations with one another. The skills involved in human conversation have developed in such a way as to exploit all the special characteristics of the human body. We make complex representational gestures with our prehensile hands, gaze away and toward one another out of the corners of our centrally set eyes, and use the pitch and melody of our flexible voices to emphasize and clarify what we are saying.

Perhaps because conversation is so defining of humanness and human interaction, the metaphor of face-to-face conversation has been applied to human-computer interface design for quite some time. One of the early arguments for the utility of this metaphor pointed to the application of the features of face-to-face conversation in human-computer interaction, including mixed initiative, nonverbal communication, sense of presence, and the rules involved in transferring control. However, although these features have gained widespread recognition, human-computer conversation has only recently become more than a metaphor. That is, only recently have human-computer interface designers taken the metaphor seriously enough to attempt to design a computer that could hold up its end of the conversation with a human user [44].

Consider some of the features of human-human conversation being implemented in this new embodied conversational agent – named Rea [44]. Because conversation is such a primary skill for humans and learned so early in life (practiced, in fact between infants and their mothers taking turns cooing and burbling to one another), and because the human body is so nicely equipped to support conversation, embodied conversational agents may turn out to be a powerful way for humans to interact with computers. However, in order to embodied conversational agents to live up to this promise, their implementation must be based on the study of human-human conversation, and their architectures reflects some of the intrinsic properties found there.

Embodied conversational interfaces are not just computer interfaces represented by way of human or animal bodies. They are just interfaces in which human or animal bodies appear lifelike or believable in their actions and their reactions to human users. *Embodied conversational agent interfaces are specifically conversational in their behaviors and specifically humanlike in the way they use their bodies in conversation.* That is, they may be defined as having the same properties as humans in face-to-face conversation, including four very human abilities:

- recognizing and responding to verbal and nonverbal input;
- generating verbal and nonverbal output;
- dealing with conversational functions, such as turn-taking, feedback, and repair mechanisms;
- giving signals that indicate the state of the conversation, as well as contributing new propositions to the discourse.

Embodied conversational agents represent a new slant on the argument about whether it is wise to anthropomorphize the interface. Clifford Nass and Byron Reeves of Stanford University and others have shown that humans respond to computers as if they were social entities [44]. Even experienced computer users interact with their computers according to social rules of politeness and gender stereotypes and view some computers as more authoritative than others, even that some are experts and others generalists. In many other ways, users react to the computer as if it were another human. Nevertheless, the fact that we react to computers in this way begs the question of whether interface designers should accede to such illogical tendencies by building computers that look like humans. Critics have asked what function building humanlike computers really serves, pointing out that anthropomorphized interfaces have never succeeded in the past and may even lead to slower user response times and confusion.

We might say that only conversational embodiment – giving the interface the appearance and the function of the human body in conversation – allows us to evaluate the function of embodiment in the interface. Simply building anthropomorphized interfaces that talk (but don't use their talk in humanlike ways) sheds no light on the debate about embodiment. However, well-designed embodied conversational interface agents address particular needs not met in conversational keyboard, mouse, and screen interfaces. Such needs include making dialogue systems robust despite imperfect speech recognition, increasing bandwidth at low cost, and supporting efficient collaboration between humans and machines and between humans mediated by machines. These missing ingredients in conventional human-computer interfaces are exactly what bodies bring to conversation.

Embodied conversational agents also bring a new dimension to discussions about the relationship between emulation and simulation, as well as to the role of foundational principles in interface design, that are true to real-world phenomena in the interfaces consumers actually buy. The first wave of interface agents with bodies and autonomous embodied characteristics in the early 1990s – often called autonomous synthetic characters – did not focus on conversation but on more general interactional social skills. Researches developing these characters discovered, sometimes to their surprise, that the best way to produce believability and lifelikeness may not be through the modeling of life. They found themselves turning to insights from Disney animators and other artists about caricaturization and exaggeration as a way of getting users to suspend disbelief and attribute reality to interactive characters. Here, like the scientists who first began to built dialogue systems to allow computers to understand human language, researches are finding they have to turn to theories of human-human interaction and investigate the nitty-gritty details of conversation to ensure their interfaces share the conversational skills of human users. A lot of researches introduced particular pointing gestures into conversation process.

We will not debate whether anthropomorphization is good or bad. Instead, we will consider one implementation of human conversational behavior into the embodied conversational interface agent called Rea [44]. Rea architecture is shown in Fig.19-3.

Rea has a humanlike body and uses humanlike ways during a conversation, that is, she uses eye gaze, body posture, hand gestures, and facial displays to contribute, to organize and to regulate the conversation. Rea's domain of expertise is real estate, she has access to a database of available condominiums and houses for sale in Boston. She can display pictures of these houses and their various rooms and point out and discuss their salient features.

The following dialog is an example of interaction with Rea.

