A Complete Classification of Occlusion Observer's Point of View for 3D Qualitative Spatial Reasoning

Chaman Lal Sabharwal
Missouri University of Science and Technology
Rolla, MO- 63128, USA
chaman@mst.edu

Abstract

Qualitative Spatial Reasoning (QSR) theories have applications in areas such as geographic information systems (GIS), robotics, biomedicine and spatial databases. Several region connection calculi have been proposed for use in this capacity. Primarily the existing QSR theories have been applied to 2D data. Yet the ability to perform qualitative reasoning over a collection of 3D spatial objects is desirable. Over the past two decades several theories have appeared for accurately representing and acquiring 3D spatial knowledge: LOS-14, its extension ROC-20, occlusion calculus OCC, fourteen occlusion states OCS-14, and a recent visual VRCC-3D+ with 17 occlusion predicates. Each has a positive impact. However there are still some issues and ambiguities that require unambiguous ontology and resolution. In this paper, we provide a new set of self-documenting predicates for 3D complete spatial relations including occlusion. In addition we provide new heuristics for eliminating the time consuming computations by employing efficient data structures. This improvement will greatly enhance the usefulness and usability of aforementioned systems.

Keywords: geographical information systems, robotic navigation, spatial objects, graphics, occlusion

1. Introduction

Most of the theories about space and time study the quantitative aspects of a problem, whereas the qualitative calculi allow for rather inexpensive reasoning about entities located in space. For example, some of spatial reasoning is implemented for handling geographical information systems (GIS) queries efficiently [1], and such reasoning is used for robotic sensors, biomedicine, spatial networking, and cognitive sciences.

Although reasoning over two dimensions is sufficient for many applications [2], other spatial reasoning applications need to consider information in more than two dimensions. Thus the ability to perform qualitative spatial reasoning over a collection of 3D objects is necessary. The 3D spatial reasoning involves the visualizing and then manipulating spatial relations. A 2D QSR system cannot be utilized for such tasks. Robots see and interpret the world with data acquired through sensors.

Further, in order to determine occlusion, the view reference point, the plane of projection, and the type of projection must be known. Over the past two decades several theories have appeared for accurately representing and acquiring 3D spatial knowledge: Galton developed Line of Sight method with 14 occlusion relations LOS-14, in 1994 [3], its extension Region Occlusion Calculus with 20 occlusion relations ROC-20 was designed by Randell et al. in 2001 [4], Kohler developed the occlusion calculus OCC in 2002 [5], RCCD-3D by Albath et al. in 2010 [6], Guha et el. Designed OCS-14 (*Occlusion States 14*) in 2011 [7] and at the same time independently Sabharwal et al. developed VRCC-3D+ in 2011 [8]. Each has a positive impact. However there were some issues that required resolution. Recently, Eloe et al. attempted to resolve such issues and improve upon the computational aspects in 2014 [9]. In their attempt, they introduced two predicates for depth as discussed in Section 4, which turned out to be obscure and inefficient. In this paper, we provide a completely new set of self-documenting predicates for occlusion relations superseding the aforementioned work. In addition we provide efficient heuristics for eliminating the time consuming ray tracing used in performing obscuration computations. These improvements will greatly enhance the usefulness and usability of the aforementioned systems. The same reasoning works for mobile objects when the observer is stationary.

This paper is organized as: Section 2 describes mathematical concepts and region connection calculus background Section 3 clarifies the graphics concept of closer, occlude, obscure, infront, Section 4 describes recent implementation issues in handling occlusions problems, Section 5 describes complete, consistent and new set of occlusion relations using first order logic, Section 6 discusses efficient implementation techniques, Section 7 is on conclusions and future work.

2. Background

A. Mathematical Concepts

Basic mathematical concepts are the same as in the point set topology. The definition of connectedness in region connection calculus is slightly different. For any non-empty bounded set A, we use symbols A^c, A^i, A^b, and A^e to represent the universal complement, interior, boundary, and exterior of a set A, respectively. In mathematics, a set is *connected* if it cannot be the union of disjoint open sets. For example, the set $(0,1)\cup(1,2)$ is disconnected as (0,1) and (1,2) are open sets. In RCC, regions A and B are *weakly* connected if $\bar{A}\cap \bar{B}\neq\emptyset$. Thus $(0,1)\cup(1,2)$ is connected in RCC because

DOI reference number: 10.18293/DMS2015-042

$[0,1] \cap [1,2] \neq \emptyset$. This is equivalent to (1) C(A,A), and (2) C(A,B) $\leftrightarrow$ C(B,A) for any tow regions A and B.

