Artificial Intelligence and Soft Computing

Behavioral and Cognitive Modeling of the Human Brain

Amit Konar

Department of Electronics and Tele-communication Engineering
Jadavpur University, Calcutta, India

CRC

CRC Press
Boca Raton London New York Washington, D.C.

11

Reasoning with Space and Time

The chapter presents the models for reasoning with space and time. It begins with spatial axioms and illustrates their applications in automated reasoning with first order logic. Much emphasis has been given on the formalization of the spatial relationships among the segmented objects in a scene. Fuzzy spatial relationship among 2-D objects has been briefly outlined. The application of spatial reasoning in navigational planning of mobile robots has also been highlighted. The second half of the chapter deals with temporal reasoning. The principles of temporal reasoning have been introduced from the first principles by situation calculus and first order temporal logic. The need for reasoning with both space and time concurrently in dynamic scene interpretation is also outlined at the end of the chapter.

11.1 Introduction

The reasoning problems we came across till now did not involve space and time. However, there exist many real world problems, where the importance of space and time cannot be ignored. For instance, consider the problem of navigational planning of a mobile robot in a given workspace. The robot has to plan its trajectory from a pre-defined starting point to a given goal point. If the robot knows its world map, it can easily plan its path so that it does not touch the obstacles in its world map. Now, assume that the robot has no prior knowledge about its world. In that case, it has to solely rely on the data it receives by its sonar and laser sensors or the images it grabs by a camera and processes these on-line. Representation of the space by some formalism and developing an efficient search algorithm for matching of the spatial data, thus, are prime considerations. Now, let us assume that the obstacles in the robot's world are dynamic. Under this circumstance, we require information about both space and time. For example, we must know the velocity and displacements of the obstacles at the last instance to determine the current speed and direction of the robot. Thus there is a need for both spatial and temporal representation of information. This is a relatively growing topic in AI and we have to wait a few more years to get a composite representation of both space and time.

Spatial reasoning problems can be handled by many of the known AI techniques. For instance, if we can represent the navigational planning problem of a robot by a set of spatial constraints, we can solve it by a logic program or the constraint satisfaction techniques presented in chapter 19. Alternatively, if we can represent the spatial reasoning problem by predicate logic, we may employ the resolution theorem to solve it. But how can we represent a spatial reasoning problem? One way of doing this is to define a set of spatial axioms by predicates and then describe a spatial reasoning problem as clauses of the spatial axioms. In this book we used this approach for reasoning with spatial constraints.

The FOL based representation of a spatial reasoning problem sometimes is ambiguous and, as a consequence, the ambiguity propagates through the reasoning process as well. For example, suppose an object X *is not very close* to object Y in a scene. Can we represent this in FOL? If we try to do so then for each specific distance between two objects, we require one predicate. But how simple is the representation in fuzzy logic! We need to define a membership function of '*Not-very-close*' versus distance, and can easily obtain the membership value of *Not-very-close* (X, Y) with known distance between X and Y. The membership values may later be used in fuzzy reasoning. A section on fuzzy reasoning is thus introduced for spatial reasoning problems.

Reasoning with time is equally useful like reasoning in space. How can one represent that an occurrence of an event A at time t, and another event B at time t+1, causes the event C to occur at time t+2? We shall extend the First order logic to two alternative forms to reason with this kind of problem. First one is called the **situation calculus**, after John McCarthy, the father of AI.

The other one is an extension by new temporal operators; we call it **propositional temporal logic**.

Section 11.2 describes the principles of spatial reasoning by using a set of spatial axioms. The spatial relationship among components of an object is covered in section 11.3. Fuzzy spatial representation of objects is presented in section 11.4. Temporal reasoning by situation calculus and by propositional temporal logic is covered in section 11.5 and 11.6 respectively. The formalisms of interval temporal logic is presented in section 11.7. The significance of the spatial and temporal reasoning together in a system is illustrated in section 11.8.