Rea: Hello. How can I help you?
User: I'm looking to buy a place near MIT.
{ Rea nods indicating that she is following}
Rea: I have a house to show you.
{A picture of the house appears on the screen behind Rea, who blinks and looks at the house and then at the user}
Rea: It's in Somerville.
User: Tell me about it.
{Rea looks up and away while she plans what to say}
Rea: It's big.
{Rea makes an expansive gesture with her hands.}
User: Tell me more about it.
Rea: Sure. It has a nice garden.
{Rea sketches a curved gesture with her hands, indicating the garden extends along two sides of the house.}
User: How far is it?
Rea: It is five minutes to the Porter Square station.
{Rea makes it clear it is five minutes on foot from the station by making a walking gesture with her fingers.}
User: How big is the house?
Rea: It has four bedrooms, three bathrooms…
{User interrupts Rea, who stops speaking immediately.}
User: Wait. Tell me, where is the master bedroom?
Rea: I'm sorry, I didn't catch that. What did you ask me?
User:Where is the master bedroom?
Rea: It's upstairs.
{Rea points up.}
User: Where is the master bathroom?
Rea: It's next to the bedroom.
{Rea brings her hands together to indicate the relationship between the bedroom and the bathroom.}
and so on.

The set of behaviors exhibited by the people doing a conversation differs from person to person and from conversation to conversation. Therefore, to build a conversational model for artificial agents like Rea we need to identify high-level structural elements that make up a conversation. Table 19-3 shows examples of conversational functions and how they are represented in body behavior (taken from Cassel [44]).

Table 19-3

Functions	Behaviors
Initiation and termination React to new person	 Short glance at other
Break away from conversation	Glance around
Farewell	Look at other, head nod, wave
Turn-talking Give turn	 Look, raise eyebrows (followed by silence)
Want turn	Raise hands into gesture space
Take turn	Glance away, start talking
Feedback Request feedback	 Look at other, raise eyebrows
Give feedback	Look at other, nod head

Finally, let us talk about factors motivating interface designer to develop embodied conversational agents. One of them is increasing computational capacity in many objects and environments beyond the desktop computer, such as smart rooms, intelligent toys, children's museums, etc. Another one is intelligent robotics, where human-robot interaction tends to be maximally close to human-human interaction. Computers and robots without keyboards with face-to-face interaction will become very useful in nearest future.

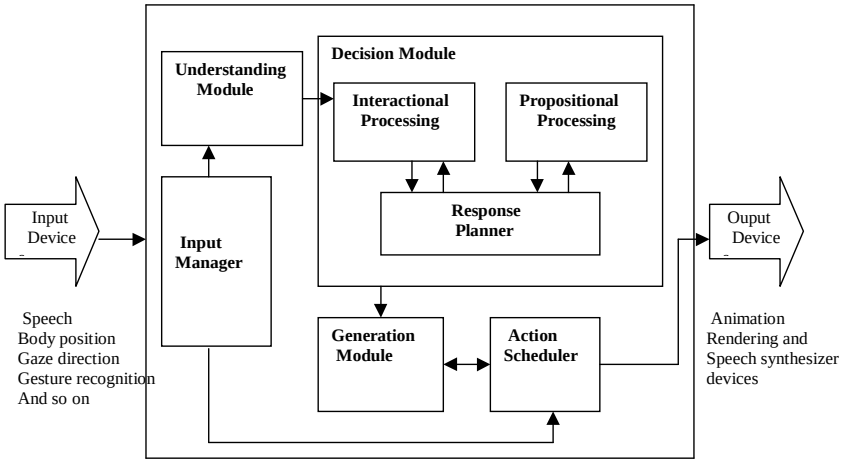


Figure 19-3. The Rea architecture.

Lecture N 20 Simulation Problems of Computer's Creation Skills

In the last of our lectures let us talk about *what is a creation?*

There is no formal definition of creation, and now we can not completely understand what is it. We admire Bach, Picasso, Shakespeare or Einstein. We can not imagine how they create their things, and this reinforces our admiration. But may be the process of creation is not so mysterious as it seems.

Let us try to formulate *essential aspects of creation process*.

1. In significant degree the *creation is a process of finding new relations* there where nobody did not see.

For example, a poet finds such relations between things that nobody did not think about that, and these new relationships surprise our brain and mind.

2. Another form of creation is a *process of generation of new structures* which are more complex than previous one.

3. Creation may be considered also as a *process of forming of new ideas.*

There are three levels of creation process: *semantic, syntactical and sign*. At semantic level we may introduce new nonstandard interpretations of old notions. On syntactical level we may change rules according which we relate different words and notions. At the sign level we may introduce new sign systems.

Any human being is capable to creation, not only genius peoples. Continuing this idea we may say that computers may also be capable to creation. Consider some simple examples of computer creations and discuss some creative models for computers.

Machine poems generation model

Let us try to construct an algorithm for machine poems generation. In order to simulate poetry creation let us begin from analysis of a structure of a verse. Consider the following main components of a verse generation process:

grammar + metrics + rhyme + semantics.

In order to generate automatically a verse we must firstly introduce grammatical structures of sentences of the generated verse.