B. Region Connection Calculi

Much of the foundational research on qualitative spatial reasoning is based on a region connection calculus (RCC) that describes 2D regions (i.e., topological space) by their possible relations to each other [10, 11]. Conceptually, for any two distinct objects, there are three possibilities on broad level: (a) *one is outside the other*, that results in the discrete spatial relation, DR(DiscRete), (b) *one overlaps the other across boundaries*. This means PO(proper overlap), (c) *one is inside the other*, that means EQ(equal) or PP(proper part). To make relations jointly exhaustive and pairwise distinct (JEPD), we have converse relation denoted PPc(proper part converse), PPc(A,B)≡PP(B,A). These five relations constitute RCC5 relations. For additional detail on discrete, specifically, DR is split into DC(disconnected) and EC(externally connected). For proper part, PP is split into TPP (tangential proper part) and NTPP (non-tangential Proper part). Similarly for proper part converse, PPc, we have converse relations TPPc and NTPPc. These eight relations constitute RCC8 relations of region connection calculus.

RCC8 was formalized by using first order logic [10] or 9-Intersection model [11], see Fig. 1. The intersections of interior (Int), boundary (Bnd), and exterior (Ext) of one object with the other object form 9-Intersections: IntInt, IntBnd, IntExt, BndInt, BndBnd, BndExt, ExtInt, ExtBnd, ExtExt.

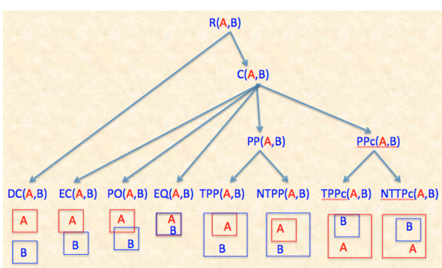


Fig. 1. RCC8 Relations in 2D.

Whereas a 2D object is in a plane, a 3D object is in space. The simplest examples of 3D objects are a pyramid, a cuboid, a cylinder, and a sphere. A concave pyramid is a complex, simply connected 3D object. Since concave objects can be partitioned into convex objects, for the rest of this discussion, we will base our analysis on convex objects.

RCC8 spatial relation for 3D objects is incomplete without occlusion consideration. VRCC-3D+ is a region connection calculus that qualitatively determines the spatial relations between 3D objects, both in terms of connectivity and obscuration [6, 8, 9]. The VRCC-3D+ connectivity relations are named the same as in RCC8; however, the VRCC-3D+ connectivity relations are calculated in 3D rather than 2D. The relative depth is denoted by tri-valent parameter InFront. Fifteen obscuration relations were defined in VRCC-3D+ [8,9]. Considered from a 2D projection, each VRCC-3D+ obscuration relation is a refinement of basic concepts of no obscuration, partial obscuration, and complete obscuration. A hybrid occlusion relation specifies both a connectivity relation and an obscuration relation.

3. **Occlusion Concepts**

Basically occlusion of one object by another object depends on the observer location relative to the objects. Our approach is to derive spatial obscuration relations and classification from projection of 3D objects on a 2D projection plane and relative distance of the objects from the observer. The 2D plane is used to determine the existence of occlusion. The depth parameter, *InFront*, coupled with projections determine the type of occlusion. Obscuration predicates are based on two parameters: projection in a plane and depth (distance of the object from viewpoint).

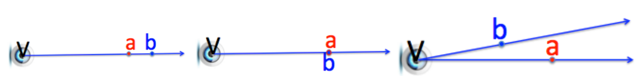
In general, there are two types of projections: Parallel and Perspective. Parallel projection loses the depth concept in the projection, because the all projectors are parallel. We use perspective view for accurate depth representation. The terms "inFront", "occlusion", "obscuration", "closer" are closely related. In natural language, the term inFront between two objects A and B is synonymously interpreted as "A is in front of B" ,"A occludes B", "A obscures B", "A is closer than B". However, there are subtle differences between these, see examples below. Here we describe these terms precisely with the help of examples.

A. Examples of occlusion detection supporting qualitative spatial reasoning

In computer graphics, there is a distinction between terms occlusion and obscuration. Occlusion means opaque (hidden, obstructed), whereas obscuration means unclear (hazy, vaguely transparent). For this reason, the graphics community prefers the term occlusion to obscuration. We are using both terms interchangeably to mean opaque.