11.2 Spatial Reasoning

Spatial reasoning deals with the problems of reasoning with space. Currently, to the best of the author's knowledge, there exist no well-organized formalisms for such reasoning. So we consider a few elementary axioms based on which such reasoning can be carried out. These axioms for spatial reasoning we present here, however, are not complete and may be extended for specific applications.

Axioms of Spatial Reasoning

Axiom 1: Consider the problems of two non-elastic objects O_i , O_j . Let the objects be infinitesimally small having 2D co-ordinates (x_i, y_i) and (x_j, y_j) respectively. From commonsense reasoning, we can easily state that

$$\forall O_i , O_j , \ x_i \neq x_j \text{ and } y_i \neq y_j .$$

Formally,

$$\forall O_i , O_j \text{ Different } (O_i , O_j) \rightarrow \neg(Eq(x_i, x_j) \wedge Eq (y_i, y_j)).$$

An extension of the above principle is that no two non-elastic objects, whatever may be their size, cannot occupy a common space. If S_i and S_j are the spaces occupied by O_i and O_j respectively,

then $S_i \cap S_j = \phi$,

$\Rightarrow \neg (Si \cap Sj) = \text{true}$

$\Rightarrow \neg S_i \cup \neg S_j$ is true.

Formally,

$\forall O_i , O_j \quad S_i (O_i) \wedge S_j(O_j) \wedge \neg Eq(O_i,O_j) \rightarrow \neg S_i(O_i) \neg \vee S_j(O_j)$.

In the above representation, the AND ($\wedge$) and OR ($\vee$) operators stand for intersection and union of surfaces or their negations (complements).

Further, $\forall O_i , O_j$ means $O_i , O_j \in S$, where S is the entire space that contains O_i , and O_j, vide fig. 11.1.

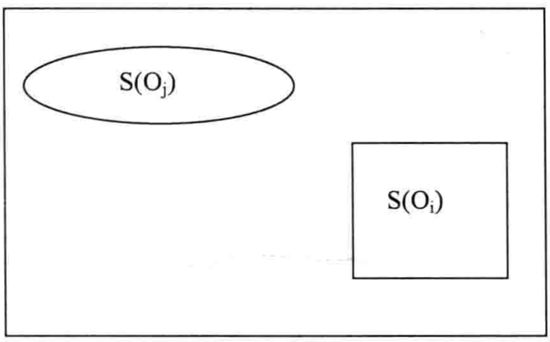


Fig. 11.1: Space S containing object O_i and O_j having 2-D surfaces $S(O_i)$ and $S(O_j)$.

In our formulation, we considered two dimensional spaces S , $S(O_i)$ and $S(O_j)$. However, we can easily extend the principle to three dimensions.

Axiom 2: When an object O_i enters the space S , $S \cap S(O_i) \neq \phi$, which implies

$S \wedge S(O_i)$ is true .

Formally ,

$\forall O_i \; S(O_i) \wedge Enter(O_i, S) \rightarrow S \wedge S(O_i)$.

Similarly, when an object O_i leaves a space S, $S \cap S(O_i) = \phi$,

Or, $\neg S \vee \neg S(O_i)$ is true.

Formally, $\forall O_i \; S(O_i) \wedge Leaves (O_i , S) \rightarrow \neg S \vee \neg S(O_i)$.

Axiom 3: When the intersection of the surface boundary of two objects is a non-null set, it means either one is partially or fully embedded within the the other, or they touch each other. Further, when a two dimensional surface touches another, the common points must form a 2-D line or a point. Similarly, when a 3-dimensional surface touches another, the common points must be a 3-D /2-D surface or a 3-D / 2-D line or a point. It is thus evident that two objects touch each other, when their intersection of surface forms a surface of at most their dimension. Formally,

$\forall O_i, \forall O_j$ Less-than-or-Equal-to (dim($S(O_i) \wedge S(O_j)$), dim ($S(O_I)$)) $\wedge$
Less-than-or-Equal-to (dim($S(O_i) \wedge S(O_j)$), dim ($S(O_j)$)) $\rightarrow$ Touch (O_i, O_j)

Where 'dim' is a function that returns the dimension of the surface of its argument and dim ($S(O_i) \wedge S(O_j)$) represents the dimension of the two intersecting surfaces: O_i and O_j. The $\wedge$-operator between the predicates Less-than-or-Equal-to denotes logical AND operation.