Let a sentence be generated by the following grammar:

S → NG+ NV+ NG + Loc/Time

NG → ART + ADJ + NUM + N/ PRO+ ADJ + N

NV → V + PRE / V

Loc/Time → NG/ PRE+NG ,

where nonterminal symbols mean the following syntactic categories:

NG- noun group, N – a noun, ART – an article, PRE – a preposition, Num – a numeral, PRO – a pronoun, Loc/Time - groups of words for description of times and places.

(Copia modificata per una migliore consultazione on-line)

Introduce terminal symbols (words) for N, ART, PRE, Num, PRO, and Loc/Time.
Consider the dictionary with the following parts:
((Nouns: (<NL-entry>, <grammatical information> <metrics-information> <rhyme-information >))
(Verbs: (<NL-entry>, <grammatical-information> <metrics-information> <rhyme-information >))
(PRO: (<NL-entry>, <grammatical-information> <metrics-information> <rhyme-information >))
(PRE: (<NL-entry >, <grammatical information>))
(NUM: (<NL-entry >, <grammatical-information> <metrics-information> <rhyme-information >))
(ADJ: (<NL-entry >, <grammatical information> <metrics-information> <rhyme-information >))
(Loc/Time: (<NL-entry >, <grammatical information> <metrics-information> <rhyme-information >))).

Grammatical information of nouns includes information about their *gender* (masculine, feminine, neuter or neutral) and *type* (singular, plural, or neutral).
Grammatical information of verbs contains information about their *personality* (I, you, he (singular) and we, they (plural)) and *time* (present, past, future, or neutral) .
Grammatical information of adjective includes information about their *gender* (masculine, feminine, neuter or neutral) and *type* (singular, plural, or neutral).

Remark 1. Different languages have different grammatical information, for example, in English the grammatical information like gender is absent, in Russian and Italian languages, information about gender is needed.

How to represent information about a metrics and a rhyme?

Consider some word, for example "location". Define the *number of syllables* and the *point of the stress* (accent) in this word: lo/ca/ti/on.
The number of syllables in this word is 4, the point of the stress in this word is the second syllable.

Information about a metrics we will describe by two *size numbers* $< n_1, n_2 >$, where n_1 is a number of syllables before the stress, and n_2 is a number of syllables after the stress. For our example we will have $n_1 = 1$ and $n_2 = 2$.

So, the total number of syllables in a word is equal $n_1 + n_2 + 1$.

Different types of verses are characterized by different sequences of stressed and unstressed syllables.
Definition: The sequence of stressed and unstressed syllables in one line of a verse defines the *metrics* (or *rhythm*) of the verse.

Example. Consider the following verses fragments:

1) When you make investigation You collect much information Then you do a lot of thinking (For the time keep off you drinking).	2) Archimedes plunged in water Landing safely on tub's bottom But seeing water on the floor He discovered a new law.
3) My advice: avoid confusion, Controversy and diffusion, Classifying all the factors And arranging all the matters.	4) A meta della nostra viva Mi sono ritrovato in un bosco scuro Perche avevo perso la strada.

Remark 2. The fragments 1) ,2) and 3) are taken from English songs, fragment 4) is taken from Italian poem of Dante Alighieri ("Divina Commedia").

Define the structure of these fragments as a sequence of stressed (/) and unstressed (-) syllables. The structure of the first fragment can be represented as follows:
/- / - / - / - (first line in the first fragment) (20-1)
/ - / - / - / - (second line in the first fragment), and so on,
where " -" means the unstressed syllable, "/" means the stressed syllable.
The verse's metrics with the structure (20-1) is called iambus.

Consider the following fragment of the verse of Pushkin (Russian poet) translated into Italian:

Ricordo il magico istante:
Davanti m'eri apparsa tu,
Come fuggevole visione,
Genio di limpida belta.

The structure of this fragment is represented as:

- / - / - - - / -
- / - / - - / - -
/ - - / - / - - - -/ -
/ - - / - - - / -

Remark 2. In Russian language this fragment is written by iambus, but the metrics of translated fragment is not iambus.

There are different types of verse's metrics, for example, trochee, dactyl, hexameter, etc.

Discuss now *how to represent information about rhyme*. There are two types of rhyme: men's rhyme and women's rhyme. If *the stress* in a last word in a verse line occurs in a *last syllable*, then a rhyme is called the *men's rhyme*. If *the stress* in a last word in a verse line occurs in a *not last syllable*, then a rhyme is called the *women's rhyme*.

In the verse fragment 1 mentioned above all lines have women's rhyme.

Rules of rhyming

Introduce the following rules of rhyming:
1) *If two words are lasted by the same syllables, they can be rhymed.*
2) *If two words are lasted as shown in special Rhyming Table they can be rhymed.*
Rhyming Table can be, for example, as shown in Table 20-1.

Table 20-1. Rhyming Rules Examples.

Last symbols of two words	*Examples*
g,k	*dog,shock*
se, ys	*base,days*
er, om	*water,botom*
or,aw	*floor,law*
and so on...	- - -

(Copia modificata per una migliore consultazione on-line)

Remark 3. For any natural language the rules of rhyming may be found from the analysis of different verses generated by human beings.
So, we described the dictionary and the Knowledge Base (a set of rhyming rules) for our verse creation computer program.
Let us now define *input parameters* for program as follows:
- the number of lines in a strophe,
- the number of syllables in a line,
- men's or women's rhyme of last word in a line,
- the order of lines rhyming in the strophe(for, example, 1 and 3, 2 and 4 , or 1 and 2 , 3 and 4),
- metrics of a verse.