A1. Point Occlusion

If V (viewer), a, b are collinear points in that order and the distance of a is smaller than the distance of b from viewer V, then the three terms are equivalent: point a is closer than point b or a occludes b or a is in front of b, see Fig. 2(a). However if both points a and b are equidistant from V, then none is in front of the other, see Fig. 2(b). Two points are equidistant, coincident, none occludes the other. This is *not* mutual occlusion as explained in Section 3.A2 . If V, a, b are not collinear, then b can be closer than a without b occluding a, see Fig. 2(c). Thus a point can be closer without being in front of the other.



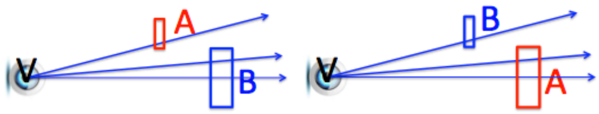
(a) (b) (c)

Fig. 2 (a) View point V, points a and b; a occludes b, a is in front of b, a is closer than b to the viewpoint. (b) View point

V, points a and b; a and b are equidistant, none is in front of the other. (c) View point, points a and b, b is closer than a, but b does not occlude a, b is not in front of a.

A2. Object Occlusion

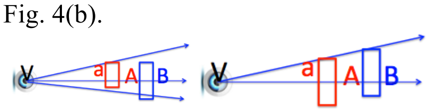
In 2D, to determine whether an area A occludes an area B, there are several configurations of A and B. If a ray from V does not intersect any of the two regions, then it does not shed light on the occlusion. If for *every* ray from V, it intersects exclusively one object and not the other object, then no object occludes the other, see Fig. 3(a, b). An object A can be closer than object B without obscuring the object B, see Fig. 3(a).



(a) (b)

Fig. 3 Objects do not intersect (a) A is closer than B (b) B is closer than A.

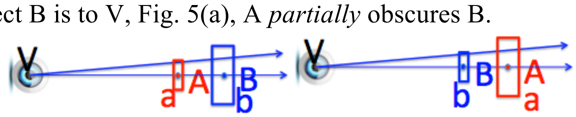
We cannot make any conclusion from a ray if the ray intersects only one object exclusively and not the other object, see Fig. 4(b).



(a) (b)

Fig. 4. (a) ray from V intersects A, but not B, another ray from V intersects B , but not A, still A occludes B (*partially*). (b) ray from V intersects both, the front point occludes the other point, still A occludes B (*completely*).

If for *every* ray from V that intersects the interiors of both objects A and B at points a and b, with a in front of b, then the object A occludes object B, or A is in front of B are synonymous, Fig. 5(a, b). Also object A is closer to V than object B is to V, Fig. 5(a), A *partially* obscures B.



(a) (b)

Fig. 5. (a) A is infront of B, A occludes B, A is closer than B is, (b) B is infront of A; B occludes A; B is closer than A is.

If there are two rays that intersect objects A and B, such that for one ray, the intersection with A is closer than intersection with B, and for the second ray, the intersection with B is closer than intersection with A, then the object A partly occludes B and conversely. They *mutually* occlude each other, see Fig. 6. None alone occludes the other completely or partially.

We require that for *every* ray from V that intersects both the objects, the two intersections are used to make judgment about the relative depth of the objects. If the ray intersection occurs at multiple points, only the closest intersections are

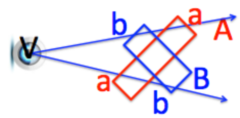
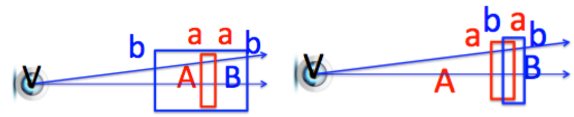


Fig. 6. Mutual Occlusion.

significant for occlusion detection. For example, if the ray intersection is Vbaab, we need to process only first two intersections, in this case, Vba, so the point b occludes point a see Fig. 7 ; If the ray intersection is Vabab, we process the first two intersections, in this case, Vab. So the point a occludes the point b.



(a) (b)

Fig. 7. (a) Ray intersection with A and B shown at two points each so that ray is Vbaab. However only Vba is sufficient for occlusion determination between A and B. (b) Ray intersection with A and B shown at two points each so that ray is Vabab. However only Vab is sufficient for occlusion determination between A and B.