Axiom 4: Now, for two scenes if d_{ij1} and d_{ij2} denote the shortest distance between the objects O_i and O_j in scene 1 and 2 respectively, then if $d_{ij2}<d_{ij1}$, we can say the objects O_i and O_j are closer in scene 2 compared to that in scene 1. Formally,

$\forall O_i, O_j$ Exists (O_i , O_j , in-scene1) $\wedge$ Shortest -distance ($d_{ij\,1}$, Oi , Oj , in-scene1) $\wedge$ Exists (O_i , O_j, in -scene2) $\wedge$ Shortest -distance (d_{ij} 2 , O_i , O_j , in-scene2) $\wedge$ smaller ($d_{ij\,2}$, $d_{ij\,1}$) $\rightarrow$ Closer (O_i , O_j , in-scene2 , wrt-scene =1);

where the predicate Exists (O_i,O_j ,in-scene k) means O_i and O_j exists in scene k; Shortest distance (d_{ijk} , O_i , O_j , in-scene k , wrt-scene =1) denotes that d_{ijk} is the shortest distance between O_i , and O_j in scene k with respect to scene 1.

The axioms of spatial reasoning presented above can be employed in many applications. One typical application is the path planning of a mobile robot. Consider, for example, the space S, where a triangular shaped mobile robot has to move from a given starting to goal point, without touching the obstacle O1, O2, O3, O4, . , O7.

We can construct a **constraint logic program** (CLP) to solve this problem. We assume that the robot R can sense the obstacles from a distance by ultrasonic sensors, located around the boundary of it. The CLP of this problem is presented below.

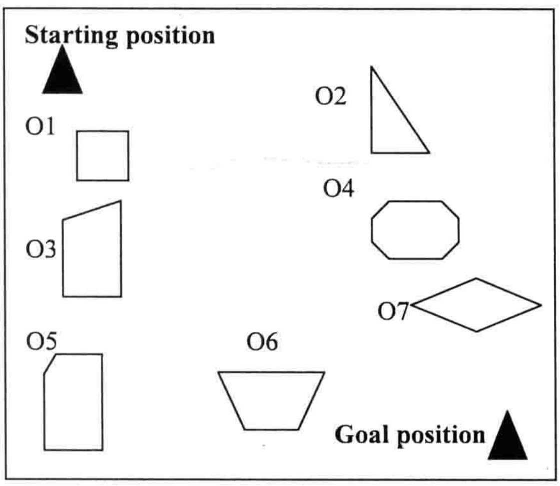
Move(R, Starting -position , goal -position) : -
Move S(R) in S,
not Touch(S(R), S (O_i)) $\forall$ O_i.

Move(R, goal-position, goal-position).

The above program allows the robot R to wander around its environment, until it reaches the goal-position. The program ensures that during the robot's journey it does not hit an obstacle. Now, suppose, we want to include that the robot should move through a shortest path. To realize this in the CLP we define the following nomenclature.

1. Next-position(R): It is a function that gives the next-position of a robot R.

2. S (next-position (R)): It is a function, representing the space to be occupied by the robot at the next-position of R.



Space S

Fig 11.2: Path planning of a robot R in space S.

It is to be noted that the robot should select arbitrary next position from its current position and then would test whether the next-position touches any object. If yes, it drops that next-position and selects an alternative one until a next-position is found, where the robot does not touch any obstacle. If more than one next-position is found, it would select that position, such that the sum of the distances between the current and the next-position, and between the next position and the goal, is minimum.