Remark 4. A strophe is a fragment of a verse consisting of some number of lines. For example, in the first verse fragment given above a strophe consists of four lines.

Let us look at *general steps of computer generation of a verse*.
1. Randomly choose one word from dictionary (with a given men or women rhyme) which can be the last word in a first line.
2. Find a last word of the next verse line which is rhymed with the first line.
3. Repeat steps 1 and 2 until a construction of a strophe with last words is finished.

For example,

- - - water
- - - bottom
- - - floor
- - - law.

If the program can not find rhymes for first word, then it can randomly choose another word and repeat.
4. Generate words in each line beginning from a right side of the line.

Let the last word in a line is w_1 which is characterized by size numbers n_{11} and n_{12}. Other words in the line are chosen randomly and according to a grammar and a given total number of syllables in the line. For example, let the next randomly chosen word w_i in the first line is characterized by the size numbers n_{i1} and n_{i2}.
Let us check available or not this word for our constructed verse.

Firstly, check if $n_{i1} + n_{i2}$ > *the given number of syllables* in the line?

If *yes*, then the word w_i is deleted, and next random choose is repeated.

If *no*, then check the number of syllables between two neighboring stresses equal to

($n_{i1} + n_{i2}$).

If the program generates *iambus* or *trochee* verses, then the word w_i is *available* if the value ($n_{i1} + n_{i2}$) is a *noneven number* (i.e., it is not divided on 2).

5. Checking grammatical properties of the word w_i .
6. When all words in the line are found, then the next line is generated.

Of course, we discuss here minimal tools needed for a verse generation. Machine verses are very far from human one.
To improve the quality of computer verses we must introduce *some heuristics describing creation laws.*
For example, the following heuristics will be useful:
1) *one syllable prepositions (or words) is better to put in a stress position in a line. In this case we have a special sounding of a verse.*
2) check repetition of words in a verse, *there must be no same words in the verse.*

One of first computer programs generating verses was the *program of Robin Shirley* (UK) written in 1972 [45].

Machine Literature Creation Modeling

In designing of a story generation computer program we encounter with the problem of generation of *connected sensed texts.*
Gunzenhauser R. (Germany, 1963) was one of first researches who wrote a program of sensed sentences generation. His program was based on a dictionary (65000 words) and a mechanism of a random generation of word's sequences with given syntactical structures. But how to generate semantically connected text?
For this purpose we will consider a *text as a sequence of some general structures*, and introduce a mechanism for generation of these structures, and then will fill given structures by words. In such task definition we encounter with the following question: *how to find these structures for some thematic area?*
We can introduce these structures for the area of *fairy tales writing.*
In the end of 1920, Russian philologist V. Propp investigated structures of all fairy tales and constructed the *functional model of any fairy tale*. We may use now this model for our computer generation.
Consider the structure of traditional fairy tale.
A fairy tale consists of three main parts: *exposition, body,* and *postposition*.
The *exposition* is an initial part of a tale, the *postposition* is a final one. The *Body* of a tale is a main part describing a subject of the tale.
V. Propp showed that *every fairy tale part can be described by finite numbers of structures.*
Figure 20-1 shows a semantic network describing exposition part of a traditional fairy tale. Any path from node 1 to node 7 describes one possible tale's beginning.
Exposition part introduces initial information about main actor (called a hero), a time and a place of her birth and life, another actors, and a so-called "*starting event"* (or starting situation). *Starting event* introduces a purpose of future actions of a hero. A *body of a tale* can be represented by a sequence of *meetings* (*key events* in a tale), which help to the hero reach the goal (for example, to kill an antihero and married on a princess). *There is a finite number of meetings*. For example, meeting of hero with antihero, meeting of hero with helpers and antihelpers, meeting of hero with her award (princess), etc. Each meeting structure is described by a set of actions characterizing this meeting. Meetings are connected by some additional elements.
For writing stories we must describe also actors and their properties. In order to describe actions and events we will use *frames* structures. For example, the event "birth of hero"(which may be considered as starting event of a tale) may be represented as follows:
(birth:
who = <person> ,
where = <place> ,
time = <data>,
parents = (<person>,<person>)
condition = <events>).