Partial or full occlusion: If for *every ray* from view point V that *intersects both* A and B, and ray points a∈A are closer than corresponding points b∈B, then A occludes (*partially* or *completely*) B, see Fig 5(a) and Fig. 7(b).

Mutual or Equal. Similarly it can be seen that if on *every* ray from view point V that intersects both A and B, and if sometimes a is closer and sometimes b is closer, then A and B *mutually* occlude each other else if points a, b are equidistant from V, then it is termed as *equal* occlusion for spatial reasoning, this is a clear distinction between *mutual* and *equal* occlusion predicates. This is a conscious decision made for spatial reasoning because it provides expressive power at no cost.

4. Problems

In Section 1, we mentioned that there are issues in the implementation of occlusion in VRCC-3D+ [9]. Let us first describe the problems and then we will present new formulation and new crisp occlusion relations in Section 5.

First Problem. Let A, B be 3D objects, (x, y) be a projection point in the projection plane, a line of sight from camera C (Center of Perspective Projection) through (x, y) intersects objects at points $f_A(x, y)$ and $f_B(x, y)$. Eloe et al. [9] define occlusion by means of two predicates o and o_c. The definition of predicate o is the following equation [9]

$$o(A,B) = \begin{cases} T: \exists x,y \ni |C - f_A(x,y)| < |C - f_B(x,y)| \\ F: otherwise \end{cases}$$

and converse $o_c(A, B) = o(B, A)$. This definition of occlusion is trivially inaccurate. By comparing points on only one ray, one cannot claim any thing about the whole object. For example, the occlusion detection is inconclusive by this definition as in Fig. 6, there are points on A and B, in one case point a is in front of point b and in the other case b is in front of a.

Second Problem Each occlusion relation is determined with a 5 parameters (IntInt, IntBnd, BndInt, o, o_c) instead of 4 parameters (IntInt, IntBnd, BndInt, InFront). This increases computational time complexity by 20%. Increased computation slows down the interaction time and is detrimental to usability of the system. Instead of two new predicates to replace a single predicate InFront, the tri-valent predicate InFront can be made quad-valent to accommodate *mutual* for accurate interpretation of InFront.

Third Problem Eloe et al. [9] point out that with their approach, 15 occlusion relations have been reduced to 12 relations. Their table 1 below is listing of 12 relations. A closer look at the table indicates that some of their relations are disjunctions. In a table each row represents a unique relation rule, the number of rows in the table is 12+1+1+2+1=17. This contradicts the assertion that it is a smaller set of occlusions than 15 unique relations. For efficiency consideration, larger set of predicates amounts to larger computation time. Baring, the disjunctions, we will show that same tasks can be accomplished with 7 crisp relations instead of 12.

Fourth Problem There is no consistent ontology for naming the occlusion relations, see table 1. We devise a complete, systematic comprehensible list of occlusion relations see table 2. Also there are repeated occlusion ray tracing computations that can be eliminated. As it is, the occlusion computation is quite inefficient in [9].

Table 1. Full obscuration relation set with identified converse relations. cited[9]

	IntInt	IntExt	ExtInt	o	o_c	Converse
nObs_e	F	T	T	F	F	nObs_e
pObs	T	T F	T	T	F	pObs_c
pObs c	T	T	T F	F	T	pObs
pObs_e	T	T	T	F	F	pObs_e
pObs m	T	T F T	F T T	T	T	pObs m
eObs	T	F	F	T	F	eObs_c
eObs c	T	F	F	F	T	eObs c
eObs e	T	F	F	F	F	**eObs e**
eObs m	T	F	F	T	T	eObs m
cObs	T	T	F	T	F	cObs_c
cObs c	T	F	T	F	T	cObs
cObs_e	T	T F	F T	F	F	cObs_e

We will present improved and enhanced set of crisp self-documenting predicates. Consequently these issues will disappear de facto in Section 5.

5. Completeness of Spatial Object Occlusions

In this section, we describe occlusion predicates gradually as follows: (1) redefine InFront accurately, (2) describe occlusion relation in natural language, (3) define occlusion predicates using first order logic, and (4) describe predicates via a table whose each row is rule for occlusion determination and classification.