The CLP for the above problem will be presented next. A pseudo Pascal algorithm is presented below for simplicity.

```
Procedure Move-optimal (R , Starting-position, goal-position)
Begin
 Current-position (R ) := Starting-position (R );
 While goal not reached do
  Begin
    Repeat
     Find-next-position ( R) ;
     j:=1;
     If S (next-position( R)) does not touch S(O_i) ∀i ;
     Then do
      Begin
       Save next-position (R ) in A[j] ;
        j=j +1;
      End ;
    Until all possible next positions are explored;
    ∀j Find the next-position that has the minimum distance from the
      current position of R and the goal; Call it A[k].
  current-position(R) := A [k] ;
 End while
End
```

We now present the CLP that takes care of the two optimizing constraints: i) movement of the robot without touching the obstacles, and ii) traversal of an optimal (near- optimal) path.

Move-optimal (R, Starting-position, goal-position):-
  Move S(R) in S,
  Not Touch(S(R), S (O_i)) $\forall$ i,
  Distance (next-position (R) , current-position (R)) +
  Distance (next-position (R) , goal-position)
  is minimum $\forall$ feasible next-position(R),
  current-position (R) $\leftarrow$ next-position (R),
  Move-optimal (R, current-position, goal-position).

Move-optimal (R, goal-position, goal-position).

It is to be noted that here we need not explicitly define Touch (S (R), S (O_i))

as it is presumed to be available in the system as a standard predicate, following axiom 3. Further, we can re-define the distance constraint in the last program by axiom 4 as follows:

Closer (next-position (R), current-position (R), in-scene k, w.r.t scene $\neq$ k), Closer (next-position (R), goal-position, in-scene k, w.r.t scene $\neq$ k).

The significance of the spatial axioms now is clear. It helps in declaring the problem specifications in simpler terms, rather than formulating the problem from the grass-root level.

11.3 Spatial Relationships among Components of an Object

Many physical and geometric objects can be recognized from the spatial relationship among its components. For instance, let us define a chair as an object consisting of two planes abcf and cdef having an angle θ between them, where $\theta \leq 90^0+\alpha$ and where $0 \leq \alpha \leq 45^0$. Further, one of its plane is perpendicular to at least 3 legs (the 4^{th} one being hidden in the image). So, we define:

Object(chair):-
Angle-between (plane1, plane2, 90+α) ,
Greater-than(α, 0),
Less-than (α, 45) ,
Parallel (line1, line2, line3) ,
perpendicular (line1, plane1),!

For actual realization of the small program presented above, one has to define equation of lines and planes; then one has to check the criteria listed in the logic program. It may be noted here that finding equation of a line in an image is not simple. One approach to handle this problem is to employ a stochastic filter, such as Kalman filtering [1] . We shall discuss this issue once again in chapter 17 on visual perception. However, for the convenience of interested readers, we say a few words on the practical issues.

A skeleton of a chair, which can be obtained after many elementary steps of image processings is presented in fig. 11.3. Now, the equation of the line segments is evaluated approximately from the set of 2-dimensional image points lying on the lines. This is done by employing a Kalman filter. It may be noted that the more the number of points presented to the filter, the better would be accuracy of the equation of the 2-dimensional lines. These 2-D lines are then transformed to 3-D lines by another stage of Kalman filtering. Now, given the equation of the 3-D lines, one can easily evaluate the equation of the planes framed by the lines by using analytical geometry. Lastly, the constraints like the angles between the planes, etc. are checked by a logic program, as described above. The graduate students of the ETCE department at Jadavpur University verified this method of recognizing a 3-D planer object from its skeletal model.