(Copia modificata per una migliore consultazione on-line)

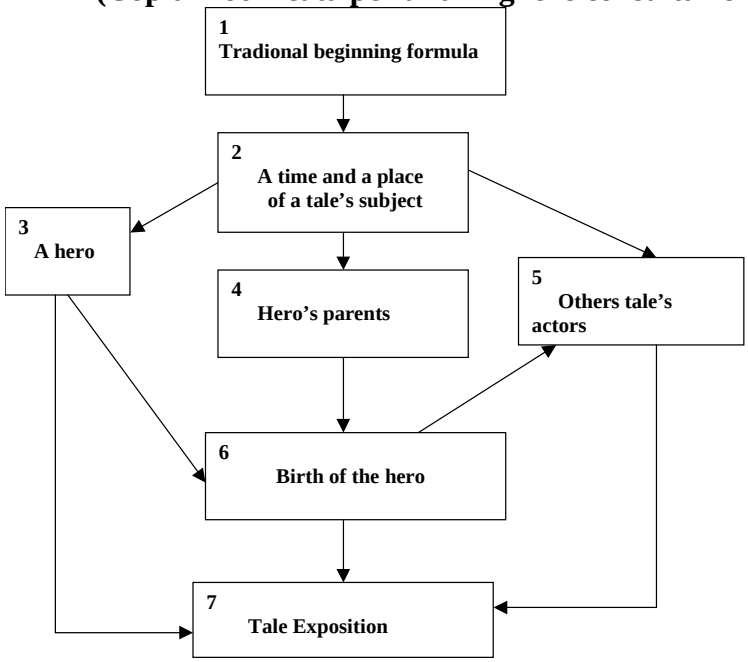


Figure 20-1. Semantic network describing exposition part of a tale.

Postposition of a tale consists of an element called an *award* for a hero and some element called *morals*.
So, in order to generate a fairy tale we must have a knowledge base consisting of following descriptions:

- exposition structures,
- a set of possible actors with their possible roles (a kind magician, a hero, an antihero, a rogue , a princess, different helpers and antihelpers, etc.),
- a set of possible starting events,
- a set meetings,
- a set of actors actions ,
- a set of possible additional elements, and finally,
- a set of possible final propositions .

Now we can discuss main steps for fairy tale's generation.

1. Find the structure of a generated tale by choosing randomly an exposition part, body, and postposition structures.
2. Fill all structures by words and generate sentences for each tale structure.

Machine Music Generation Models

First programs for music generation were the interactive human-computer systems for music creation. In this case an interactive system plays a role of an assistant for a composer, and the output of that system is a set of possible fragments of a musical composition.
Composing by hand is very exhausting work, and many human-composers prefer to use a computer help.
One among first music composition programs was the program developed by Italian mathematician E.Galyardo in 1970.
Discuss briefly main ideas of music compositions creation.
There is one *general law of any creation activity* - the *variation (of situation) law*. The variation of situation is a process when some elements of the situation are changed and others are conserved. Unchanged elements are called *invariants.*
Transformation of invariant structures is a general and important property of thinking and creation processes.
The mechanism of transformation is performed by human beings unconsciously. *How to simulate this mechanism in computer creation algorithm?*
Consider the task for music composition generation based on the variation law.
Variation of melodies is a part of music composition (and not only music composition, for example, the same law is applied in poetry, literature, etc.).

Markov chains and computer music

Shenon was the first who introduced a simple model of note text analysis in order to investigate his statistical properties. He proposed the model of musical note texts based on Markov chains.
The main idea of music composition generation based on this model is following.
First step. Analysis of some number of melodies introduced in a computer. As a result all *n*-note combinations possible for these melodies and their statistical characteristics are defined. A so-called *transition matrix* is constructed based on obtained statistical observations. Elements of the transition matrix describe the probability that any (*n*-1)-note combination is combined with *n*-th note in *n*-note combination.
Second step. Synthesis of a melody note by note. (We will call it as a constructed composition.)
From all *n*-note compositions obtained during analysis, we choose a subset of compositions with the following property: first (*n*-1) notes of a composition are equal to last (*n*-1) notes of constructed composition. Then randomly one composition from the subset is chosen. And after that accordingly to the transformation matrix, *n*-th note is joined to the constructed melody. Then synthesis step is repeated for next note. (Remark. Usually, $n = 1,2,\ldots 8$.)
Third step. The chain of notes is correlated with the given rhythm and final melody is generated.
Of course, this method is too primitive in order to have successful results. More perspective methods are based on the structural methods of music modeling using rules and laws of music composition [46].

Variation-based computer music generation

Any type of composition may be described as a set of parameters. Two types of parameters are considered: important and unimportant.
For unimportant parameters their value is not introduced, while for important parameters a set of available values is defined. Music generation is based on the variation law.
At first step a given melody-theme is introduced as input. Special mechanism of transformation of initial theme into variations is performed. Then one of variation is chosen as a result.
By using the variation-based music generation model, the algorithm for the synthesis of music composition according with the given songs rhythm was developed by Russian mathematician R. Zaripov [46].

Evaluation of machine creation products

How to evaluate machine creation product, music, for example? This situation is like the Turing Test for AI systems. Experiments show that if a human expert did not know before what kind of music he evaluates, then the evaluation some times is positive.

(Copia modificata per una migliore consultazione on-line)

Computer Painting

In painting computers may be used in two directions: as instruments, or tools, for human painter, and as generators of different pictures, movies, etc. A lot of systems of computer graphic are developed now. Some systems can generate pictures at the very high level of art. A lot of examples of computer painting you may find in [45].