In general, occlusion analysis is performed in two steps. First, the RCC5 relation is computed between the projections in 2D. Second, the qualitative spatial distance between the viewer and 3D objects is determined. The projection alone is not sufficient to determine obscuration. In Sabharwal et al. [8], the predicate InFront is used with value Y if A is closer than B; N is used if B is closer than A, and E is used imply that A and B are equidistant. As shown in the examples in Section 2, though three relations are accurate for single points, but they are not exhaustive for objects. There is a possibility that the objects cross, see Fig. 6. This leads to inaccurate classification due to tri-valent interpretation of the *InFront* predicate.

However, it is noted that the predicate InFront is sound in principle, but incomplete and inaccurate in implementation. We propose that predicate InFront accommodate *mutual* occlusion explicitly, when objects cross each other. In our further discussion, we will have InFront represent four possibilities; "A is in front of B", "B is in front of A", "A and B are equidistant from V", and "A and B *mutually* obscure each other to indicate that A obscures B *partly* and B obscures A *partly* exclusively". Using four values of InFront, we will update occlusion relations accordingly. The characterization of crisp occlusion relations is detailed in table 2.

5.1 Description of Depth Parameter InFront

Let V be the viewer, let A_P and B_P be the projections of A and B on the projection plane. The observer captures the scene that is in field of view, FOV. For occlusion purposes, the *viewer sees* objects through the *window* $A_P \cup B_P$ only in the projection plane. The ray from V, to intersect both objects, is through points in $A_P \cap B_P$. Any reference to objects, A and B for *InFront* predicate, is reference to the part of objects seen via *only* $A_P \cap B_P$. Let P(x,y) be a point in $A_P \cap B_P$, let the ray VP intersect A and B at points $f_A(x,y)$ and $f_B(x,y)$ closest to the viewer. For qualitative distance, the value of *InFront* is formally stated as follows.

Algorithm for depth *InFront* determination

If $A_P \cap B_P = \varnothing$

then there is no obscuration: *InFront ="na"*

elseif $\forall(x,y) \in A_P \cap B_P$,

if $f_A(x,y)=f_B(x,y)$, then A and B are equidistant: *InFront ="E"*

elseif $f_A(x,y) \leq f_B(x,y)$, then A obscures B: *InFront ="A*

elseif $f_A(x,y) \geq f_B(x,y)$,then B obscures A: *InFront ="B"*

elseif $\exists(x,y),(x',y') \in A_P \cap B_P$ such that $f_A(x,y)<f_B(x,y)$, and $f_A(x',y')>f_B(x',y')$ then A and B mutually obscure each other: *InFront ="M"*

end

Caution: For the sake of simplicity, we may loosely write InFront = A, B, E, M instead of InFront="A", "B", "E", "M".

Now the quad-valent distance parameter InFront is accurate description depth relation parameter. To make the occlusion relations self-documenting, we denote the occlusion predicate as xObs_z(A,B). Since some relations have converse while others do not, to make it completely symmetric, we have (1) z=a for "A is in front of B", (2) for converse, "B in front of A", we use z=b; (3) for equality, z=e if "A and B are *equidistant*", and (4) z=m for *mutual* occlusion when "partly A is in front of B and partly B is in front of A". This way it is easier to comprehend the predicates when z is used to describe depth. Clearly there is distinction between *mutual* and *equal* occlusion for QSR. For natural language expressiveness in classification, we will use two distinct predicates, one for *mutual* and one for *equal*, instead of combining them into one as has been done in the past [8].

5.2 Occlusion Predicates in Natural Language. The system has to interpret data as viewed by the observer. Most difficult part is the representation of spatial occlusion predicate with complete expressive power. With xObs_z(A,B), in essence x refers to the type of occlusion, n, p, m, e, c, exclusively and z refers to qualitative distance of the objects from the viewer. There are four types of distances for objects and five types of obscuration. Out of 20 relations, some combinations are impossible, only the possible combinations are described here.

The discussion of obscuration predicate is incomplete without reference to projections per se. It is not possible to know the type of occlusion apriori. As we define the term xObs_z(A,B) lucidly, we refine occlusion by integrating contribution of RCC5 relations, namely, $DR(A_P,B_P)$, $PO(A_P,B_P)$, $EQ(A_P,B_P)$, $PP(A_P,B_P)$, and $PPc(A_P,B_P)$. As such, we upgrade the predicate xObs_z to xObs***y***_z where y refers to the RCC5 relation in projection plane, x refers to type of projection, and z refers to relative distance parameter InFront. The complete listing of the predicates is given in Section 5.3 and for visual inspection a table 2 is given in Section5.4. This approach eliminates errors and leads to efficient reasoning.