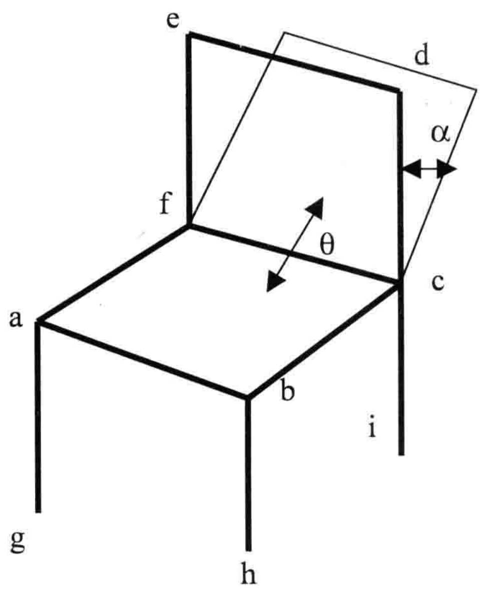


Fig. 11.3: Spatial relations among components of a skeleton chair.

11.4 Fuzzy Spatial Relationships among Objects

Consider the objects A and B in fig. 11.4 (a) and (b). We would say that B is left to A. It , however, is to be noted that B and A have some overlap in (a) but there is no overlap in (b).

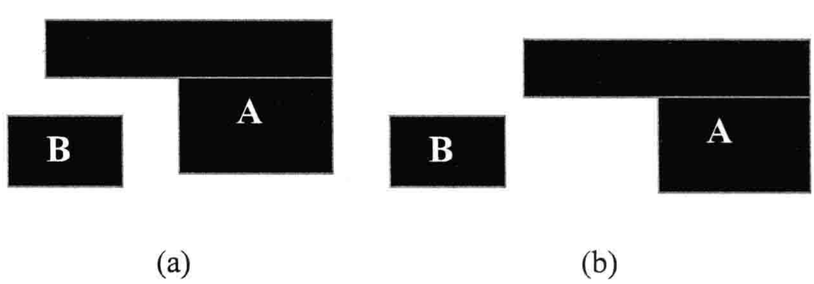


Fig. 11.4: Object B is left to object A: (a) with overlap, (b) without overlap

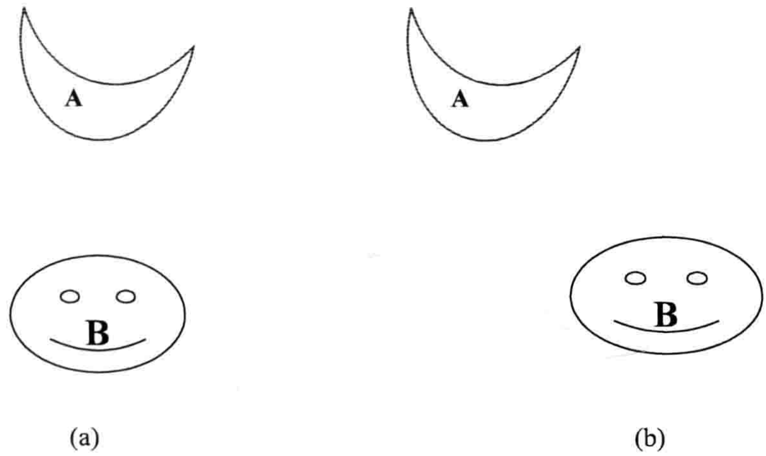


Fig. 11.5: Object B is down to object A: (a) exactly down, (b) down but right shifted.

Now consider fig. 11.5 where in both (a) & (b) B is down to A; but in (a) B is exactly down to A, whereas in (b) it is right shifted a little. To define these formally, we are required to represent the spatial relationships between the objects A and B by fuzzy logic.