Art Tools Example: Music and Visual Images

Consider one example of art tools representing a computational system for translating musical compositions into a visual performance. The system is called "The 21th Century Virtual Reality Color Organ" (or VRCO) [49]. VRCO uses supercomputing power to produce 3D visual images and sound from Musical Instrument Digital Interface (MIDI) files and can play a variety of compositions. Visual performances take place in interactive, immersive, virtual reality environment installed by the Color Organ. The Color Organ implemented as C++ standard module which is based on the EAI Sense8 World Toolkit VR library (WTK) (Bell Labs). This library provides rich programming interface for rendering 3D objects in an immersive computer display.
VRCO consists of three basic parts:

- a set of systems or syntax that provides logarithmic transformations from an aural vocabulary to a visual one;
- a 3D visual environment that serves as a performance space and the visual vocabulary from which the 3D environment was modeled. This visual vocabulary consists of landscape and/or architectural images and provides the objects on which the syntax acts,;
- a programming environment that serves as the engine of interaction for the first two parts.

The primary system's function is the analysis of music. The analysis task is to determine the structural parameters of the piece of music to be visualized. The output of analysis stage contains data sets including patterns of rising and failing melodic lines, changes in dynamics (loudness) and rhythmic units, and patterns with the initial "attack" of the notes and their articulations. The Color Organ encodes this information in MIDI files.
Creation of corresponding data sets:
Special visual vocabulary, that is, images that express attributes of the music in a metaphorical way, has been developed. In the first step, eight different desert landscapes gathered from real places in California and Arizona will be chosen. Each image itself is a collection of data containing patterns of lines and colors, and also the metaphorical connection to attributes of the music. Changes of image scale represent louder or softer music. Melodic changes create shifts in the vertical placement of image units in the virtual reality 3D space.

When the performance begins, viewers are in a world of landscapes modeled in 3D. All the landscapes are black and white. The system creates a set of colored and textured planes dynamically, at the same time when music is playing. After the music has played, there remains a complete sculpture that can be further explored interactively. Viewers can move through the space and touch elements of the sculpture and hear the sound that originally produced it.
The artwork that has emerged from this process embodies principle of *intermedia* [49]. Intermedia is considered as a combinatory structure of syntactical elements that come from more than one medium but combine into one. The final form can only be seen after going through the entire creative process.

Post Scriptum

" The brain is rational, the mind may be not be. "
Douglas R. Hofstadter

Intelligent Behavior and Metaprocedures

There is a lot of success in AI technologies capable to solve different intelligent tasks. AI-based systems can play chess, understand Natural Language, can perform intelligent control, prove theorems, can be expert in many problem area, etc. Let us add one more capability, for example, creation of music, or love stories. Ok, we may develop an AI- system with this capability, but it will be one more AI-system.
By using a metaphor, we may say that each AI system is one key to open some definite door. A pack of keys correspond to a set of computer programs, but even the pack of keys can not open a door, which has an unknown lock. Instead of such pack of keys is better to have universal tools for opening any possible doors. That is, I mean that in our case, instead of different AI systems imitating different intelligent capabilities, will be better to design one intelligent system which can imitate universal procedures of any intelligent behavior. We will name such procedures as *metaprocedures*. One of them is the *problem solving procedure*.
Human beings have a capability to solve any problem (task). How to describe this *problem solving* capability?
There is a hypothesis called the labyrinth hypothesis describing the problem solving capability as a goal-oriented search in the labyrinth of possibilities. But by using the *metaprocedure of a goal-oriented search* only, we cannot solve any problem. There is some moment of a time when a human being solving a problem understands that he cannot find a solution by searching in the existing labyrinth. What is a next step? Psychologists say that there is a special state called "insight" when the human being finds the solution of the given problem. "Insight" implies a transition to another kind of possibilities labyrinth, where the unknown solution is contained. So, *the construction of a search labyrinth containing the solution of a problem* is a central *metaprocedure* of any creative capability [47].There is another, also very important, *metaprocedure* of intelligent activity of human beings - *discovery laws from observed data*. How much metaprocedures there exist? We in AI believe that there is a finite number of metaprocedures.

AI's Impact on Society

Mankind has dreamed for centuries of creating intelligent artificial creatures that can remove the drudgery from everyday life. Hence AI's impact on society can be measured by answering on the following questions: Do the products that result from AI help society at large?
AI technologies have been used successfully by industry in tasks involving analysis (for example, machine diagnosis), synthesis (such as design), planning and scheduling, simulation, decision-making, all of which lead to significant economic gains. Expert systems save over a billion dollars each year through this technology. Speech analysis and speech generation reach the market. Vision and robotics transform manufacturing. The computer and communication technologies make it possible for a rapid inexpensive sharing of knowledge.
Chinese philosopher Kuan-Fu once said: "If you give a fish to a man, you will feed him for a day. If you give him a fishing rod, you will feed him for life."
We can go one step further and say: "If we can provide him with the know-how for making that fishing rod, we can feed the whole village".
Therein lie the promise and the challenge of Artificial Intelligence.