Now *for x = n*, y in nObsy_z(A,B) corresponds to the relation $DR(A_P,B_P)$. In this case nObs will be true independent of the value of z, so z is not applicable for this. Therefore four versions of nObsy_z can be simplified to a single version nObsDR and the value of the InFront is "na",

Now *for x = p*, y in pObsy_z(A,B) corresponds to the relation $PO(A_P,B_P)$, $PP(A_P,B_P)$, and $PPc(A_P,B_P)$. From $PO(A_P,B_P)$ there are two relations for InFront value A, B. There are two other relations, one from $PP(A_P,B_P)$ with InFront equal to A, one from $PPc(A_P,B_P)$ with InFront equal to B. There are 4 *partial* obscurations in all. They are named descriptively where these come from.

Now *for x = e*, y in eObsy_z(A,B) corresponds to the relation $EQ(A_P,B_P)$, $PO(A_P,B_P)$, $PP(A_P,B_P)$, $PPc(A_P,B_P)$. In each case, there is one predicate for equal obscuration with InFront value E. There are 4 *equal* obscuration predicates.

Now *for x = m*, y in mObsy_z(A,B) corresponds to the relation $EQ(A_P,B_P)$, $PO(A_P,B_P)$, $PP(A_P,B_P)$, $PPc(A_P,B_P)$. In each case, there is one predicate for mutual obscuration with InFront value M. There are 4 *mutual* obscuration predicates.

Now *for x = c*, y in cObsy_z(A,B) corresponds to $EQ(A_P,B_P)$ or $PPc(A_P,B_P)$ or PP(A,B). From $EQ(A_P,B_P)$, there are two relations for InFront value A, B. There are two other relations, one from $PP(A_P,B_P)$ with InFront equal to B, one from $PPc(A_P,B_P)$ with InFront equal to A. There are 4 *complete* obscurations in all.

5.3 Occlusion Predicates using first order logic

Now that we have described the obscuration predicates in natural language, we will define them formally in first order logic as follows. The complete and comprehensive occlusion relations *nObs, pObs, eObs, mObs, cObs* supported with y for RCC5 relation between projections and z for parameter InFront are coherently denoted by xObsy_z.

There is one nObs occlusion relation:

nObsDR (A,B) $\equiv DR(A_P,B_P) \wedge$(InFront(A,B) == "na")

There are 4 types of pObsy_z occlusion relations. There are two predicates from $PO(A_P,B_P)$ with parameter InFront values A, B for partial obscuration. There is one predicate from $PP(A_P,B_P)$, with InFront=A and there is one predicate from $PPc(A_P,B_P)$, with InFront=B for partial obscuration, namely:

pObsPO_a(A,B)$\equiv PO(A_P,B_P) \wedge$(InFront (A,B) =="A")
pObsPO_b(A,B) $\equiv PO(A_P,B_P) \wedge$(InFront (A,B) =="B")
pObsPPc_b(A,B)$\equiv PPc(A_P,B_P) \wedge$ (InFront (A,B) =="B")
pObscPP_a(A,B) $\equiv PP(A_P,B_P) \wedge$ (InFront (A,B) =="A")

There are 4 types of eObsy_e occlusion relations. For y, there corresponds one predicate from each $PO(A_P,B_P)$, $EQ(A_P,B_P)$, PP(A_P,B_P), and PPc(A_P,B_P) with InFront=E when objects A and B are equidistant, namely :

eObsEQ_e(A,B) ≡ EQ(A_P,B_P)∧ (InFront (A,B) ==“E”)
eObsPO_e(A,B) ≡ PO(A_P,B_P)∧ (InFront (A,B) ==“E”)
eObsPPc_e(A,B)≡ PPc(A_P,B_P)∧ (InFront (A,B) ==“E”)
eObsPP_e(A,B) ≡ PP(A_P,B_P)∧ (InFront (A,B) ==“E”)

There are 4 types of mObsy_m occlusion relations. For y, there is one predicate from each PO(A_P,B_P), EQ(A_P,B_P), PP(A_P,B_P), and PPc(A_P,B_P) with InFront=M when objects A and B are properly cross, namely :

mObsEQ_m(A,B)≡EQ(A_P,B_P)∧(InFront (A,B) ==“M”)
mObsPO_m(A,B)≡PO(A_P,B_P)∧(InFront(A,B) == “M”)
mObsPPc_m(A,B)≡PPc(A_P,B_P)∧(InFront(A,B)==“M”)
mObsPP_m(A,B)≡PP(A_P,B_P)∧(InFront (A,B) == “M”)