Let us first define spatial relations between points A and B. We consider four types of relations: right, left, above and below. Here following Miyajima and Ralescu [4], we define the membership function as a square of sine or cosine angles θ (vide fig. 11.6), where θ denotes the angle between the positive X axis passing through point A and the line joining A and B. The membership functions for the primitive spatial relations are now given as follows:

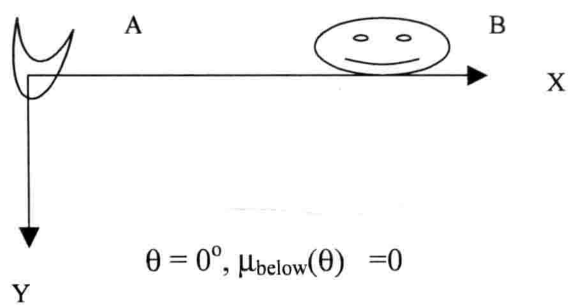
$$\mu_{right}(\theta) = \cos^2(\theta) \text{, when } -\Pi/2 \le \theta \le \Pi/2 \text{,}$$
$$= 0 \text{, otherwise.}$$

$$\mu_{left}(\theta) = \cos^2(\theta) \text{, when } -\Pi < \theta < -\Pi/2 \text{,}$$
$$\text{and } \Pi/2 \le \theta \le \Pi$$
$$= 0 \text{, otherwise.}$$

$$\mu_{below}(\theta) = \sin^2\theta \text{, when } 0 \le \theta \le \Pi \text{,}$$
$$= 0 \text{, otherwise.}$$

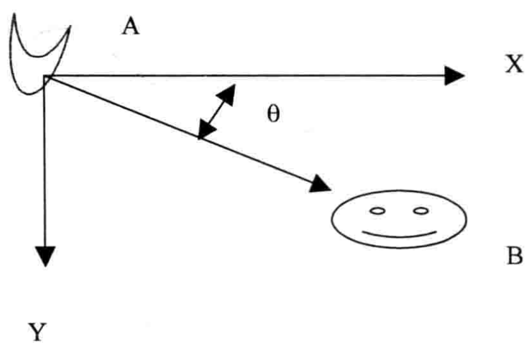
$$\mu_{above}(\theta) = \sin^2\theta \quad , \text{ when } -\Pi \le \theta \le 0 ,$$
$$= 0, \quad \text{otherwise.}$$

A common question that now arises is why we select such functions. As an example, we consider the 'below membership function'. Let us compute $\mu_{below}(\theta)$ at a regular interval of $\theta = \Pi/4$, in the graph $0 \le \theta \le \Pi$. Fig. 11.5 presents the membership values for different θ. It is clear from the figure that when B is exactly below A (fig. 11.6(c)) $\mu_{below}(\theta = \Pi/2)=1$, which is logically appealing. Again when $\theta=\Pi/4$ or $\theta=3\Pi/4$ (fig. 11.6 (b) & (d)), the membership value of $\mu_{below}(\theta)=1/2$; that too is logically meaningful. When $\theta = 0$ (fig. 11.6(a)) or Π, $\mu_{below}(\theta) =0$, signifying that B is not below A. The explanation of other membership functions like $\mu_{right}(\theta)$, $\mu_{left}(\theta)$, $\mu_{above}(\theta)$ can be given analogously.



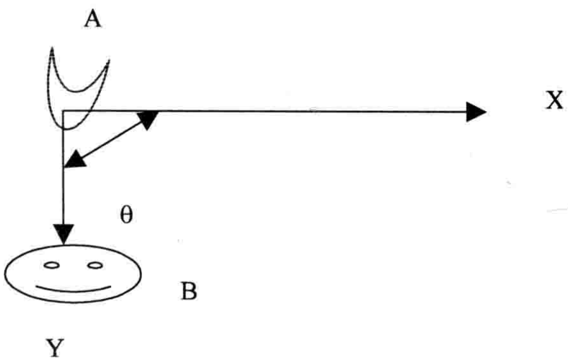
$\theta = 0^o$, $\mu_{below}(\theta) \;\; =0$

(a)



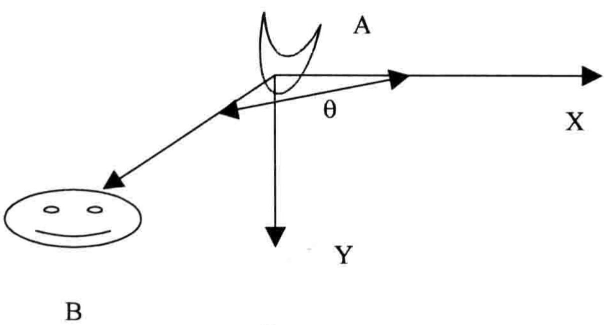
$\theta = {}^{\Pi}/4$, $\mu_{below}(\theta) \;\; =1/2$