Acknowledgments

We would like to thank our colleagues, especially Professor Giovanni Degli Antoni and Professor Nello Scarabotollo, for the invitation to work together in Polo Didattico e di Ricerca di Crema and for useful discussions and comments. We are also thankful to Dr. R.Yamashita, Dr. I. Kurawaki for supporting this work and to Dr. L. Zadeh for valuable discussions.

(Copia modificata per una migliore consultazione on-line)

AI Bibliography

References to PART 3:

1. Prior A.N. (1957 *Time and modality*. Oxford Univ. Press.
2. Wright G.H. (1965) *Von and next*. Acta Philosophica Fennica .
3. Russel B.(1967) On the notion of cause. *Proceedings of the Aristotelian Society*,1-26.
4. McCarthy J. (1986) Applications of circumscription to formalizing common sense Knowledge, *Artificial Intelligence* 28, 89-116.
5. Allen J.F.(1983) Maintaining knowledge about temporal intervals, *Communication of the ACM*, 26 832-843; Allen, J.F. (1984) Towards a general theory of action and time. *Artificial Intelligence* 23, 123-154.
6. Galton A. (1990) A critical examination of Allen's theory of action and time, *Artificial Intelligence*, 42, 159-188.
7. Kandrashina E.Yu., Litvintseva L.V. and Pospelov D.A. (1989) *Knowledge representation of time and space in intelligent systems* , "Nauka" Press, Moscow.
8. Qian D. (1992) Representation and use of imprecise temporal knowledge in dynamic systems, *Fuzzy sets and Systems*, 50 (1), 59-77.
9. Johnson, C.W. and Harrison, M.D. (1992) Using temporal logic to support the specification and prototyping of interactive control system, *Int. Journal of Man-Machine Studies*, 37 (3), 357-385; Yager R. (1997) Fuzzy temporal methods for Video multimedia information systems, *Journal of Advanced Computational Intelligence*,Vol.1, N 1,37-44
10. Geverini, A. and Schubert L. (1995) Efficient algorithms for qualitative reasoning about time, *Artificial Intelligence* 74 ,207-248.
11. Huang C-M. and Wang C. (1998) Synchronization for interactive multimedia presentations, *IEEE MultiMedia*, October-98,44-51.
12. Freksa C. (1992). Using orientation information for qualitative spatial reasoning. *Proceedings of the International Conference GIS - from space to Territory. Theories and Methods of Spatio-Temporal Reasoning*, Pisa, Italy.
13. Reiter, R.and Mackworth A.K.(1990) A logical framework for description and image interpretation, *Artificial Intelligence* 41 125-155.
14. Donikian S. and Hegron G. (1991) The kernel of a declarative method for 3D scene sketch modeling, *Proceedins of Third International Conference on interactive systems*, Milan, Italy.
15. Tanaka T., Ohwi, J., Litvintseva L.V., Yamafuji K. and Ulyanov S.V. (1997) Soft Computing algorithms for intelligent control of a mobile robot for service use. Part1: Direct human-robot communications and managing system for cooperative control. *Soft Computing*, 1 (2) , 88-98.
16. Litvintseva L., Baidoun V., and Nalitov S. (1996) The system "Text-picture" for the direct human-robot communication. *Proceedings of Symposium on Robotics and Cybernetics*, CESA'96 IMACS Multiconference, Lille, France, 866-870.
17. Clay S.R. and Wilhelms J. (1996) Put: Language based interactive manipulation of objects. *IEEE Computer Graphics and Applications*, March, 31-39.
18. Liu J. (1996). Spatial reasoning about robot compliant movements and optimal paths in qualitatively modeled environment. *The International Journal of robotics Research*, 5 (2) , 181- 210.
19. Tanaka T., Ohwi J., Litvintseva L.V., Yamafuji K. and Ulyanov S.V. (1996) Intelligent control of a mobile robot for service use in office buildings and its soft computing algorithms. *Journal of Robotics and Mechatronics*, 8 (6) , 558-554.
20. McCarthy J. and Hayes P.J. (1969) Some philosophical problems from the standpoints of Artificial Intelligence, *Machine Intelligence* 4 (3), 124-156.
21. Genesereth M.R. and Nilsson N.J. (1992) *Logical Foundations of Artificial Intelligence*, Academic Press, N.Y.
22. Stein L.A. and Morgenstern L. (1994). Motivated action theory: a formal theory of causal reasoning, *Artificial Intelligence*, 71, 1-42.
23. Prendinger H. and Schurz G.(1996) Reasoning about Action and Change, a dynamic Logic Approach, *Journal of Logic, Language, and Information*, 5, N 2, 209-245.
24. Pratt V.R. (1980) Applications of modal logic to programming, *Studia Logica*, 39,257- 274.
25. Shoham Y. (1988). *Reasoning about change.* Massachusetts, MIT Press; Shoham, Y. (1989). Time for action: on relation between time, knowledge and action. *Planning, Scheduling, Reasoning about Actions*, Amsterdam, Holland.
26. Lifschitz V. (1991). Toward a metatheory of action. *Proceedings International Conference on principles of Knowledge Representation and Reasoning*, Cambridge, MA, 376-386.
27. Prasad Sistla A. and Clement Yu. (2000) Reasoning about Spatial Relationships, *Journal of Automatic reasoning*, 25, 291-328.
28. McDermott J. and Hendler D. (1995) Planning: what it is, what it could be, an introduction to special issue on planning and scheduling, *Artificial Intelligence*,76, 1-16.
29. Weisenbaum J. (1966) ELIZA, *Communication of the ACM*, 9, 36-45.
30. Winograd T. (1972) *Understanding Natural Language*, Academic press, New York.
31. Schank R.C. (1975) *Fundamental Studies in computer science, Conceptual information processing*, North Holland.
32. Woods W.A. (1970) Transition network grammars for natural language analysis, *Communication of the ACM*, 13, N 10, 591-606.
33. Adorni G., Poggi A. and Ferrari G., Attribute grammars as a robust technical basis for a human-computer interaction general purpose architecture, *Int. J. Human-Computer Studies*, 47, 531-563.
34. Kahaner D.(1994) Japanese Activities in Virtual reality, *IEEE Computer Graphics & Applications*, 1, 75-78.
35. Latta J.N. and Oberg D.J. (1994) A conceptual Virtual Reality Model, *IEEE Computer Graphics & Applications*, N 1, 23-29.
36. Litvintseva L.V. and S.D. Nalitov (1995) Virtual Reality: problem analysis and Solution Methods (in Russian) , *Artificial Intelligence News*, 3, Moscow, 24-89.
37. Gibson J.J (1966) *The Senses Considered as Perceptual Systems*, Houghton, Mifflin, Boston.
38. Sturman D.J. and Zeltzer D. (1994) A Survey of Glove-based Input, *IEEE Computer Graphics & Applications*, 1, 31-39.
39. Shimoga K.B., Murray A.M. and P.K. Khosla (1995) A Touch Reflection System for Interaction with Remote and Virtual Environments", *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Pittsburg, PA, August 5-9, 1-8.
40. Schmidt R.F. (1977) Somatovisceral sensibility, *Fundamentals of sensory phychology* R.F.Schmidt (Ed.), New York: Springer-Verlag.
41. Astheimer P. (1993) What you see is what you hear -Acoustics applied to Virtual Reality, *IEEE Symposium on Virtual Reality*, San Jose, USA.
42. Milgram P. and F. Kishino F. (1994) A Taxonomy of mixed reality virtual displays, *IEICE Transaction Information & Systems*, E-77-D, 12.
43. Sims D. (1995) See how they run: modeling evacuation in VR, *IEEE Computer Graphics&Applications*, March -95,11-13.
44. Cassel J. (2000) Embodied Conversational Interactive Agents, *Communications of the ACM*, Vol.43, N 4, 70-76.
45. Michie D. and R.Johnson R. (1984) *The Creative computer, Machine intelligence and Human knowledge*, UK,Viking.
46. Zaripov R.(1971) *Cybernetics and music*, 1971, Nauka, Moscow.
47. Pospelov D.A. (1982) *Toward to Artificial Intelligence: a fantasy or science*, Nauka, Moscow.
48. Nack F. (2000) All Contents Counts: The Future in Digital Media Computing is Meta, *Multimedia*, July-September, 10-13.
49. Artful Media (ed. D.D. Seligmann) (2000), The 21th Century Virtual reality Color Organ, *Multimedia*, July-September, 6-9.