There are 4 types of x=c occlusion relations. There are two from PP and PPc with values B and A and there are two from EQ with values of InFront as A, B, for complete obscuration.

cObsPPc_a(A,B)≡PPc(A_P,B_P)∧ (InFront (A,B) ==“A”)
cObsPP_b(A,B) ≡ PP(A_P,B_P)∧ (InFront (A,B) ==“B”)
cObsEQ_a(A,B) ≡ EQ(A_P,B_P)∧ (InFront (A,B) ==“A”)
cObsEQ_b(A,B) ≡ EQ(A_P,B_P)∧ (InFront (A,B) ==“B”)

This is a complete classification of seventeen JEPD unique occlusion relations, see Table 2.

		Infront			
		A	B	E	M
RCC5	DR		n		
	PO	p	p	e	m
	EQ	c	c	e	m
	PP	p	c	e	m
	PPc	c	p	e	m

Table 2 Let x be table entry, y=RCC5(A_P,B_P), z=InFront(A,B), then xObsy_z(A,B). The first row entry indicates, there is no occlusion irrespective of InFront value.

5.4 Tabular form of Occlusion Predicates

We have completely described the occlusion predicate x*Obs*y_z(A,B), see Table 2. Now we have one predicate for nObs, four predicates each for pObs, eObs, mObs, and cObs. There are seventeen JEPD predicates in all, described in natural language and in first order logic. We can reduce them to 7 by suppressing the y but adding the details of y in table entries: xObs_z, see Table 3. This is essential for display, but not necessary for development.

6. Implementation Consideration

For any theoretical development, its practical usefulness depends on the implementation followed its use in client applications. Clearly we have improved the theoretical representation of the solution to computation of obscuration classification relations. We presented crisp ontology for obscuration relations: xObsy_z.

Table 3 Complete Set of Occlusion Predicates

xObs_z	IntInt	IntBnd	BndInt	InFront	converse
nObs	F	F	F	na	nObs
pObs_a	T T	F T	T T	A	pObs_b
pObs_b	T T	T T	F T	B	pObs_a
eObs_e	T T T T	F F T T	F T F T	E	eObs_e
mObs_m	T T T T	F F T T	F T F T	M	mObs_m
cObs_a	T T	F T	F F	A	cObs_b
cObs_b	T T	F F	F T	B	cObs_a

First viewpoint and viewplane are selected. The objects are projected on the view plane. With the projections of the objects, RCC5 relations are determined using IntInt, IntBnd, and BndInt predicates with the 2D projections A_P and B_P of the 3D objects A and B, respectively.

The 5-step ***algorithm*** for obscuration detection becomes:

Algorithm for xObsy_z determination

Input: objects A and B, view point V and projection plane P.
Output: predicate **x**Obs**y_z**
1. Project Objects A and B, determine A_P and B_P
2. Determine RCC5 relations between A_P and B_P
3. Determine InFront parameter values
4. Integrate steps 2 and 3 to classify the obscuration type

There are standard algorithms for step 1, and 2. The step 3 is most complex and computation intensive in practice. In order to determine the obscuring object, as shown in Section 5.4, a semi-infinite ray is drawn from viewer through points in $A_P \cap B_P$ and analyzed for intersection with the objects. This is a computation intensive step as it is repeated thousands of times depending on digitization of projection plane.

Computations of ray intersections in step 3 can be eliminated altogether by judiciously performing the step 1. As soon as the projection is computed, we know the functional relation between objects and their projections: A to A_P, and B to B_P. We can record it in step 1 to use it in step 3 as a lookup table to avoid repeated ray intersections. This can be done with an appropriate intelligent grid data structure that keeps track of the closest *intersection* points on the objects. Now for step 3, we can look up the computed value for each A_P grid point. This eliminates tens of thousands of ray-object intersection computations.

By using this heuristic the algorithm can be implemented very efficiently. Test case was written in Python and implemented on Apple, using synthetic data of 500 objects of various shapes. The simulation showed a remarkable improvement. Computation efficiency will increase significantly if one object information is reused for obscuration with several other objects in the application.

6.1 Hybrid Spatial and Occlusion Relations

The same reasoning also works for mobile objects and stationary observer. Topological relation is a static RCC8 relation for 3D objects. Static relation is independent of the observer, it is the same for every observer. The occlusion relation (xObsy_z) is the spatial dynamic relation as seen by the viewer. The dynamic occlusion relation varies from viewer to viewer location. By consolidating the two, we have complete hybrid spatial relations. If R is an RCC8 relation in 3D, and xObsy_z is occlusion relation, then R and xObsy_z in tandem coalesce to represent coherent spatial relation, **R_ x**Obs**y_z**.

For the sake of simplicity and space availability, we suppress y, and display the relations in the form R_xObs_z. There are 8 RCC8 connectivity relations and 7 xObs_z occlusion relations. Not all obscuration relations are physically possible with each RCC8 relation. There are 23 hybrid relations: 5 DC, 5 EC, 6 PO, 1 EQ, 2 TPP, 2 TPPC, 1 NTPP, 1 NTPPc relations, see Fig. 8.

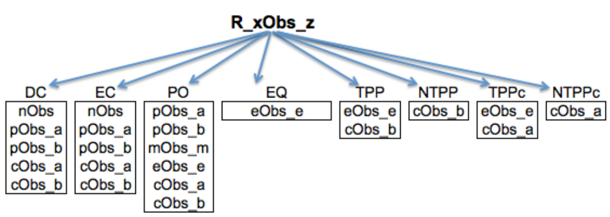


Fig. 8. A hierarchy tree of composite spatial relations.

7. Conclusion and Future Directions

We have given a complete description and classification of qualitative spatial occlusion relations for 3D objects as seen by an observer. The same reasoning works well with the objects that are mobile and the observer is stationary. The spatial relations are self-documenting and easy to understand. We optimized the set of occlusion relations from 12 to 7 and reduced the composite relations from 34 to 23. These computations are performed repeatedly in any application. This development will be useful in GIS, robotic sensors for navigation, biomedicine, and related areas. Conceptual neighborhoods and composition tables are integral part of any qualitative spatial reasoning system, we plan to develop these ideas to produce conceptual neighborhood graphs and composition tables. This work also supersedes the existing 3D spatial reasoning systems.

8. References

[1] B. Bennett.: *Spatial Reasoning With Propositional Logics*, Proceedings of the 4th International Conference on Principles on Knowledge Representation and Reasoning (KR-94), Bonn, Germany, pp. 165-176, 1994.

[2] Galata, A., Cohn, A., Magee, D., and Hogg, D.: *Modeling interaction using learnt qualitative spatio-temporal relations and variable length markov models*. In *ECAI*, pp. 741–746, 2002.

[3] A.P. Galton, Lines of Sight, in *AISB Workshop on Spatial and Spatio-Temporal Reasoning*, 1994.

[4] D.A. Randell, M. Witkowski, and M. Shanahan, *From Images to Bodies: Modelling and Exploiting Spatial Occlusion and Motion Parallax*, *IJCAI-01*, pp. 57-63, 2001.

[5] Kohler, C.: *The Occlusion Calculus*. In: Cognitive Vision Workshop. ICVW ’02 , pp. 1-6, 2002.

[6] J. Albath, J. Leopold, and C. Sabharwal, Visualization of Spatio-Temporal Reasoning Over 3D Images, *Proceedings of the 2010 International Workshop on Visual Languages and Computing* (in conjunction with the *16th International Conference on Distributed Multimedia Systems*), pp. 277-282, 2010.

[7] Guha, P., Mukerjee, A., Venkatesh, K.S.: OCS-14 : *You Can Get Occluded in Fourteen Ways*. In: Walsh, T. (ed.) Proceedings of the 22nd International Joint Conference on Artificial Intelligence. pp. 1665–1670, 2011.

[8] C. L. Sabharwal, J. L. Leopold, and N. Eloe: *A More Expressive 3D Region Connection Calculus* In The 2011 International Workshop on Visual Languages and Computing (in conjunction with the 17th International Conference on Distributed Multimedia Systems (DMS'11)), pp. 307-311, 2011.

[9] N. Eloe, C. Sabharwal and J. Leopold: *A More Efficient Representation of Obscuration for VRCC-3D+ Relations*, Polibits Journal of Computer Science, pp. 29-34, 2014.

[10] D.A. Randell, Z. Cui, A.G. Cohn, *A Spatial Logic Based on Regions and Connection* KR92:165–176, 1992.

[11] M. J. Egenhofer, R. Franzosa, *Point-Set topological Relations,* International Journal of Geographical Information Systems 5(2), pp.161-174, 1991.