(b)



$\theta = \pi/2$, $\mu_{below}(\theta) = 1$

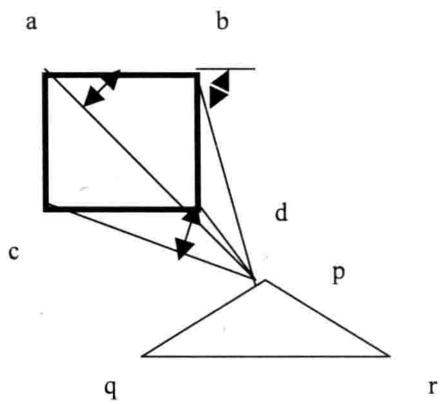
(c)



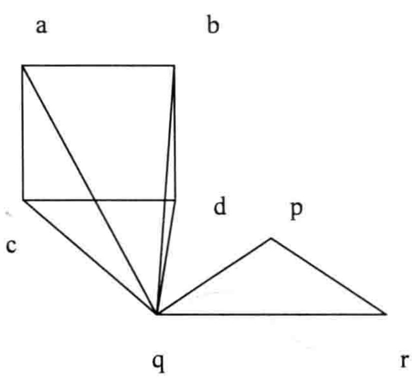
$\theta = 3\pi/4$, $\mu_{below}(\theta) = 1/2$

(d)

Fig. 11.6: Illustrating significance of the $\mu_{below}(\theta)$ function.



(a)



(b)

Here in (a) and (b) computation of angles (w.r.t the horizontal axis) of the lines joining the vertices of the rectangle to the vertices p and q of the triangle have been illustrated. Similar computations have to be performed for the line joining the vertices of the rectangle to the vertex r of the triangle. All 8 angles have not been shown in the figure for clarity.

Fig. 11.7: Demonstrating f (θ) / (m .n) computation.

So far we discussed spatial relationship between two points by fuzzy measure. Now, we shall discuss the spatial relationships between two objects. Let A and B be two objects and $\{a_i , 1 \le i \le n \}$, $\{ b_j , 1 \le j \le m \}$ be the set of points on the boundary A and B respectively. We first compute the angle θ_{ij} between each two points a_I and b_j . Since there are n a_I points and m b_j points, the total occurrence of θ_{ij} will be (m X n). Now, for each type spatial relation like b_j below a_{Ii}, we estimate $\mu_{below}(\theta_{ij})$. Since θ_{ij} has a large range of value [0, Π], we may find equal value of $\mu_{below}(\theta_{ij})$ for different values of θ_{ij}. A frequency count of $\mu_{below}(\theta_{ij})$ versus θ_{ij} is thus feasible. We give a generic name f (θ) to the frequency count. Since f (θ) can have the theoretical largest value (n. m), we divide f (θ) by (m. n) to normalize it. We call that normalized frequency $\underline{f}(\theta)$ = f (θ) /(m .n). We now plot $\underline{f}(\theta)$ versus θ and find where it has the largest value. Now to find the spatial relationship between A and B, put the values of θ in $\mu_{below}(\theta)$ where $\underline{f}(\theta)$ is the highest.

In fig. 11.7 we illustrate the method of measurement of the possible θ_{ij}s. Since abcd is a rectangle and pqr is a triangle, considering only the vertices, m .n =3. 4 =12. We thus have 12 possible values of θ_{ij}. So $\underline{f}(\theta)$ = f(θ)/12. It is appearing clear that $\underline{f}(\theta)$ will have the largest value at around 45 degrees (fig. 11.8); consequently $\mu_{below}(\theta=45^\circ)$ gives the membership of pqr being below abcd.