Additional Reading:

50. Rich E. and Knight K. (1991) Artificial Intelligence, Second Edition, McGraw-Hill, N.Y.
51. Winston P.H. (1992) Artificial Intelligence, Third Edition, Addison-Wesley, Reading Mass.
52. Dubois D. & Prade H. (1989). Processing Fuzzy temporal knowledge. *IEEE Tranc.on Syst., Man and Cybernetics*, 19 (4), 729-743.
53. Emerson A.E. & Shinivasan V. (1989). *Branching time temporal logic*. Lecture Notes in Computer Science.
54. Haddawy P. (1996). A logic of time, change, and action for representing plans. *Artificial Intelligence*, 80 ,243-308.
55. Shanahan M. (1995). A circumscriptive calculus of events . *Artificial Intelligence*, 77 , 249-284.
56. Shanahan M. (1995). Default reasoning about spatial occupancy. *Artificial Intelligence* 74 , 147-163.
57. *Artificial Life*, An Overview (Ed. Ch.G.Langton) (1997), MIT Press.
58. Shneiderman B. (1998) *Designing the User Interface: strategies for Effective Human-computer Interaction*, 3rd edition, Addison-Wesley, Reading. Mass.
59. Arkin R. (1998) *Behabior-based Robotics*, AAAI Press.
60. Murphy R. (2000) *An Introduction to AI Robototics*, MIT Press.
61. *Music, Gestalt, and Computing* (1997) (ed. M. Leman), Springer-Verlag.

Embodied Conversational Interactive Agents (2000) (ed. Cassel J., Sulivan J., Prevost S, and Churchill E.) MIT Press.