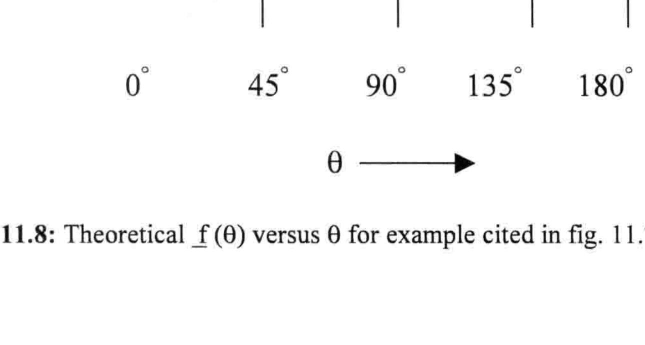


Fig. 11.8: Theoretical $\underline{f}(\theta)$ versus θ for example cited in fig. 11.7.

11.5 Temporal Reasoning by Situation Calculus

'Temporal reasoning', as evident from its name, stands for reasoning with time. The problems THE real world contain many activities that occur at a definite sequence of time. Further there are situations, when depending upon the result of occurrence of one element at a time t, a second event occurs at some time greater than t. One simple way to model this is to employ 'situation calculus' devised by John McCarthy [3].

The reasoning methodology of situation calculus is similar with first order predicate logic. To understand the power of reasoning of situation calculus, we are required to learn the following terminologies.

Definition 11.1: An **event** stands for an activity that occurs at some time.

Definition 11.2: A **fluent** is a fact that is valid at some given time frame but becomes invalid at other time frames.

Definition 11.3: A **situation** is an interval of time during which there is no change in events.

The following example illustrates the above definitions.

Example 11.1: Consider the following facts:

1.It was raining one hour ago.

2.People were moving with umbrellas in the street.

3.The rain has now ceased.

4.It is now noon.

5.The sky is now clear.

6.The sun is now shining brightly.

7.Nobody now keeps an umbrella open.

Statement 1, 2, 3, 6 and 7 in this example stand for events, while all statements 1-7 are fluent. Further, it is to be noted that we have two situations here; one when it was raining and the other when the rain ceased.

11.5.1 Knowledge Representation and Reasoning in Situation Calculus

To represent the statements 1-7 in situation calculus, we use a new predicate 'Holds'. The predicate Holds(s, f) denotes that the fluent f is true in situation s. Thus statement (1-7) can respectively be represented as:

Holds (0, it-was-raining) (1)

Holds (0, people-moving-with-umbrellas) (2)

Holds (now, rain-ceased) (3)

Holds (now, it-is-noon) (4)

Holds (now, the-sky-is-clear) (5)

Holds (now, the-sun-shining-brightly) (6)

Holds (Results (now, the-sun-shining-brightly),

not (anybody-keeps-umbrella-open)) (7)

The representation of the statements (1-6) in situation calculus directly follows from the definition of predicate 'Holds'. The representation of statement (7), however, requires some clarification. It means that the result (effect) of the sun shining brightly is the non-utilization of the umbrella. Further, 'not' here is not a predicate but is treated as a term (function). In other words, we cannot write not(anybody-keeps-umbrella-open) as ¬(anybody-keeps-umbrella-open).

For reasoning with the above facts, we add the following rules:

If it rains, people move with umbrellas. (8)

If the rains ceased and it is now noon then the result of sun shining brightly activates nobody to keep umbrella open. (9)

The above two rules in situation calculus are given by

∀s Holds(s, it-was-raining)→ Holds(s, people-moving-with-umbrellas) (8)

∀s Holds(s, rain-ceased) ∧ Holds(s, it-is-noon) → Holds (result (s, sun-shining-brightly), not (anybody-keeps-umbrella-open)) (9)

Reasoning: Let us now try to prove statement (7) from the rest of the facts and knowledge in the statements (1-9). We here call the facts axioms. So, we have: