Departament D'Enginyeria I Ciència dels Computadors

Universitat Jaume I.

# QUALITATIVE THEORIES ON SHAPE REPRESENTATION AND MOVEMENT.

## APPLICATION TO INDUSTRIAL MANUFACTURING AND ROBOTICS

Ph. D. Thesis

**Lledó Museros Cabedo**

Supervisors:

Mª Teresa Escrig

*Universitat Jaume I, Castellón, Spain*

and

Christian Freksa

*University of Bremen, Germany*

Castelló, 2006

*In memory of my father, Manolo, who always encouraged me.*
*To my mother, Ana Maria, for her complicity and understanding.*
*To my family and to you, Ismael, for being always when I need you.*

DEPARTAMENT D'ENGINYERIA I CIÈNCIA DELS COMPUTADORS

UNIVERSITAT JAUME I.

# TEORÍAS CUALITATIVAS DE REPRESENTACIÓN DE FORMAS Y MOVIMIENTO.

## APLICACIÓN A LA ROBÓTICA Y A LA INDUSTRIA.

TESIS DOCTORAL

**Lledó Museros Cabedo**

Directores:

Mª Teresa Escrig

*Universitat Jaume I, Castellón, España*

y

Christian Freksa

*Universidad de Bremen, Alemania*

Castelló, 2006

*A la memoria de mi padre, Manolo, por su estímulo continuo.*

*A mi madre, Ana María, por su complicidad y comprensión.*

*A mi familia, y a ti, Ismael, por estar siempre ahí.*

# ABSTRACT

Since the end of 80's there has been great interest in the study of qualitative models to represent and to reason with spatial aspects. The present work is centred on the development and application of a model to reason about shape and about movement in a qualitative way, which means in a way similar to the human reasoning. The interest of this study originates from the necessity to find solutions for the recognition of objects and the description and reasoning about the movement in situations with high uncertainty, as it is the case of robotic applications, where robots only have limited and vague sensorial information. In these situations the use of a qualitative reasoning, that allows us to handle ambiguities and errors, will be the most suitable.

The movement of an object can be considered as a shape whose topologic relation with its environment (considered as another shape) changes in time. On the other hand the shape of the objects is a spatial aspect in itself, and again for its study we have used topological concepts. The recognition of objects is important during the movement of a robot since for the accomplishment of certain tasks the robot must be able to recognize the objects which it comes across during its trajectory, since these objects can be landmarks or reference points that provide the robot with spatial information about its environment.

Therefore this work will be centred on the study of three aspects of space: the shape of the objects, the topology and the movement. Several works exist about the shape of the objects [Jungert 94; Park and Gero 99, 00; Chase 96, 97; Shokoufandeh, Dickinson et al. 02], on topology [Cohn, Bennett ET al. 97; Renz & Nebel 98; Egenhofer & Franzosa 91; Clementini & Di Felice 95] and on movement [Zimmermann and Freksa 93; Musto, Stein et al. 00; Musto et al. 99; Rajagopalan and Kuipers 94; Forbus 83; Muller 98a, 98b], that will be studied in greater depth during the chapter 2, which describes a state-of-the-art of each one of them. But, most of these works are theoretical and they have not been applied to robotics.

This PhD thesis presents a motion model as a qualitative representational model for integrating qualitatively time and topological information for reasoning about dynamic worlds in which spatial relations between regions and between regions and objects may change with time. This qualitative integration of time and topology has been accomplished thanks to the definition of an approach with the following three steps: (1)

the definition of the algebra of the spatial aspect to be integrated, which will be time and topology. The representation of each aspect is seen as an instance of the Constraint Satisfaction Problem (CSP); (2) the definition of the Basic Step of the Inference Process (BSIP) for each spatial aspect to be integrated. In general, the BSIP consists of giving two relationships which relate three objects A, B, and C (one object is shared among the two relationships, for instance A is related with B and B is related with C), we will find the third relationship between objects A and C; and (3) the definition of the Full Inference Process (FIP) for each spatial aspect to be integrated which consists of repeating the BSIP as many times as possible with the initial information and the information provided by some BSIP, until no more information can be inferred.

On the other hand, the theory for the recognition of shapes developed is able to describe several types of shapes, as they are regular and non-regular polygons, with or without holes, with or without curved segments and even completely curvilinear forms. The theory describes shapes considering qualitatively the angles, relative side length, concavities and convexities, and types of curvatures of their boundaries using only their relevant points, which are defined as vertices, and the initial, final point and point of maximum curvature of the curves. To describe shapes with holes, topological and qualitative spatial orientation aspects have been considered in order to relate the hole with its container. Each object is described by a string which describes its qualitative distinguished features (symbolic representation), which is used to match an object against the others. This theory has been applied, in an industrial domain, for the automatic and intelligent assembly of ceramic mosaics. Mosaics are made of pieces of different shapes, colours and sizes, named tesseraes, which once they are assembled create a unique composition with high added value, due to its artistic and decorative value. Mosaics are usually made following a design describing the position of each tesserae in the final composition. The application developed in this dissertation, recognises individual tesseraes from pictures, which represent the tesserae coming over a conveyor, against a vectorial mosaic design. Therefore, the application returns the position of the tesserae in the mosaic together with the angle that a robot arm has to adopt when picking the tesserae up by its centroid in order to place it in the correct orientation inside the mosaic. On the other hand the simplest version of this theory, especifically the part that describes regular and non-regular polygonal objects, jointly with the developed theory of movement has been applied too for the simulated navigation of a real robot, namely the Khepera[2] robot. This application consists of a world formed by two rooms connected by a corridor. The robot first learns the topological map of the world. Then in each room there is an object and the robot has to decide if both objects represent the same object or not, for that purpose the robot uses the movement theory to plan its route and to detect possible deviations during its moving, and finally by using the qualitative theory for shape matching developed decides if the objects have the same shape or not. Both applications are described with greater detail in chapter 6.

# RESUMEN

## 1. Introducción

Desde finales de los años 80 ha habido un gran interés en el estudio de modelos cualitativos para representar y razonar con aspectos espaciales. El presente trabajo se centra en el desarrollo y aplicación de un modelo para razonar acerca de la forma y del movimiento de un modo cualitativo, es decir de manera similar al razonamiento humano. El interés de dicho estudio se origina en la necesidad de soluciones para el reconocimiento de objetos y para la descripción y razonamiento acerca del movimiento en situaciones con elevada incertidumbre, como es el caso de aplicaciones robóticas, donde los robots disponen únicamente de información sensorial limitada e imprecisa. En estas situaciones el uso de un razonamiento cualitativo, que permite manejar ambigüedades y errores, será el más adecuado.

El movimiento de un objeto puede considerarse como una forma cuya relación topológica con su entorno (considerado como otra forma) cambia en el tiempo. Por otro lado la forma de los objetos es un aspecto espacial en sí mismo, y de nuevo para su estudio hemos utilizado conceptos topológicos. El reconocimiento de objetos es importante durante el movimiento de un robot puesto que para la realización de determinadas tareas el robot debe ser capaz de reconocer los objetos con los que se encuentra durante su trayectoria, ya que dichos objetos pueden ser "landmarks" o puntos de referencia que proporcionan al robot información espacial de su entorno.

Por tanto en este trabajo nos centraremos en el estudio de tres aspectos espaciales: la forma de los objetos, la topología y el movimiento. Existen diversos trabajos acerca de la forma de los objetos [Jungert 94; Park and Gero 99, 00; Chase 96, 97; Shokoufandeh, Dickinson et al. 02] , sobre topología [Cohn, Bennett et al. 97; Renz & Nebel 98; Egenhofer & Franzosa 91; Clementini & Di Felice 95] y sobre movimiento [Zimmermann and Freksa 93; Musto, Stein et al. 00; Musto et al. 99; Rajagopalan and Kuipers 94; Forbus 83; Muller 98a, 98b], que serán estudiados en mayor profundidad realizando un estado del arte de cada uno de ellos como se describe en la sección 2.1. de este resumen. La mayoría de estos trabajos son teóricos y no han sido aplicados a la robótica.

Las teoría para el reconocimiento de formas desarrollada es capaz de describir diversos tipos de formas, como son polígonos regulares y no-regulares, con o sin agujeros, con o sin segmentos curvos e incluso formas completamente curvilíneas. Esta teoría ha sido aplicada en dos vertientes distintas, en una vertiente industrial, para el ensamblaje automático e inteligente de mosaicos cerámicos. Mientras que la versión más simple de ésta teoría, en concreto la parte que describe objetos poligonales regulares y no-regulares, conjuntamente con la teoría de movimiento desarrollada han sido aplicadas en una más académica para la navegación simulada de un robot real, en concreto del robot Khepera[2]. Ambas aplicaciones son descritas con mayor detalle en la sección 2.5 de este resumen.

## 2. Tareas

### 2.1 Estudio de los modelos de razonamiento espacial cualitativo existentes.

Esta tarea consiste en la realización de tres estados del arte:

1. Estado del arte en modelos sobre topología, dado que la topología es un factor interesante tanto para el desarrollo de una teoría cualitativa de reconocimiento de formas como para el desarrollo de una teoría de movimiento vista como la integración de dos conceptos: espacio, representado con conceptos topoló gicos, y tiempo.
2. Estado del arte en modelos sobre movimiento cualitativo, dado que pretendemos desarrollar una teoría del movimiento y aplicarla a la robótica.
3. Estado del arte en modelos para el reconocimiento de formas cualitativo, dado nuestro objetivo de desarrollar una teoría cualitativa de reconocimiento de formas aplicado a la robótica.

A continuación se presenta un breve resumen de los diversos estudios realizados.

#### *2.1.1.ESTADO DEL ARTE SOBRE MODELOS TOPOLÓGICOS.*

Hasta la actualidad se han desarrollado diversos trabajos sobre relaciones topológicas dentro del campo del razonamiento espacial cualitativo. Estos trabajos los podemos dividir en dos tendencias: los trabajos para los que la relación espacial básica es la "región espacial", y los trabajos que consideran una región como un conjunto de puntos. En el primer grupo, como "región espacial" se entiende todas aquellas regiones espaciales regulares que no sean vacías, de ahí que los puntos, las líneas y las fronteras no pueden considerarse como regiones espaciales [Gotts 1996; Bennett 1994; Renz and Nebel 1998; Cohn, Bennett et al. 97]. Para estos trabajos la relación básica es C(x,y) (x conecta con y). C(x,y) aparece cuando la clausura topológica de x e y comparten al menos un punto. Utilizando la relación C(x,y) se definen un conjunto de relaciones topológicas atómicas que son mutuamente exclusivas y es un conjunto completo, es decir, dadas dos regiones una y solo una de las relaciones topológicas atómicas puede aparecer entre ellas. Una teoría de gran importancia dentro de este grupo es la teoría Region Connection Calculus (RCC) desarrollada en [Randell et al. 92].

El segundo conjunto de trabajos, consideran una región como un conjunto de puntos, por ello se denominan "Teorías de Conjunto de Puntos". Para estas teorías las entidades básicas son puntos, líneas y áreas y las relaciones topológicas entre estas entidades se definen en función de las intersecciones de los interiores y las fronteras de cada conjunto de puntos [Egenhofer and Franzosa 1991; Pullar and Egenhofer 1988; Egenhofer 1991; Clementini and Di Felice 1995]. Cada trabajo dentro de este grupo define su propio conjunto de relaciones topológicas atómicas.

### *2.1.2 ESTADO DEL ARTE SOBRE MODELOS DE MOVIMIENTO CUALITATIVO*

Cohn and Harazika afirman en [Cohn and Hazarika 01] que en la actualidad se ha desarrollado poco trabajo sobre el movimiento bajo una perspectiva cualitativa. En [Freksa and Zimmermann 93], [Musto, Stein et al. 00] y [Musto et al. 99] se estudia el movimiento de manera cualitativa. Pero en la mayoría de estos trabajos el movimiento se modela como una secuencia de cambios de posiciones, considerando para ello el concepto de vecindad conceptual, pero sin integrar el concepto de tiempo en el mismo modelo. Un trabajo que si que integra el concepto de tiempo en el mismo modelo es el trabajo de [Escrig and Toledo 02] que introduce un álgebra cualitativa para representar y razonar con la velocidad.

### *2.1.3 ESTADO DEL ARTE SOBRE MODELOS CUALITATIVOS PARA EL RECONOCIMIENTO DE FORMAS*

La mayoría de trabajos cualitativos para la descripción de formas se pueden clasificar en cinco grupos como se describe a continuación:

- *Representaciones axiales*: estos trabajos se basan en la descripción de los ejes del objeto, describiendo la forma cualitativamente al reducirla a un "skeleton" o "axis". El "axis" es el arco que refleja alguna simetría o regularidad local dentro de la forma. La forma se puede generar desde el "axis" mediante el desplazamiento de una figura geométrica (denominada "generator") a lo largo del axis. El "generator" es una forma constante que mantiene un punto específico (por ejemplo el centro) pero que cambia su tamaño e inclinación con respecto al axis. [Leyton 88 y Brady 83] son algunos de los trabajos dentro de este grupo.

- *Representaciones basadas en primitivas*: en estos trabajos los objetos más complejos se describen como combinaciones de objetos más primitivos y simples. Dentro de este grupo se pueden distinguir dos tendencias:
  - *Cilindros generalizados o representaciones basadas en "geones"*, que describen un objeto como un conjunto de primitivas junto con un conjunto de relaciones espaciales de conectividad entre ellas. [Biederman, 1987; Flynn & Jain, 1991].
  - *Representaciones constructivas*, que describen un objeto como la combinación Booleana de conjuntos de puntos primitivos [Park & Gero, 1999; Brisson, 1989; Ferrucci & Paoluzzi, 1991].
- *Representaciones basadas en proyecciones y en el ordenamiento de la información*. Dentro de este grupo diversos aspectos de la forma de un objeto se representan bien observando el objeto de diferentes ángulos, bien proyectando el objeto a diferentes ejes [Jungert, 1994; Schlieder, 1996; Freeman & Chakravarty, 1980; Park & Gero, 1999; Damski & Gero, 1996].
- *Representaciones topológicas y basadas en la lógica*. Los trabajos dentro de este grupo utilizan topología y lógica para representar las formas [Cohn, 1995; Randell & Cui & Cohn, 1992; Clementini & Di Felice, 1997].
- *Representaciones basadas en Recubrimientos*. En estos trabajos la forma des objeto se describe recubriendo el objeto con figuras simples, como rectángulos y esferas [Del Pobil & Serna, 1995].

## 2.2 Desarrollo de un Álgebra sobre Relaciones Topológicas adecuado a nuestros intereses.

Se trata de desarrollar un álgebra para razonar sobre relaciones topológicas que se pueda integrar con el trabajo realizado en [Escrig & Toledo 01], puesto que son las líneas seguidas por el grupo de investigación ya que se trata de uno de los pocos trabajos que integra diversos aspectos espaciales de manera cualitativa y permite no solo la representación de todos ellos bajo un mismo modelo sino razonar con ellos. El trabajo de [Escrig & Toledo 01] integra orientación 2D, distancia comparada y absoluta y direcciones cardinales. Con esta tarea se integra en el mismo modelo la capacidad de representar y razonar acerca de relaciones topológicas.

El trabajo dentro de esta tarea ha sido realizado desarrollando un álgebra adecuada para razonar acerca de relaciones topológicas entre entidades representadas como puntos, líneas y áreas. Se trata de un trabajo basado en restricciones inspirado en el trabajo desarrollado en el "Calculus Based Method" (CBM) de Clementini, Di Felice y Oosterom [Clementini & Di Felice, 1995; Clementini et al., 1993]. Para ello se ha desarrollado un álgebra basada en aquella para intervalos temporales de Allen (1983). Las relaciones del álgebra desarrollada son las 4 relaciones atómicas del cálculo CBM junto con otras 3 relaciones provenientes del refinamiento de la relación topológica in del CBM, puesto que habitualmente nos interesa conocer si una región esta completamente dentro de otra, si bien está tocando la segunda región desde dentro, o bien si ambas regiones son iguales. Además, dentro del trabajo desarrollado se calcula los resultados de aplicar dos operaciones: la inversa y la composición a cada una de las relaciones atómicas. Estos resultados componen la tabla de inversas y las tablas de composición que juegan un papel central en la propagación del conocimiento en el álgebra utilizando el algoritmo de propagación de restricciones, es decir a la hora de razonar [Museros 98; Isli, Museros et al. 00; Museros and Escrig 01a; Museros and Escrig 01b y Museros and Escrig 01c]. Con ello se definen las siguientes relaciones topológicas:

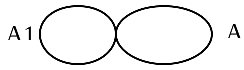
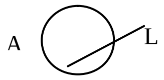
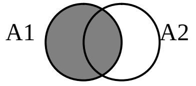
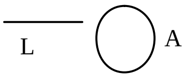
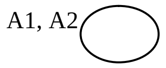
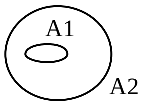
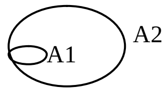
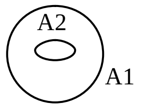
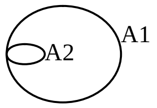
| Relación | Definición | Ejemplo Gráfico |
|---|---|---|
| *touch* | ($h_1$, touch, $h_2$) ↔ $h°_1 \cap h°_2 = \emptyset \wedge h_1 \cap h_2 \neq \emptyset$ | <br> |
| *cross* | ($h_1$,cross,$h_2$)↔dim($h°_1 \cap h°_2$)= max(dim($h°_1$),dim($h°_2$)) – 1 $\wedge$ $h_1 \cap h_2 \neq h_1 \wedge h_1 \cap h_2 \neq h_2$ | <br> |
| *overlap* | ($h_1$, overlap, $h_2$)↔dim($h°_1$)= dim($h°_2$)= dim($h°_1 \cap h°_2$) $\wedge$ $h_1 \cap h_2 \neq h_1 \wedge h_1 \cap h_2 \neq h_2$ | <br> |
| *disjoint* | ($h_1$, disjoint, $h_2$) ↔ $h_1 \cap h_2 = \emptyset$ | <br> |
| *equal* | Dado ($h_1$, in, $h_2$) ↔ $h_1 \cap h_2 = h_1 \wedge h°_1 \cap h°_2 \neq \emptyset$:<br>if ($h_2$, in, $h_1$) then ($h_1$, equal, $h_2$) | <br> |
| *completely-inside* | Dado ($h_1$, in, $h_2$) y not ($h_1$, equal, $h_2$):<br>if $h_1 \cap \delta h_2 \neq \emptyset$ then ($h_1$, touching-from-inside, $h_2$) | <br> |
| *touching-from-inside* | If ($h_1$, in, $h_2$), not ($h_1$, equal, $h_2$) and not ($h_1$, touching-from-inside, $h_2$) then: ($h_1$, completely-inside, $h_2$) | <br> |
| *completely-inside$_i$* | ($h_1$, completely-inside$_i$, $h_2$) ↔ ($h_2$, completely-inside, $h_1$) | <br> |
| *touching-from-inside$_i$* | ($h_1$, touching-from-inside$_i$, $h_2$) ↔ ($h_2$, touching-from-inside, $h_1$) | <br> |

***Tabla i.*** *Definición y representación gráfica de las relaciones topológicas definidas.*

## 2.3 Desarrollo de un Álgebra para el Movimiento: Integración de Topología y Tiempo.

El movimiento puede verse como una forma de cambio espacio-temporal, es por ello que dentro de esta tarea se ha desarrollado un álgebra que permita representar y razonar sobre el movimiento del mismo modo al realizado en [Escrig & Toledo 01] integrando para ello 2 tipos de conceptos: topología (información espacial) y tiempo. De este modo se desarrolla un modelo de representación del movimiento que permite razonar en mundos dinámicos en los que las relaciones espaciales entre las regiones (considerando al robot como una región en si mismo) cambian con el tiempo.

Esta tarea se ha llevado a cabo formalizando la noción de continuidad espacio-temporal en una teoría cualitativa del movimiento en diversos artículos como se detalla a continuación: en el informe técnico [Museros and Escrig 02a] se presenta el trabajo preliminar acerca de la teoría del movimiento desarrollada; [Museros and Escrig 02b] presenta la teoría de movimiento finalmente desarrollada; [Museros and Escrig 02c] explica la aplicación de la teoría de movimiento a un problema de robótica móvil; [Museros and Escrig 02d] desarrolla la integración de la teoría del movimiento con otros aspectos espaciales como orie ntación y distancia, y dada la calidad de este artículo se nos invito a extenderlo y publicarlos en la revista Journal of Universal Computer Science [Museros and Escrig 03]; [Museros and Escrig 02e]; [Museros and Escrig 02f] desarrollan la teoría de movimiento como un modelo de satisfacción de restricciones; y finalmente [Museros and Escrig 02g] se trata de una publicación en la revista Inteligencia Artificial de diversos modelos desarrollados en el grupo de investigación integrando espacio (2-D y 3-D) y tiempo, como son velocidad, y movimiento.

Las bases para la integración de diferentes aspectos espaciales en el campo del razonamiento espacial se han inspirado en el trabajo realizado en el razonamiento temporal, donde se ha integrado con éxito el álgebra de puntos, el álgebra intervalar y la información métrica [Meiri 91]. Para llevar a cabo la misma tarea integrando diversos aspectos espaciales en el mismo modelo se definen 3 pasos a seguir [Escrig and Toledo 00]:

- La representación de cada aspecto espacial a ser integrado.
- La definición del Paso Básico de Inferencia (BSIP). Este paso se define como dada la relación espacial entre los objetos A y B, y la relación espacial entre los objetos B y C, el BSIP consiste en obtener la relación espacial entre los objetos A y C.
- La definición del Proceso de Inferencia Completo (FIP), que consiste en repetir el BSIP tantas veces como sea posible, con la información inicial y la información obtenida en previos pasos por el BSIP, hasta que no obtengamos más información.

En el trabajo realizado, topología junto con información temporal se integran siguiendo esta idea. Para definir el movimiento como integración de espacio y tiempo, el cálculo topológico desarrollado en la tarea anterior se integra a un álgebra temporal que hemos definido. En el álgebra temporal definida las variables representas puntos en el tiempo, y con ello se definen 5 relaciones primitivas: ==, next, prev, >>, <<. Para completar el álgebra temporal se definen las operaciones inversa y composición de estas relaciones.

Con todo ello se define el álgebra del movimiento cualitativo, en la cual las relaciones binarias entre 2 objetos que pueden ser puntos, líneas o áreas, denominados $h_1$ y $h_2$ en un punto del tiempo t se definen como restricciones terciarias o proposiciones, donde la relación topológica r entre $h_1$ y $h_2$ en el punto del tiempo t se denota como $(h_1,r,h_2)_t$. Para el álgebra dada, se construyen las tablas de composición y de inversas, un ejemplo de una de las tablas de composición creadas se muestra en las tabla ii). Con estas tablas se define tanto el BSIP como el FIP, lo cual permitirá razonar con el movimiento cualitativo.

| **Relación Temporal ®** | **next o prev** | << **o** >> | == |
|---|---|---|---|
| **Relación Espacial ¯** | | | |
| *Touch (T)* | {C,D,T} | {T,TFI,CI} | {T} |
| *Cross (C)* | {T,TFI,C} | {C,D,CI} | {C} |
| *Disjoint (D)* | {T,D} | {D,C,TFI,CI} | {D} |
| *Touching- From-Inside (TFI)* | {C,CI,TFI} | {TFI,T,D} | {TFI} |
| *Completely-Inside (CI)* | {TFI,CI} | {CI,T,C,D} | {CI} |

***Tabla ii).*** *Tabla LAt-Tabla, que muestra la compasión de h1 = línea y h2 = área en el momento del tiempo $t_1$ y $t_2$ respectivamente.*

La teoría de movimiento desarrollada es útil para permitir a un robot razonar sobre la secuencia de situaciones topológicas que se encontrará durante su navegación de una región inicial a una región objetivo. De este modo, el robot será capaz de detectar situaciones en las cuales este perdiendo su dirección de movimiento. Por ejemplo, si tenemos como situación inicial la descrita en la figura i) en el momento del tiempo $t_0$, y queremos que el robot navegue de la región 1 a la región 2, con el álgebra de movimiento desarrollada sabemos que la secuencia de relaciones topológicas entre el robot (interpretado como una región en sí mismo) y ambas regiones será la siguiente (la figura i) muestra esta secuencia gráficamente):

(Robot,Completely-Inside$_i$,Region$_1$)$t_0$ and (Robot,Disjoint,Region$_2$)$t_0$,

(Robot,Touching-From-Inside$_i$,Region$_1$)$t_1$ and (Robot,Touch,Region$_2$)$t_1$,

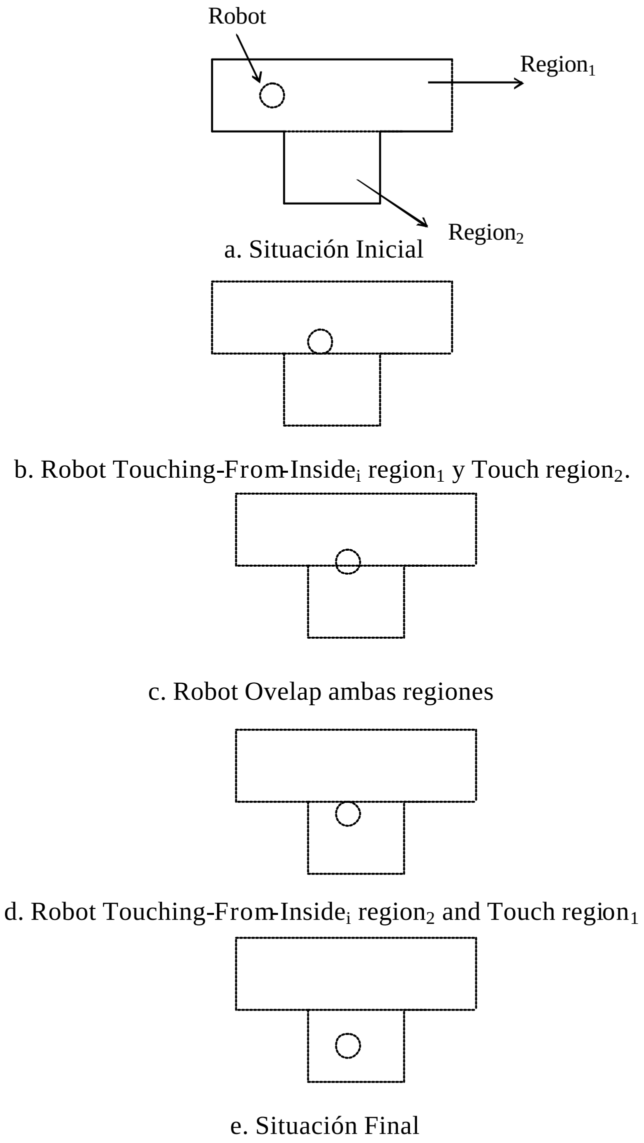
(Robot,Overlap,Region$_1$)$t_2$ and (Robot,Overlap,Region$_2$)$t_2$,

(Robot,Touch,Region$_1$)$t_3$ and (Robot,Touching-From-Inside$_i$,Region$_2$)$t_3$,

(Robot,Disjoint,Region$_1$)$t_4$ and (Robot,Completely-Inside$_i$,Region$_2$)$t_4$

donde $t_0$ prev $t_1$ prev $t_2$ prev $t_3$ prev $t_4$.

Si en algún momento se encuentra con una situación topológica diferente a la esperada según la secuencia descrita, el robot está perdiendo su dirección del movimiento y podemos tomar las medidas oportunas para solventarlo.



a. Situación Inicial

b. Robot Touching-From-Inside$_i$ region$_1$ y Touch region$_2$.

c. Robot Ovelap ambas regiones

d. Robot Touching-From-Inside$_i$ region$_2$ and Touch region$_1$

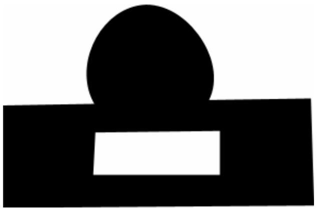
e. Situación Final

***Figura i).*** *Representación Gráfica de la secuencia topológica seguida para que el robot navege de la región 1 a la región 2.*

## 2.4 Desarrollo de una Teoría Cualitativa para el Reconocimiento de Objetos.

Finalmente se desarrollará una teoría de reconocimiento de formas, capaz de reconocer tanto polígonos regulares como no regulares, siempre y cuando sean cerrados, como objetos con segmentos curvos o completamente curvilíneos, ambos tipos con o sin agujeros.

Esta tarea se ha llevado a cabo desarrollando un trabajo para la descripción cualitativa de formas considerando sus ángulos, longitud comparada de las aristas, concavidades y convexidades y tipos de curvatura de las fronteras de las formas junto con el área y color de los objetos ambos tratados cualitativamente. Para la descripción de formas con agujeros se utilizan dos conceptos para relacionar cada agujero con su contenedor: topología y orientación cualitativa espacial. Cada objeto se describe como un "string" que contiene las características cualitativas relevantes del objeto, y este "string" es el utilizado en el proceso de correspondencia del objeto frente a otros objetos. Un ejemplo del "string" resultante al aplicar la teoría desarrollada se muestra en la figura ii).



QualShape(S)=[with-holes, with-curves, [[0,0,0], [right-angle,convex,bigger], [curve, convex, acute], [right-angle, convex,bigger], [right-angle, convex, smaller], [right-angle, convex,bigger]],CIi,C,[[right-angle, convex, smaller], [right-angle, convex, bigger],[right-angle, convex, smaller], [right-angle, convex, bigger]]].

***Figura ii).*** *Ejemplo de una figura y su descripción cualitativa (QualShape(S)) de la misma utilizando la teoria de reconocimiento de formas desarrollada.*

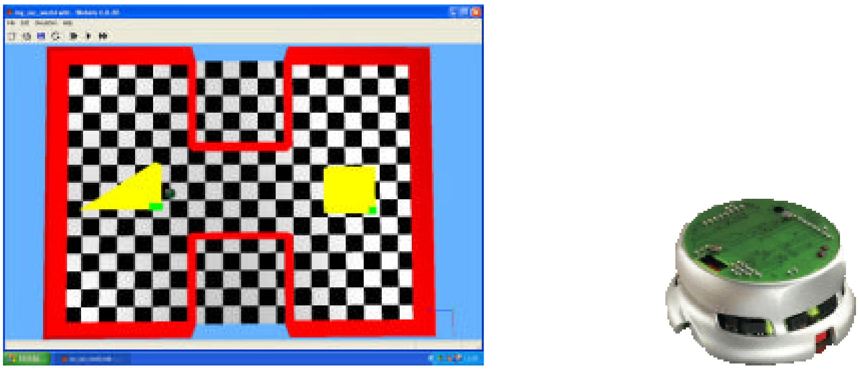
La teoría desarrollada ha sido presentada en diversos artículos como se describe a continuación: [Museros and Escrig 04a] describe la fase inicial de la teoría cualitativa, que permite la descripción y reconocimiento únicamente de objetos poligonales; [Museros and Escrig 04c] describe la teoría cualitativa ampliada para describir y reconocer objetos poligonales y con curvas y [Museros and Escrig 04d] describe la teoría cualitativa final capaz de reconocer objetos poligonales, con curvas y con uno o varios agujeros.

## 2.5 Aplicación de las Teorías Desarrolladas.

Se han implementado dos aplicaciones diferentes, una de ellas centrada en la robótica móvil y la otra en la industria cerámica.

### *2.5.1. APLICACIÓN EN EL CAMPO DE LA ROBÓTICA MÓVIL.*

Por un lado tanto la teoría de movimiento que integra el álgebra topológica y tiempo, como la teoría para el reconocimiento de formas en su versión más simple (reconocimiento de formas poligonales no regulares) se han aplicado en un primer paso a la navegación simulada de un robot real, en concreto del robot Khepera[2], utilizando como simulador el entorno Webots de Cyberbotics. Una vez simulado el software realizado se ha probado en el robot real. El mundo simulado se muestra en la siguiente figura, y esta compuesto por dos habitaciones conectadas por un pasillo en las cuales se colocan 2 objetos que pueden ser iguales o no. El robot (figura iiib), que inicialmente no conoce el entorno, en primer lugar construye su mapa topológico del entorno, y a continuación busca el primer objeto en una de las habitaciones, lo recorre creando su descripción cualitativa y pasa a buscar el siguiente objeto utilizando el álgebra de movimiento para evitar perderse. Cuando encuentra el segundo objeto lo recorre y realiza su descripción cualitativa, que compara con la inicial de manera cíclica y determina si ambos objetos poseen la misma forma o no [Museros and Escrig 04e]. El mundo simulado se ha construido a escala para el robot real.



a) b)

***Figura iii).*** *a) Mundo simulado y b) robot utilizado.*

### *2.5.2. APLICACIÓN EN EL CAMPO DE LA INDUSTRIA CERÁMICA.*

Por otro lado, toda la teoría de reconocimiento de formas desarrollada se aplicará al reconocimiento de teselas individuales captadas a partir de un sistema de visión frente a un diseño gráfico de un mosaico cerámico. Esta aplicación puede ser utilizada en un futuro para su aplicación en un brazo manipulador real para el ensamblaje de mosaicos cerámicos. Esta tarea se ha completado creando un software que a partir de imágenes de teselas capturadas con una cámara digital, el software trata dichas imágenes de manera semi-cualitativa [Museros and Escrig 05] para obtener los puntos relevantes de dicha imagen y de ese modo poder crear la descripción cualitativa de las formas de dichas imágenes. Las imágenes se tratan semi-cualitativamente aplicando inicialmente el algoritmo cuantitativo de Canny [Canny 86] y posteriormente se tratan de manera cualitativa comparando las pendientes entre puntos consecutivos tomados en función de una granularidad, para obtener los puntos relevantes en coordenadas (x,y) absolutas respecto a la imagen y el tipo de segmento entre dos puntos relevantes, bien se trata de un segmento recto (denotado por el símbolo straight-line) o una curva (denotado por el símbolo curve). Dada la imagen que se muestra en la figura iv)a, los puntos relevantes obtenidos son los siguientes:

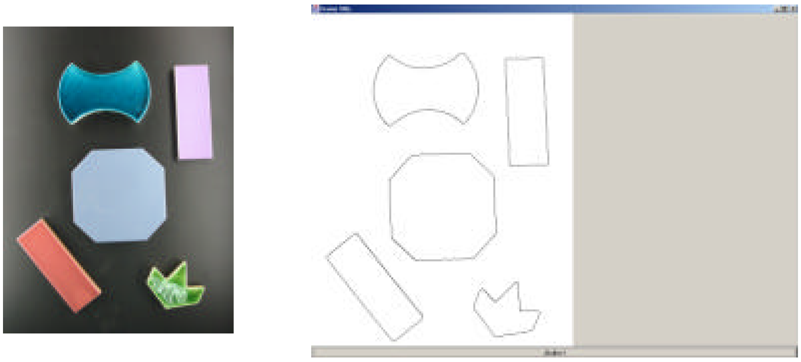
- Para la figura al Norte de la imagen, Figure1_v= [(164,87), curve, (249,113), curve, (317,81), curve, (351,167), curve, (322,229), curve, (241,200), curve, (167,233), curve, (131,152), curve];

- Para el rectángulo al este de la imagen, Figure2_v=[(487,94), straight-line, (497,315), straight-line, (418,315), straight-line, (410,94), straight-line];

- Para el octágono de la imagen, Figure3_v=[(210,297), straight-line, (327,291), straight-line, (384,351), straight-line, (389,457), straight-line, (342, 510), straight-line, (220, 513), straight-line, (170,467), straight-line, (164,347), straight-line];

- Para el rectángulo al sur-oeste de la imagen, Figure4_v=[(95,457), straight-line, (235,632), straight-line, (172, 683), straight-line, (32,506), straight-line];

- Finalmente para la figura en forma de hoja, Figure5_v=[(360,573), straight-line, (388,600), straight-line, (435,560), straight-line, (438,619), straight-line, (480,622), straight-line, (467,658), straight-line, (388,674), straight-line, (342,608), straight-line,].

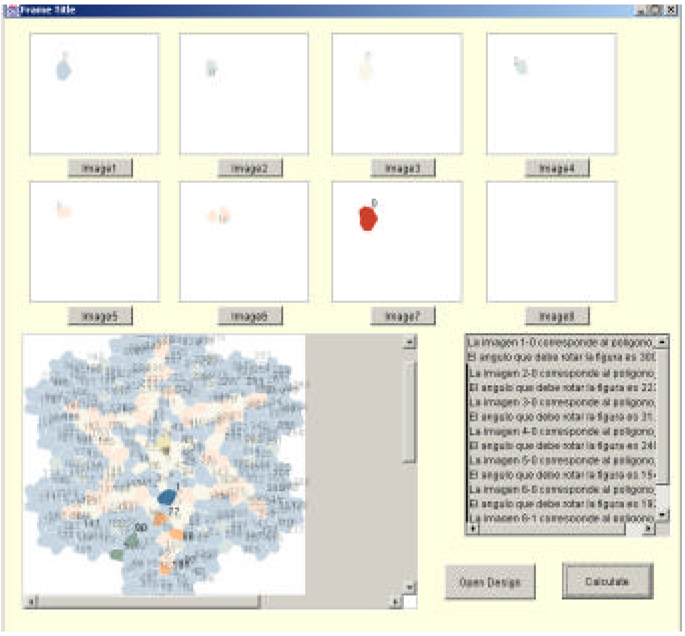


a) b)

***Figure iv).*** *a)Ejemplo de fotografía, y b) del resultado gráfico obtenido tras tratar la imagen semi-cualitativamente.*

La figura iv)b muestra el resultado obtenido al tratar la imagen. El color de cada tesela también se calcula, utilizando las coordenadas RGB del píxel posicionado en el centroide de la tesela, en el caso de que en el centroide se encuentre un agujero, se calcula las coordenadas RGB del punto más próximo al centroide fuera del agujero.

Por otro lado, la aplicación final desarrollada, recibe las imágenes, las trata y recibe por otro lado un diseño vectorial (formato .AI) del diseño de un mosaico cerámico. De este diseño también se construye la descripción cualitativa de todas sus teselas. La descripción de cada tesela en las imágenes se compara con la descripción de las teselas del mosaico y se determina si la tesela de la imagen aparece en el mosaico o no, y en caso de aparecer no solo se determina su posición sino también el ángulo que se debe rotar la tesela de la imagen (que esta situada sobre una cinta transportadora) para que en un futuro un brazo manipulador tomando la tesela por su centroide la coloque en la posición y orientación dada por el mosaico. La siguiente figura muestra una imagen de la aplicación desarrollada [Museros and Escrig 04b].



La imagen 1-0 corresponde al poligono
El angulo que debe rotar la figura es 30
La imagen 2-0 corresponde al poligono
El angulo que debe rotar la figura es 22
La imagen 3-0 corresponde al poligono
El angulo que debe rotar la figura es 31
La imagen 4-0 corresponde al poligono
El angulo que debe rotar la figura es 24
La imagen 5-0 corresponde al poligono
El angulo que debe rotar la figura es 15
La imagen 6-0 corresponde al poligono
El angulo que debe rotar la figura es 18
La imagen 6-1 corresponde al poligono

***Figura v).*** *Imagen de la aplicación para el reconocimiento de teselas frente a un mosaico.*

# ACRONYMS TABLE

| | |
|---|---|
| **4IM** | 4-Intersection Method |
| **9IM** | 9-Intersection Method |
| **AI** | Artificial Intelligence |
| **BSIP** | Basic Step of the Inference Process |
| **CAD** | Computer Aided Design |
| **CBM** | Calculus Based Method |
| **CSP** | Constraint Satisfaction Problem |
| **CST** | Constraint Satisfaction Techniques |
| **CLP** | Constraint Logic Programming |
| **CLP(FD)** | Constraint Logic Programming over Finite Domains |
| **CHR(s)** | Constraint Handling Rule(s) |
| **CRF** | Frank's Cardinal Reference System |
| **CS(s)** | Constraint Solver(s) |
| **DEM** | Dimension Extended Method |
| **FIP** | Full Inference Process |
| **FR** | Frame of Reference |
| **GIS** | Geographic Information System |
| **JEPD** | Jointly Exhaustive and Pair wise Disjoint |
| **LRS** | Length Reference System |
| **QR** | Qualitative Reasoning |
| **QSR** | Qualitative Spatial Reasoning |
| **QTR** | Qualitative Temporal Reasoning |
| **RCC** | Region Connection Calculus |
| **RO** | Reference Object |
| **RGB** | Red Green and Blue colour space. |
| **RS(s)** | Reference System(s) |
| **SR** | Spatial Reasoning |
| **wrt** | with respect to |

# ACKNOWLEDGMENTS

I am grateful for the support and encouragement of many people, in particular, my supervisors, Tere Escrig and Christian Freksa.Many of the key ideas in this work have come up in conversations with them. Thanks to Tere for her guidance and suggestions in the development of the work and for her guidance to reach a didactic way of explaining the approach. For the initial insights and suggestions during the earlier stages of this work and my career as a researcher, thanks Christian. Without the help and advice of Tere and Christian this work would have been impossible.

Also, I would like to thank all the people who have contributed to this work. I especially want to mention:

Amar Isli who has spent many hours discussing technical questions at the beginning of this work. Thomas Barkosky, and Reinhard Moratz, among other members of the University of Hamburg gave me valuable critical comments during my stay with them which allowed me to take my work forward. Technical discussions on different aspects of this work at several meetings with the ARCA research group (Spanish Association in Qualitative Reasoning and Applications) have improved the final result. To Zoe Falomir, Beatriz Delgado, Vicente Lázaro, and Amalia Diéguez among other people of ALICER (Ceramic Design Technology Institute) who gave me value critical comments and advice on the thesis proposal which helped me in the development of this work, and who helped in the development and implementation of the applications described in this dissertation. Charles has made corrections of the English style of this text. All the members of the Engineering and Computer Science Department department of Jaume I University (Castellón) where I have been teaching and doing research. This University has always encouraged research work. To all the members of the Cognition for Robotics Research Group (C4R2) of the University Jaume I, who also gave value criticism and insights on this work. I would also like to express my appreciation for the hints provided by the anonymous reviewers who helped me to improve some parts of the work which have been published in the proceeding of conferences and magazines. This work has been partially supported by CICYT under grant number TIC 2003-07182, and by ALICER.

Finally, I wish to thank to my parents, Manolo and Ana Maria, for providing me an education, for their endless and unconditional support during so many years.

# CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# LIST OF ALGORITHMS

# CHAPTER 1

## INTRODUCTION

The present PhD thesis is centred on the development and application of a model to reason about shape and about movement in a qualitative way, which means in a way similar to the human reasoning. The interest of this study originates from the necessity to find solutions for the recognition of objects and the description and reasoning about the movement in situations with high uncertainty, as it is the case of robotic applications, where robots only have limited and vague sensorial information. In these situations the use of a Qualitative Reasoning (QR), which allows the system to handle ambiguities and errors, will be the most suitable. Therefore, since this PhD thesis is developed within the QR frame, which is a subfield of Artificial Intelligence this chapter, especially section 1.1, represents an introduction to QR as a mature subfield of the Artificial Intelligence field.

Since shape is itself a spatial feature and movement can be seen as an integration of space and time by describing it as an object whose spatial relation with its environment changes in time, then there are two subfields inside QR directly related to the work developed in this dissertation: Qualitative Temporal Reasoning and Qualitative Spatial Reasoning, which are presented in section 1.2 and 1.3 respectively.

So, the movement of an object has been considered in this PhD thesis as a shape whose topologic relation with its environment (considered as another shape) changes in time. On the other hand the shape of the objects, which is a spatial aspect in itself, has also been studied using topological concepts. The recognition of objects is important during the movement of a robot since for the accomplishment of certain tasks, the robot must be able to recognize the objects which it comes across during its trajectory, since these objects can be landmarks or reference points that provide to the robot with spatial information about its environment. Therefore this work will be centred on the study of three features of space: the shape of the objects, the topology, and the movement. Several approaches exist on the shape of objects, on topology and on movement and some of them will be studied in greater depth during chapter 2, which describes a state-of-the-art of each one.

## 1.1 INTRODUCTION TO QUALITATIVE REASONING.

From its origins until now, Artificial Intelligence (AI) has been defined differently by several authors. For instance Stuart Russell in 1995 [Russell 95] defines AI as the branch of computer science concerned with making computers that can do tasks that up to the moment only human beings can do. But this definition implies that when the computer is able to do a human task, this task is not more a task inside AI. Therefore I would prefer to define AI as intelligence exhibited by an artificial entity (usually a computer). In order to understand this definition, we should explain what we consider as *intelligence.*

***Intelligence*** has been defined in a public statement named “Mainstream Science on Intelligence”, which was signed by 52 intelligence researchers in 1994, as a very general mental capability that, among other things, involves the ability to reason, plan, solve problems, think abstractly, comprehend complex ideas, learn quickly and learn from experience. It is not merely book learning, a narrow academic skill, or being smart at test-taking. Rather, it reflects a broader and deeper capability for comprehending our surroundings—"catching on", "making sense" of things, or "figuring out" what to do.

But, since individuals differ from one to another in their ability to understand complex ideas, to adapt effectively to the environment, to learn from experience, to engage in various forms of reasoning, or to overcome obstacles by taking thought, we consider that this definition is not complete. Two individuals differ in solving problems due to emotional and personal aspects, such as creativity, personality, character or wisdom. So we consider that these aspects have to be included in the definition of intelligence. For us, intelligence is the mental capacity to reason, plan, solve problems, think abstractly, comprehend ideas and language, and learn using the individual mental capacities, as creativity, personality, character or wisdom.

Although AI has a strong science fiction connotation, it forms a vital branch of computer science, dealing with intelligent behavior, learning and adaptation in machines. Research in AI is concerned with producing machines to automate tasks requiring intelligent behavior. Examples include control, planning and scheduling, the ability to answer diagnostic and consumer questions, handwriting, speech, and facial recognition. As such, it has become a scientific discipline, focused on providing solutions to real life problems. AI systems are now in routine use in economics, medicine, engineering and the military, as well as being built into many common home computer software applications, traditional strategy games like computer chess and other video games.

Therefore, Artificial Intelligence generates different methods allowing one to use computers for some typically human activities: theorem proving, diagnoses, commercial activities, scheduling, data interpretation, oral speaking and communication, making decisions, manufacturing, designing, predicting, etc.

A human being relies on reasoning based on commonsense knowledge in everyday situations as well as in highly specialised domains. Commonsense is needed for both understanding natural language sentences and for predicting the results of certain actions in real world environment. It helps to restrict the number of considered cases in complex situations and to focus attention on crucial problems of the considered task. Modelling commonsense is vital for construction of autonomous robots, who have to reason about the results of their actions in order to avoid going anything unsafe. Commonsense reasoning relaxes the strongly mathematical formulation of physical laws in real numbers, and the quantitative obsession of classical physics. Instead, reasoning proceeds in qualitative notions so that heavy and light objects of those of very small, normal and large size may be distinguished. From this idea we can derive the definition of Qualitative Reasoning (QR) which is the reasoning related to a non-numerical description of a system, preserving all its important behavioural properties and distinctions. Qualitative Models aim to capture the fundamental aspects of a system or mechanism, while suppressing much of the detail. Methods such as abstraction and approximation are often used to build models based on qualitative rather than numerical aspects of a system.

The principal goal of Qualitative Reasoning (QR) is to represent not only our everyday commonsense knowledge about physical world, but also the underlying abstractions used by engineers and scientists when creating quantitative models. So, QR generates representations for continuous aspects of the world, such as space and time, which support reasoning with very little information. As Forbus stated [Forbus 96] typically QR has focused on scientific and engineering domains, hence its first name Qualitative Physics. It is motivated by two observations:

- People draw useful conclusions about the physical world without differential equations. In our daily lives we figure out what is happening around us and how we can affect it, working with far less data, and less precise data, than would be required to use traditional, purely quantitative methods.
- Scientists and engineers appear to use qualitative reasoning when they understand a problem, when they set up more formal methods to solve particular problems, and when they interpret the results of quantitative simulations, calculations, or measurements.

Qualitative Physics began with de Kleer’s investigation on how qualitative and quantitative knowledge interacted in solving a subset of simple textbook mechanism problems [de Kleer 77]. After roughly a decade of initial explorations, the potential for important industrial applications created a lot of high interest in the field in the mid-1980s, and the area has been growing steadily with rapid progress. Given the strong potential that has been detected for industrial applications and its importance in understanding human cognition, work in qualitative modelling is likely to remain an important area in AI.

Nowadays, QR has become more than just Qualitative Physics. Most of the work has dealt with reasoning about scalar quantities, whether for instance they denote the level of a liquid in a tank, or the amount of investment in an economical model. Using the knowledge captured by QR and appropriate reasoning methods, a computer could make predictions, diagnoses, and explain the behaviour of physical systems in a qualitative manner, even when a precise quantitative description is not available or is computationally intractable. The key to a qualitative representation is not simply that it is symbolic, and utilises discrete quantity steps, but that the distinctions made in these discretisations are relevant to the behaviour being modelled.

As with many other representation issues, there is no single, universal “best” qualitative representation. Instead there exist several choices, each with their own advantages and disadvantages for particular tasks. What all of them have in common is that they provide notations for describing and reasoning about continuous properties of the physical world.

Several qualitative representations have been developed for different aspects as quantity [Forbus 84; Abbot et al. 87; Paritosh 03; Paritosh 04], mathematical relationships [Forbus 84; Williams 91; Trave-Massuyes et al. 03], ontology [Alvarez-Bravo et al. 04; Bessa Machado and Bredeweg 03], causality [Kuipers 84; Bochman 03; Ferguson et al. 00; Forbus, Ferguson and Usher 01], space, and time. Represent and reason about space and time are aspects directly related with the work developed in this PhD thesis, therefore the concepts of space, time and reasoning are dealt in sections 1.1.1, 1.1.2 and 1.1.3 respectively. Sections 1.1.4, and 1.1.5 present two more concepts involved with the representation and reasoning about time and space, which are the concept of granularity and the concept of conceptual neighbourhood.

### 1.1.1 Time

Time has long been a major subject of philosophy, art, poetry, and science. There are widely divergent views about its meaning; hence it is difficult to provide an uncontroversial definition of time.

The Oxford English Dictionary defines it as "the indefinite continued progress of existence and events in the past, present, and future, regarded as a whole." Another standard dictionary definition is "a nonspatial linear continuum wherein events occur in an apparently irreversible order."

The measurement of time has also occupied scientists and technologists, and was a prime motivation in astronomy. Time is also a matter of significant social importance, having economic value ("time is money") as well as personal value, due to an awareness of the limited time in each day and in our lives. Units of time have been agreed upon to quantify the duration of events (a significant occurrence point in time) and the intervals between them. Regularly recurring events and objects with apparent periodic motion have long served as standards for units of time. Examples are the apparent motion of the sun across the sky, the phases of the moon, and the swing of a pendulum.

In fact, *time* is currently one of the few fundamental quantities (quantities which cannot be defined via other quantities because there is nothing more fundamental known at present). Thus, in common with other fundamental quantities (like space and mass), time is defined via measurement, defining as the standard unit for time second, from which larger units are defined like the minute, hour, and day.

The seconds and minutes are expressed using a number consisting of two decimal digits and having modulo of 60. It is not to be confused with base-60 which refers to sexagesimal numerals. Hours are expressed using a number consisting of two decimal digits and having modulo of 24, but is commonly also expressed using the 12-hour clock.

Another form of time measurement consists of studying the past. Events in the past can be ordered in a sequence (creating a *chronology*), and be put into chronological groups (*periodization*). One of the most important systems of periodization is Geologic time, which is a system of periodizing the events that shaped the Earth and its life. Chronology, periodization, and interpretation of the past are together known as the study of *history*.

The **use of time** is an important issue in understanding human behaviour, education, and travel behaviour. But, different people may judge identical lengths of time quite differently. Time can "fly"; that is, a long period of time can seem to go by very quickly. Likewise, time can seem to "drag," as in when one performs a boring task. Time appears to go fast when sleeping, or, to put it differently, time seems not to have passed while asleep. Time also appears to pass more quickly as one gets older. For example, a day for a child seems to last longer than a day for an adult. One possible reason for this is that with increasing age, each segment of time is an increasingly smaller percentage of the person's total experience.

In explaining his *theory of relativity*, Albert Einstein is often quoted as saying that although sitting next to a pretty girl for an hour feels like a minute, placing one's hand on a hot stove for a minute feels like an hour. This is intended to introduce the listener to the concept of the interval between two events being perceived differently by different observers. Due to this fact which reveals that time is perceived differently by different people and therefore it is subjective, the information we manage about time is incomplete, uncertain, and even inconsistent. So a good way to study time is using qualitative models which deal with uncertain information in a suitable way.

### 1.1.2 Space

The nature of space has also been a prime occupation for philosophers and scientists for much of human history, and hence it is difficult to provide an uncontroversial and clear definition outside of specific defined contexts.

Space has a range of definitions:

- One view of space is that it is part of the fundamental structure of the universe, a set of dimensions in which objects are separated and located, have size and shape, and through which they can move.
- A contrasting view is that space is part of a fundamental abstract mathematical conceptual framework (together with time and number) within which we compare and quantify the distance between objects, their sizes, their shapes, and their speeds. In this view space does not refer to any kind of entity that is a "container" that objects "move through".

The **notion of space** is also one of the few fundamental quantities in physics, meaning that it cannot be defined via other quantities because there is nothing more fundamental known at present. Therefore, similar to the definition of other fundamental quantities (like time), space is defined via measurement, defining as the standard unit for space the meter.

Space is typically described as having three dimensions, and that three numbers are needed to specify the size of any object and/or its location with respect to another location. Modern physics (mainly relativistic physics) does not treat space and time as independent dimensions, but treats both as features of ***spacetime*** – a conception that challenges intuitive notions of distance and time.

In relativistic physics, *spacetime* is a model that combines 3-D space and 1-D time into a single construct called the space-time continuum (in which time plays the role of the 4th dimension). According to Euclidean space perception, our universe has three dimensions of space, and one dimension of time. By combining the two concepts into a single manifold, physicists are able to significantly simplify the form of most physical laws, as well as to describe the workings of the universe in a more uniform way.

Space-times are the arenas in which all physical events take place — for example, the motion of planets around the Sun may be described in a particular type of space-time, or the motion of light around a rotating star may be described in another type of space-time.

The **use of space** has also cognitive aspects associated. In fact, each individual has his/her cognitive space, resulting in a unique categorization of their ideas. The dimensions of this cognitive space depend on information, training and finally on a person's awareness. Subjective ideas about space will depend heavily on cultural background. Therefore, as in the case of time, the spatial information is obtained through perception and is subjective, and the information we manage about space is once more incomplete, uncertain, and even inconsistent. In other words, we only deal with partial information about space. So, the use of qualitative models to manage space concepts seems to be the most suitable in order to manage this uncertainty.

### 1.1.3 Reasoning

***Reasoning*** is defined very differently depending on the context of the understanding of reason as a form of knowledge.

The *logical definition* of reasoning is the act of using reason, to derive a conclusion from certain premises, using a given methodology. In this definition we use the term *reason* to describe the concept of reasoning; therefore we should explain what means *reason* specifically.

***Reason*** is a term used in philosophy and other human sciences to refer to the higher cognitive faculties of the human mind. It describes a type of thought or aspect of thought, especially abstract thought, and the ability to think abstractly, which is felt to be especially human.

Within idealist philosophical contexts, reasoning is the mental process which informs our imagination, perceptions, thoughts, and feelings with whatever intelligibility these appear to contain; and thus links our experience with universal meaning. The specifics of the methods of reasoning are of interest to a lot of disciplines such as philosophy, logic, psychology, and artificial intelligence.

There are two most commonly used explicit methods to reach a conclusion which are called *deductive reasoning* and *inductive reasoning*. But there are also other methods of reasoning, which are the *abductive reasoning* and *analogy*.

In *deductive reasoning,* given true premises, the conclusion must follow and it cannot be false. This type of reasoning is non-ampliative - it does not increase one's knowledge base - since the conclusion is inherent to the premises. A classical example of deductive reasoning is syllogisms for example:

- all humans are mortal,
- Socrates is a man,
- therefore, Socrates is mortal.

In *inductive reasoning,* on the other hand, when the premises are true, then the conclusion follows with some degree of probability. This method of reasoning is ampliative, as it gives more information than what was contained in the premises themselves. A classical example would be “The sun rose to the east every morning up till now, therefore the sun will rise to the east also tomorrow”.

A third method of reasoning is called *abductive reasoning,* or *inference* to the best explanation. This method is more complex in its structure and can involve both inductive and deductive arguments. The main characteristic of abduction is that it is an attempt to favor one conclusion above others by either attempting to falsify alternative explanations, or showing the likelihood of the favored conclusion given a set of more or less disputable assumptions. In fact, we can define **inference** as the act or process of drawing a conclusion based solely on what one already knows. Suppose you see rain on your window — you can infer from that, quite trivially, that the sky is grey. Looking out the window would have yielded the same fact, but through a process of perception, not inference (note however that perception itself can be viewed as an inferential process).

Finally, a fourth method of reasoning is *analogy*. *Analogy* is the cognitive process of transferring information from a particular subject (the analogue or source) to another particular subject (the target). In a narrower sense, analogy is an inference or an argument from a particular to another particular, as opposed to deduction, induction, and abduction, where at least one of the premises or the conclusion is general. The conclusion of an analogy is only plausible. Analogical reasoning is very frequent in common sense, science, philosophy and the humanities, but sometimes it is accepted only as an auxiliary method.

### 1.1.4 Granularity

**Granularity** is not a question of scale, it is a matter of the amount of information which is included in the representation. A coarse level of granularity will provide more abstracted information whereas a fine level of granularity will provide more detailed information. In a coarse level of granularity some of the fine details included in the fine level of granularity have been smoothed over or averaged out.

In qualitative representations, in which some relations are established (for example, “close” or “far” in the determination of the distance between objects), coarser or finer descriptions can be defined by cascading relations, for instance by making fewer or more comparisons of the quality involved. In absence of a basic reference unit in the qualitative treatment, granularity becomes a relative concept: coarseness of a representation depends on the context of the concepts involved. For example, in the following sentences which determine the comparative distance between spatial object, the same qualitative relation, “close”, is used with different granularity:

*“The cinema is close to the city center.”*

*“Segovia is close to Madrid.”*

The distance involved in the second sentence is much bigger than in the first one, therefore we can say that the first sentence is given at a high (or fine) level of granularity.

### 1.1.5 Conceptual Neighbourhood.

Conceptual neighborhood was first defined for Qualitative Reasoning by Freksa [Freksa 91] for temporal intervals as follows:

*"Two relations between pairs of events are conceptual neighbors if they can be directly transformed into another by continuous deformation (for instance shortening or lengthening) of the events."*

Events are related with time, but this definition has been translated into qualitative spatial concepts, as object boundaries, or regions by [Escrig and Toledo 98] as follows:

*"Two qualitative regions, A and B, are conceptual neighbors if, and only if, in a continuous translation from a position of the qualitative region A to a position of the qualitative region B, there does not exist a position belonging to another qualitative region C".*

Conceptual neighborhood is not a concept exclusively related to QR, but it is also useful in this field since conceptually neighboring relationships between events or regions have similar behavior [Freksa 91].

Conceptual neighborhood is important since it intrinsically reflects the structure of the represented world with their operations. Such representations of properties of the represented domain allow us to implement reasoning strategies which are strongly predisposed towards the operations in the represented domain. They can be viewed as procedural models of this domain. In the case of representing the spatial domain, conceptual neighborhoods contribute to the implementation of imagery processes. An imagery process means the creation (or re-creation) of any experience in the mind. From a computational point of view they have the advantage of restricting the problem of space in such a way that only operations will be considered which are feasible in the specific domain.

## 1.2 QUALITATIVE TEMPORAL REPRESENTATION AND REASONING.

Representing and manipulating knowledge about time and space is recognised as a crucially important part of commonsense reasoning, and therefore of QR. In this section we present an introduction about Qualitative Temporal Representation and Reasoning and section 1.3 presents an introduction to Qualitative Spatial Representation and Reasoning.

Temporal reasoning arises when dealing with problems involving time; the ability to represent and manage temporal knowledge is innate in humans as well as in artificial agents. This explains why temporal reasoning appears in so many areas, including planning, discourse analysis, natural language understanding, etc. In any activity that involves change, time is an essential feature.

The main goals of temporal reasoning are:

- The *formalisation* of the notion of time.
- The construction of a *computational rule-based system* to reason about time.

A typical temporal reasoning problem must be able to represent and manage problems as next information presents:

*Joseph and Frank work together. Joseph takes less than 20 minutes to get to work and Frank 15-20 minutes. Today Joseph left home between 7:05-7:10 a.m., and Frank arrived at work between 7:50 and 7:55 a.m.*

If we want to represent and then reason qualitatively about such knowledge, for instance answering queries such as *"who was the first to arrive to work?"*, then it is necessary to represent qualitatively the information presented. To represent this information we need two types of temporal objects: *points* and *intervals*.

### 1.2.1 Types of Temporal Objects.

To represent time information we can chose between using **intervals** or **points**.

***Intervals*** correspond to time periods during which events occur or proposition hold. ***Points*** represent the beginning and ending points of some events, as well as neutral points of time. For example, in our story we have two meaningful events: “Joseph went to work” and “Frank went to work” respectively. These events are associated with the intervals $J=[P_1,P_2]$ and $F=[P_3,P_4]$. The extreme points of these intervals $P_1$, $P_2$, $P_3$ and $P_4$ represent times at which Joseph and Frank left home and arrived to work. We can also introduce a neutral point $P_0$ to represent the *“beginning of the world”* in our story. One possible choice for $P_0$ is 7:00 a.m.

### 1.2.2 Origins of Qualitative Temporal Reasoning.

Based in the distinction between points and intervals to represent time, we can find in the literature two main temporal algebras. These two algebras which represent the origins of the Qualitative Temporal Reasoning (QTR) field are:

- Allen’s interval algebra [Allen 83].
- Vilain and Kautz’s point algebra [Vilain and Kautz 89].

**Allen’s interval algebra** [Allen 83] has become an increasingly popular formalism for representing and reasoning about temporal relationships between time intervals.

This algebra is based on 13 relations plus a set of operations defined over these relations which can be used when reasoning about time. The 13 relations defined by the Allen’s interval algebra correspond to the simple definite mutually exclusive relations that may exist between two intervals (Figure 1.1). The most general case allowing any arbitrary disjunction on relationships between temporal intervals that can be expressed in first-order logic is too complex to be considered for most AI applications. Therefore, Allen takes vectors of simple relations that are interpreted as the disjunction of relations. For instance, the vector ($I_1$ Before Meets Overlaps $I_2$) means that the interval $I_1$ either occurs before, meets or overlaps the interval $I_2$. This is a way to represent *uncertainty* on the interval temporal relation and allows us to express any possible relation between two intervals.

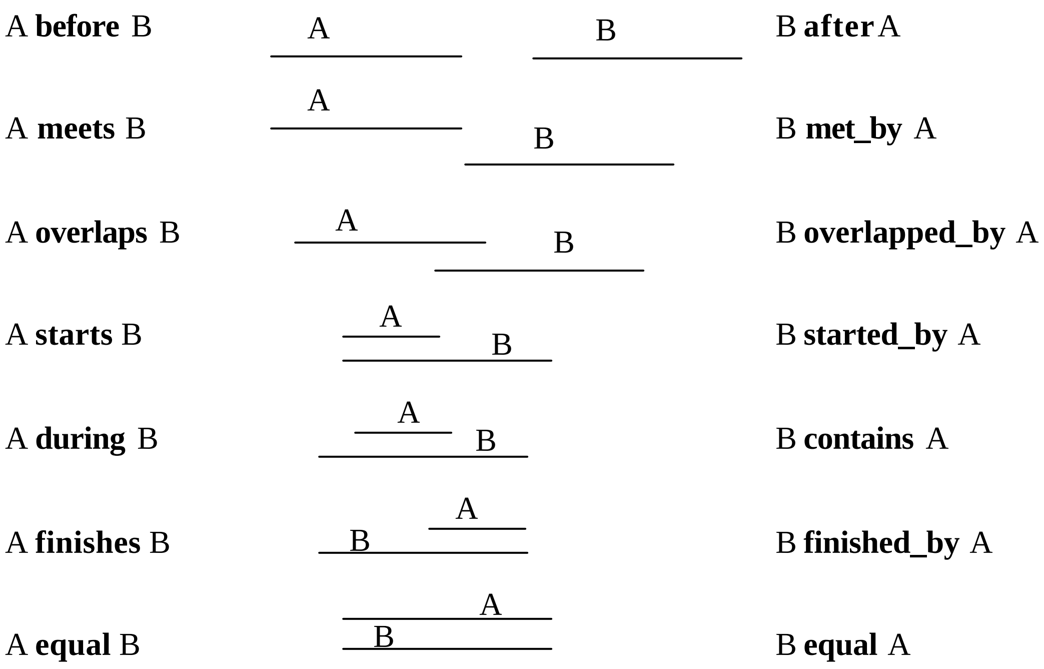


***Figure 1.1.*** *The 13 Allen's relations between temporal intervals*

In Allen's interval algebra two operations are introduced: *Addition* interpreted as the intersection of vectors (the least restrictive relation that the two vectors together admit) and *Multiplication* is the 3-elements transitivity, or composition operation: from $[I_1V_1I_2]$ and $[I_2V_2I_3]$, $V_1*V_2$ is the least restrictive relation between $I_1$ and $I_3$.

Allen also introduces an algebra which defines the relations between a point and an interval, algebra that we have called **Allen's point-interval algebra**. Figure 1.2 represents graphically the five basic temporal relations between a point, named *p*, and an interval, named *I*. The figure shows the meaning of each relation.

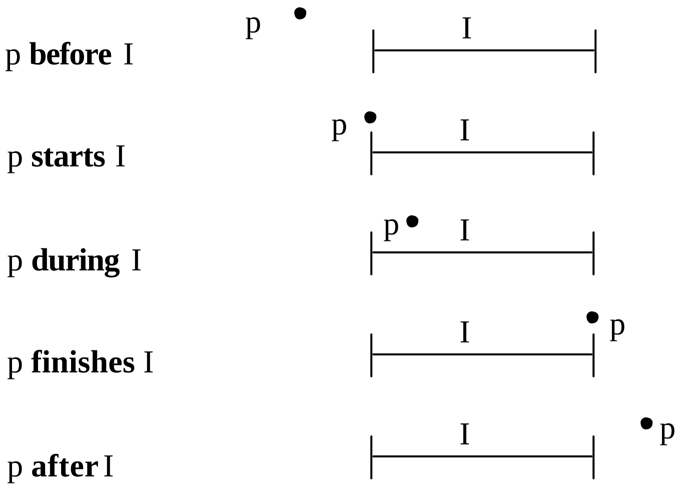


***Figure 1.2*** *The basic temporal relations between a point p and an interval I as [Allen 83] describes.*

The **Vilain and Kautz's point algebra** [Vilain and Kautz 89] defines three basic relations that can hold between two points, which are: < (which means precedes), = (which means equal), and > (which means follows). As in the interval algebra, we want to be able to represent indefinite information so they allow the relationship between two points to be a disjunction of the basic relations. Point Algebra is the algebraic structure with underlying set {<, =, =, >, =, ?, ?), unary operator converse, and binary operators intersection and composition. Note that =, for example, is an abbreviation of {<, =} and ? means there is no constraint between two points, {<, =, >}.

However, the notion of interval appears to be necessary in many cases. The idea of representing an interval as a pair of points is not new [McDermott 82], and it benefits from the computational advantages of the point algebra. But only a fragment of the *interval algebra* can be translated to the point algebra, and this fragment is called *the restricted interval algebra* [Vilain and Kautz 89]. The restricted interval algebra corresponds to the case where all constraints represent *convex* sets of intervals and, therefore, they can be expressed without the use of disjunctions between constraints on different pairs of endpoints. Figure 1.3 shows some examples. In this figure each interval is defined by its two endpoints as follows: A=[$A^-$,$A^+$] and B=[$B^-$,$B^+$].

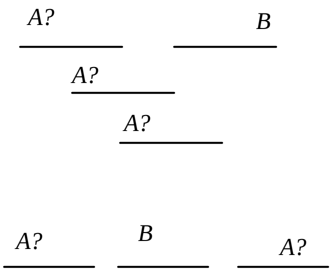
| *INTERVAL VECTOR* | *POINT TRANSLATION* | *FIGURE* |
|---|---|---|
| *A(before meets overlaps)B* | *$A^-$ precedes $B^-$* |  |
| | *$A^-$ precedes $A^+$* | |
| | *$A^+$ precedes $B^+$* | |
| | *$B^-$ precedes $B^+$* | |
| *A(before after)B* | *No equivalent point algebra* | |

***Figure 1.3***. *Translating interval algebra to point algebra examples.*

Using the Allen's point-interval algebra representation we can represent graphically the example given at the beginning of the section as figure 1.4 shows, where the point $P_0$ means the neutral point *"beginning of the world"*, the interval $J_0$ means the interval of time when Joseph left home and the point $J_1$ means the point of time in which Joseph got work, and $F_0$ is the interval in which Frank left home and the interval $F_1$ is the interval in which Frank got work.

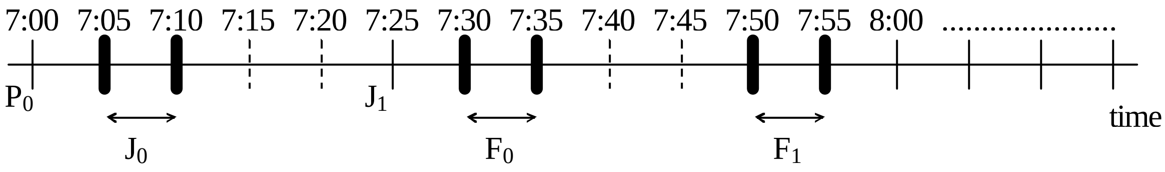


***Figure 1.4***. *Example representation following Allen's interval algebra.*

But if we want to infer new information from the one represented we need to reason about the time represented. The next section explains how reasoning is carried out in Qualitative Temporal Reasoning (QTR).

### 1.2.3 Reasoning in Qualitative Temporal Reasoning.

QTR consists of formalizing the notion of time in a qualitative way- as humans beings manage temporal information- and providing means to represent and reason about temporal aspects of knowledge. It is notable how naturally and efficiently humans are able to manage time during everyday life when interacting with the environment. The QTR community tries to understand the principles that make possible these forms of reasoning and then incorporate them into automatic reasoning systems. Moreover, the QTR community agrees that for several problems the representation of qualitative temporal relations and the reasoning about them is essential, as for natural language understanding [Allen 84; Song and Cohen 91; Beaumont, Sattar et al. 01], general planning [Allen 91; Song and Cohen 96; Clement and Durfee 99], diagnosis of technical systems [Dressler and Struss 03], and knowledge representation [Weida and Litman 92].

Following with the most renamed work inside the Qualitative Representation and Reasoning field, Allen's work [Allen 83], Allen not only introduces an interval algebra of binary relation on intervals, for representing qualitative temporal information, but he also addresses the problem of reasoning about such information, with his *Allen's* interval algebra.

For reasoning, Allen gives a polynomial-time constraint propagation algorithm for computing the closure of a set of statements in the interval algebra. Nevertheless the algorithm is sound, in the sense that it never infers an invalid consequence of a set of statements, the algorithm is not complete, in the sense we can find an example in which the algorithm does not make all the inferences that follow from a set of statements. Completeness does not always have an affordable computational cost. Vilain and Kautz [Vilain and Kautz 89] prove that constraint consistency of statements (or determining their closure) in the *interval algebra* is *NP-hard*. They suggest several strategies to work in practical systems:

1. To limit the number of statements.
2. To accept the incompleteness of the polynomial algorithm (it can be acceptable for applications that do not require much inference from the temporal reasoner).
3. Moving to a less expressive formalism, such as the point algebra.

An additional problem of the Interval Algebra is that the less we know about relationships between two intervals, the longer the symbolic representation of that knowledge. Therefore, alternative sets of relations have been proposed for representing uncertainty between intervals. Matuszec et al. [Matuszec et al. 88] approach is based on the partial information about endpoints of the intervals, for instance X sbs Y means that X starts before starts Y. Freksa [Freksa 92] generalizes this approach with the concept of semi-intervals. It is based on the concept of neighbourhood. The relation R is neighbour of another region S iff R can be transformed to S by a process of gradual, continuous change which does not involve passage through any third relation.

Figure 1.5 shows how Allen's relations are arranged according to it, following next notation: *before* (<) ,*after* (>), *during* (**d**), *contains*(**di**), *overlaps* (**o**), *overlapped-by* (**oi**), *meets* (**m**), *met-by* (**mi**), *starts* (**s**), *started-by* (**si**), *finishes* (**f**), *finishes-by* (**fi**), and *equal* (=). Freksa develops optimised transition tables for his neighbourhood primitives to perform coarse reasoning and the required computational effort decreases when knowledge is coarser.

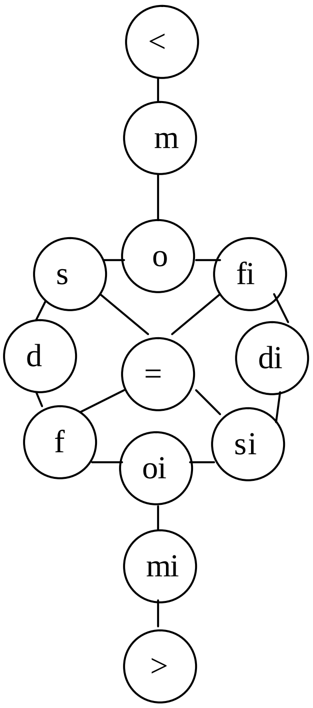


***Figure 1.5.*** *Allen's Interval temporal relations arranged according to Freksa's conceptual neighbourhood [Freksa 92].*

## 1.3 QUALITATIVE SPATIAL REPRESENTATION AND REASONING.

Space, which is a multidimensional and not adequately represented by single scalar quantities, has become a significant research area within the field of QR, generating the field called Qualitative Spatial Reasoning (QSR).

The challenge of QSR is to provide calculi which allow a machine to represent and reason with spatial entities without resorting to the traditional quantitative techniques.

There are many different aspects to space and therefore to its representation. We have to decide, not only what kind of spatial entity we will admit but also we can consider developing different kinds of ways of describing the relationships between these kinds of spatial entities; for example we may consider just their topology, or the distance between them, their relative orientation or their shape.

As QSR is a field which has defined itself over the last few years as researchers in a variety of subject areas have recognised the extent to which they have interests in common, there are many spatial aspects which qualitative spatial representation have been investigated, such as distance [Zimmermann 93], [Jong 94], [Clementini et al. 97], [Escrig & Toledo 01], and orientation [Guesgen 89], [Jungert 92], [Mukerjee & Joe 90], [Freksa 92b, Freksa & Zimmermann 96], [Hernández 94]. One approach integrating several spatial aspects, such as orientation, cardinal directions, and distances is the work by [Escrig & Toledo 98] which introduces a model for representing and reasoning with all the spatial aspects mentioned above using Constraint Logic Programming (CLP) extended with Constraint Handling Rules (CHRs).

In all these areas, sophisticated automated reasoning about the spatial relations between physical objects or regions of space is of fundamental importance; and in all the cases of studying, this must be done without precise, quantitative information about these relations.

For instance, typically, some knowledge of the topological relationships between the entities of interest may be available, along with incomplete and imprecise information about distances, directions and relative sizes; and from this partial information, useful conclusions must be drawn. Examples of the kind of question for which qualitative spatial reasoning is required could be:

*"Identify the islands in the lake and the largest one, or which parts of a network of corridors can the robot traverse without getting crashed into them?"*

These are all examples of the kind of problem human beings solve (and sometimes fail to solve) without making precise measurements; if we are to maximise the potential of computer systems to help them, we must understand the principles that make possible these forms of reasoning. This is not to assume that computer systems will necessarily use the same methods as human beings; but the fact that people can answer such questions constitutes a form of proof that usable methods exist. The qualitative spatial reasoning community has set itself the task of finding them.

As in the case of QTR first we have to define the qualitative representation of each spatial aspect considered and then to define the reasoning process using the defined representation. There are many different aspects to space and therefore to their representation; sections 1.3.1.1 to 1.3.1.4 give an introduction of same of them as: topology, orientation, distance, and shape. As topology and shape are matters of this PhD dissertation they will be studied deeply in section 2. Most of the approaches for each one of the spatial concepts (topology, orientation, distance, and shape) describe the basic relations and a set of operations between them, which means that they have concentrated on representational aspects. Some of them, as [Escrig and Toledo 98] have dealt also with the process of reasoning about these spatial aspects. Section 1.3.2 explains briefly the most common reasoning techniques used in QSR.

## 1.3.1 Spatial Aspects

This section defines several spatial aspects which are topology, orientation, distance, and shape and gives a brief introduction of how these aspects are managed in QSR by explaining some of the most relevant works for the management of each concept inside the QSR field.

### 1.3.1.1 Topology.

Topology is one fundamental aspect of space and certainly one that has been studied extensively within the mathematical literature.

Topology is a branch of mathematics concerned with spatial properties preserved under bicontinuous deformation (stretching without tearing or gluing); these are the topological invariants (properties of a topological space which is invariant under homeomorphisms).

Since topology can only make qualitative distinctions, it is clear that topology must form a fundamental aspect of QSR. So the question arises: can one simply import a traditional mathematical topological theory wholesale into a qualitative spatial representation? Although various qualitative spatial theories have been influenced by mathematical topology, there are a number of reasons why such a wholesale importation seems undesirable in general [Gotts et al. 96]; not only does traditional topology deal with much more abstract spaces than the ones to be found in the kinds of applications for QSR, but also we are interested in qualitative spatial *reasoning* not just representation, and this has been paid little attention in mathematics and indeed since typical formulations involve higher order logic, no reasonable computational mechanism would seem to be immediately obvious.

One exception to the disregard of earlier topological theories by the QSR community is the work of Clarke [Clarke 81], that has built an axiomatic theory of space which are predominantly topological in nature, and which is based on taking regions rather than points as primitive. The work of Clarke has lead to the development of so called RCC systems [Gotts et al. 96; Bennett 94; Cui, Cohn and Randell 92; Randell and Cohn 92, Randell, Cui and Cohn 92; Randell and Cohn 89]. As RCC systems are studied deeply in section 2.1, bellow we only give some basic notions about them.

Clarke took as a primitive notion the idea of two regions $x$ and $y$ being connected (sharing a point if we think on regions as sets of points): $C(x,y)$. But in the RCC systems this interpretation is changed to the closures of the regions sharing a point, and this has the effect to collapsing the distinction between region, its closure and its interior, which, it is argued, has no relevance for the kinds of domain with which QSR is concerned, and it is another reason for abandoning traditional mathematical topology. This primitive is surprisingly powerful: it is possible to define many predicates and functions, capture interesting and useful topological distinctions and therefore relations (see section 2 for details).

#### 1.3.1.2 Orientation

If we want to specify the orientation of an *object O*, with respect to a *reference object RO*, then we need some kind of *frame of reference FR*. We can define three types of frame of reference:

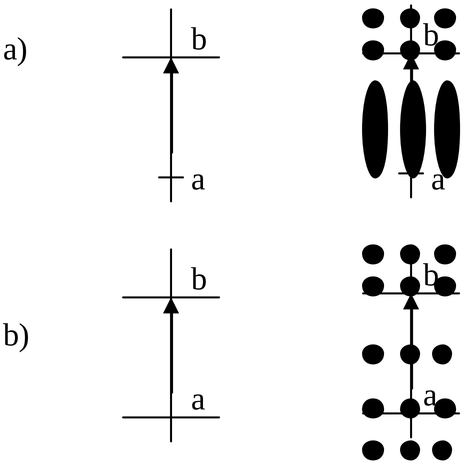
- An *extrinsic* frame of reference imposes an external, immutable orientation: a fixed coordinate system, or a third object (such as the North Pole, which represents an example of cardinal directions).
- A *deictic* frame of reference is with respect to the "speaker" or some other internal observer.
- An *intrinsic* frame of reference exploits some inherent property of the object O- many objects have a natural "front", for instance humans or buildings.

This classification can be used to classify the set of qualitative orientation models found in the literature.

Some approaches presuppose and extrinsic frame of reference, for example using cardinal directions [Frank 92; Hernández 94].

Of those with deictic triadic (using three elements to give an orientation) relations it is especially worth mentioning the work of Schlieder [Schlieder 93] who develops a spatial representation, called *panorama,* which describes the position of an object with respect to some reference points. The *panorama* is defined with reference points or *landmarks* $P_1$, $P_2$, …, $P_n$ and a point S (position of the internal observer which can be for instance a robot) as the clockwise order of the lines (or vectors) $SP_i$ and $P_iS$. With this representation the work presents restrictions of the way left(S, $P_iP_j$) which means that the robot S is at the left of the line between $P_iP_j$.

Another important deictic triadic orientation calculus is that of [Freksa 92b], which represents the orientation of an object, *c*, with respect to the Reference System (RS) defined by two points, *a* and *b*. The vector from a to b and the perpendicular line by b define the coarse RS (figure 1.6a) which divides the space into 9 qualitative regions (straight-front (sf), right-front (rf), right (r), right-back-coarse (rbc), straight-back-coarse (sbc), left-back-coarse (lbc), left (l), left-front (lf), identical-front (idf)). The vector from a to b and the two perpendicular lines between a and b define the fine RS (figure 1.6b) which divides the space into 15 qualitative regions(straight-front (sf), right-front (rf), right (r), right-middle (rm), identical-back-right (ibr), back-right (br), straight-back (sb), identical-back (ib), straight-middle (sm), identical-front (idf), left-front (lf), left (l), left-middle (lm), identical-back-left(ibl), back-left (bl)).



| lf | sf | rf |
|---|---|---|
| l | idf | r |
| lbc | sbc | rbc |

| lf | sf | rf |
|---|---|---|
| l | idf | rb |
| lm | sm | rm |
| ibl | ib | ibr |
| bl | sb | br |

***Figure 1.6.*** *a) Freksa's coarse orientation RS and b) Freksa's fine orientation RS.*

### 1.3.1.3 Distance

Distance is a way of describing how far apart two things lie. In physics or everyday discussion, distance may be a physical length, a period of time, or an estimation based on other criteria (e.g. "two countries over").

In day-to-day discussion, distance often refers to the length of a straight line between objects. Distance is sometimes expressed in terms of the time to cover it, for example walking or by car. Sometimes, these informal treatments do not meet the criterion for a metric. For example, if one measures distance by car and there are one-way streets, then that distance probably will not be symmetric. Measuring distance by time might also not be symmetric, as a road may be more crowded in one direction than in the other, for instance. Even in a given direction, though, time might not give a distinct distance.

Downtown might be an hour from home in good traffic and five hours in bad, and it can always increase by going slower. Therefore measurement of distance and its representation is a matter of study in the QSR field.

The spatial representations of distance can be divided into two main groups:

- those which measure on some “absolute” scale, called naming distances,
- those which provide some kind of relative measurement, called comparing distances.

For comparing distances, usually people uses the set of predicates <, =, >, which characterize the result of direct comparison.

With respect to naming distances, the types of objects involved and the context in which they are embedded are decisive factors for establishing the set of relations to be used, which means that naming distances depend on *granularity.*

Taking into account granularity, the first level of granularity that comes to mind distinguishes between *close* and *far*. Those two relations can subdivide the plane into two regions centred on the reference object, where the outer region goes to infinity. But, cognitive considerations suggest the need for systems of distance relations organized along various levels of granularity, for example with three distinctions *close, medium,* and *far;* or a level with four distinctions *very close, close, far* and *very far*, and so on (see figure 1.7).

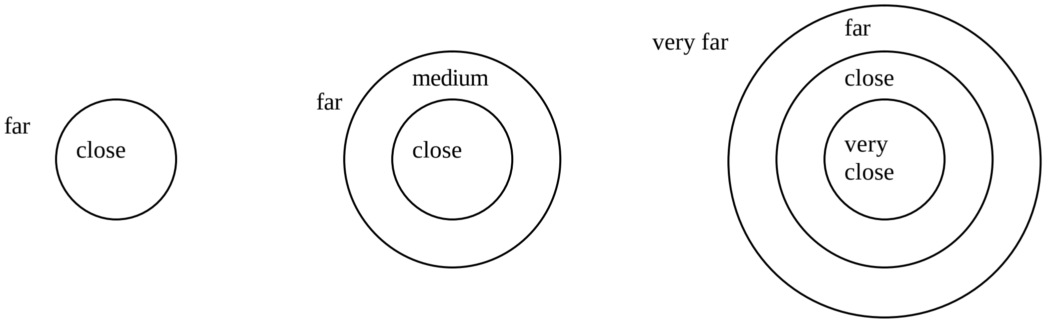


***Figure 1.7.*** *Various levels of distance distinctions (granularity)*

In order to understand better the way how distance is managed in QR here we present briefly how two models implement it. One of the most renamed qualitative named distance calculus is the framework for representing distances [Hernández, Clementini and Di Felice 95]. This qualitative framework needs three elements to establish a distance relation: the *primary object (PO)*, the *reference object (RO)*, and the *frame of reference (FR)*. The distance between the reference object A and the primary object B is expressed by $d_{AB}=d(A,B)$. In general, at a given granularity level, space surrounding a reference object is partitioned according to a number of totally ordered distance distinctions $Q=\{q_0, q_1, q_2, \ldots, q_n\}$, where $q_0$ is the distance closed to the reference object and $q_n$ is the one farthest away. Distance relations are organised in *distance systems (D)* consisting of:

- a list of *distance relations* (the set of qualitative distinctions being made and their increasing distance order), which is based on the qualitative distance symbols in increasing order:

$$Q=\{q_0, q_1,\ldots, q_n\}$$

- a set of *structure relations* describing how the distance relations in turn relate to each other (order-of-magnitude relations between the various named distance ranges). The *structure relations* are defined by intervals which define acceptance areas for each symbol ($\delta_0$ corresponds to the acceptance area of symbol $q_0$ and so on).

In order to describe distance relations it distinguishes between $\delta_i$ being the "distance range *i*" (acceptance areas), and $\Delta_i$ being the "distance range from the origin and including the distance range $\delta_i$" (figure 1.8). The distance symbol $q_i$ labels all distances starting from the origin and falling in the range $\Delta_i$.

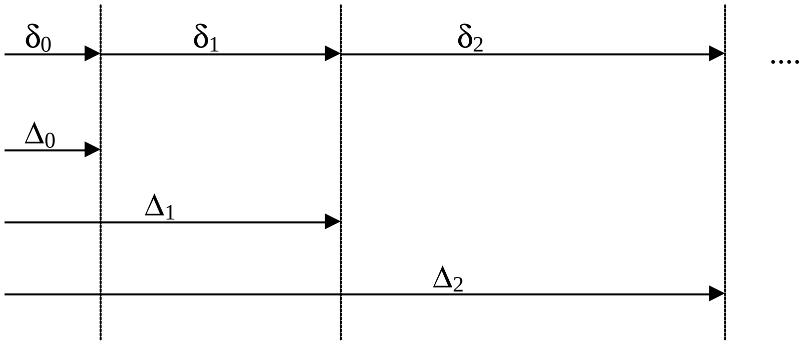


***Figure 1.8.*** *Distance ranges versus distance from origin.*

Based in this idea, [Escrig and Toledo 98] develop a representation of naming distances in which the acceptance areas overlap (figure 1.8). By cognitive considerations the acceptance areas in [Escrig and Toledo 98] have been chosen in increasing length, and due to the imprecision of human perception, this work argues that it is more difficult to reflect reality with sharply separated regions. In fact, people do not worry about the exact point or line which divides the *"close"* area from the *"far"* area. Instead of that, subjective considerations related to the culture or experience provides the distinction between regions. Thus overlapped acceptance areas seem to serve as a better model of human perception.

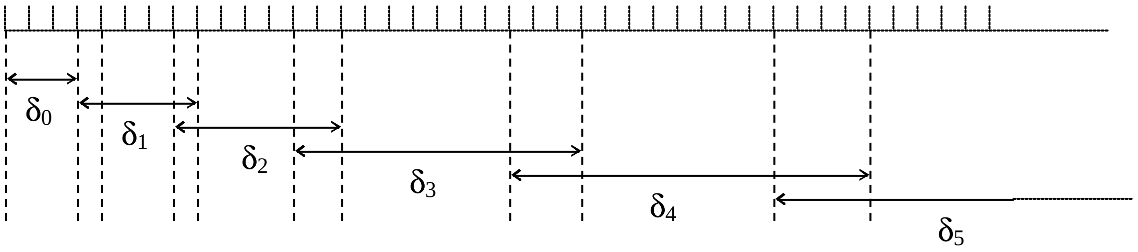


***Figure 1.9.*** *[Escrig and Toledo 98] structure relations whose acceptance areas overlap.*

Finally it is important to remark that distance is closely related to the notion of orientation: for instance distances cannot usually be added unless they are in the same direction, or depending on the orientation. [Escrig and Toledo 98] combines concepts of distance and orientation for adding distances of the same orientation, of opposite orientation and of any orientation.

#### 1.3.1.4 Shape

In geometry, two sets have the same shape if one can be transformed to another by a combination of translations, rotations and uniform scaling. In other words, the shape of a set is all the geometrical information that is invariant to location, scale and rotation. Shapes of physical objects are equal if the subsets of space that these objects occupy satisfy the definition given above.

Shape can also have a looser definition as the appearance of something, especially its outline. Such a definition agrees with the above, in that the shape does not depend on its position, size or orientation. However it does not always imply an exact mathematical transformation. For example it is common to talk of star-shaped objects even though the number of points of the star is undefined. Typically the shape of an object can be characterized by basic geometry such as points, line, curves, plane, and so on.

Objects which are geometrically similar either have the same shape or one has the same shape as the other's mirror image (or both if they are themselves symmetric).

One can think of theories of shape forming a hierarchy ordered expressiveness (in terms of the spatial distinctions made possible) with topology at the top and a fully metric/geometric theory at the bottom. Clearly in a purely topological theory of shape very limited statements can be made about the shape of a region: whether it has holes, or interior voids, or whether it is in one piece or not. However, if one’s application demands finer grained distinctions than these, then some kind of semi-metric information has to be introduced. There is a huge choice extending topology with some kind of shape primitives whilst still retaining a qualitative representation (which means not becoming fully metric). Depending on this extension we can found different types of approaches which will be studied in section 2.

### 1.3.2 Reasoning in Qualitative Spatial Reasoning.

For reasoning in QSR various computational paradigms have been investigated including constraint based reasoning [Hernández 94]. As [Cohn and Hazarika 01] stated, most of the reasoning techniques studied in QSR are based on the composition table ([Egenhofer and Frenzosa 91], [Cohn 94], [Randell, Cui and Cohn 92], [Röhrig 97], [Schlieder 95], [Escrig and Toledo 98]), which allows a compositional inference.

A compositional inference is a deduction, from two relational facts of the form $R_1(a,b)$ and $R_2(a,b)$, of a relational fact of the form $R_3(a,c)$, involving only $a$ and $c$. [Cohn and Hazarika 01] explains that the validity of compositional inferences does not depend in many cases on the constants involved but only on the logical properties of the relations. In such a case the composition of pairs of relations can be maintained for table look up when required. This technique is of particular significance when we are dealing with relational information involving a fixed set of relations. Given a set of $n$ relations which are Jointly Exhaustive and Pair wise Disjoint (JEPD), one can store in a $nxn$ composition table the relationships between $x$ and $z$ for a pair of relations $R_1(x,y)$ and $R_2(y,z)$. In general each entry of the table will be a disjunction because of the qualitative nature of the calculus.

## 1.4 BASES FOR THE INTEGRATION OF SEVERAL SPATIAL ASPECTS.

One objective of this dissertation is to integrate the models here developed with other spatial aspects, specifically we want to integrate the topological and movement models with the work by [Escrig and Toledo 98] which accomplishes the integration of several spatial aspects by considering them as instances of the Constraint Satisfaction Problem (CSP). This integration has been inspired by the temporal reasoning field, where the integration of point algebra, interval algebra and metric information has been successfully accomplished [Meiri 91]. In order to accomplish the integration of different spatial aspects (orientation, cardinal directions, and absolute and relative distances) in the same model, the next three steps are defined:

- The representation of the spatial aspect to be integrated, which is seen as an instance of the Constraint Satisfaction Problem (CSP).

- The definition of the Basic Step of the Inference Process (BSIP). It is defined such as: given the spatial relationship between objects A and B, and the spatial relationship between objects B and C, the BSIP consists of obtaining the spatial relationship between A and C. This means that for each spatial aspect to be integrated to have to define its composition tables.

- the definition of the Full Inference Process (FIP), that consists of repeating the BSIP as many times as possible, with the initial information and the information provided by previous steps of the BSIP, until no more information can be inferred. For computing the FIP it is considered as a CSP problem.

A CSP for binary constraints can be formulated such that: given a set of variables *{$X_1$,….,$X_n$}*, a discrete and finite domain for each variable *{$D_1$,…, $D_n$}*, and a set of constraints {$c_{ij}(X_i,X_j)$}, which define the relationship between every couple of variables $X_i$, $Xj$ (1≤ i < j ≤ n); the problem is to find an assignment of values <$v_1$,…, $v_n$>, $v_i$ Î $D_i$ to variables so that all constraints are satisfied, i.e. $c_{ij}(X_i,X_j)$ is true for every *i,j* (1≤ i < j ≤ n). Every different assignment of values that satisfies all the constraints is called a *solution.*

The best solution known to the CSP problem doesn't have polynomial temporal complexity; however, there exist algorithms which approximate the solution. These algorithms approximate the complete propagation process by local constraint propagation, as path consistency. If the constraint graph is complete (that is, there is a pair of arcs, one in each direction, between every pair of nodes) it suffices to repeatedly compute paths of two steps in length at most. This means that for each group of three nodes (i,k,j) we repeatedly compute the following operation until a fixed point is reached [Fruehwirth 94]:

$$c_{ij} := c_{ij} \oplus c_{ik} \otimes c_{kj} \qquad (1)$$

This operation computes the composition of constraints (⊗) between nodes *ik* and *kj*, and the intersection (⊕) of the result with constraints between nodes ij. The complexity of this algorithm is O($n^3$) , where n is the number of nodes in the constraint graph (that is, the number of objects involved in the reasoning process) [Kumar 92].

Constraint Handling Rules (CHRs) are a tool which helps to write the above algorithm. They are an extension of the Constraint Logic Programming (CLP) which facilitates the definition of constraint theories and algorithms which solve them. They facilitate the prototyping, extensions, specialization and combination of constraints. There exist mainly two types of CHRs: *propagation and simplification*. Propagation CHRs add new constraints which are logically redundant but may cause further simplification. A *propagation CHR* is of the form:

$$H_1,\cdots,H_i \Longrightarrow G_1,\cdots,G_j \mid B_1,\cdots,B_k \qquad (i > 0, j \geq 0, k \geq 0) \qquad (2)$$

The propagation from user-defined constraints, H', means the addition of the set of constraints B to the initial set of constraints if H' matches the head (H) of a propagation rule and the guard G is satisfied. This kind of rule is used to compute the part '⊗' of formula (1).

*Simplification CHRs* replace constraints by simpler constraints preserving logical equivalence. A simplification CHRs is of the form:

$$H_1,\cdots,H_i \Longleftrightarrow G_1,\cdots,G_j \mid B_1,\cdots,B_k \qquad (i > 0, j \geq 0, k \geq 0) \qquad (3)$$

The multi-head ($H_1$, …,$H_i$) is a conjunction of user-defined constraints and the guard ($G_1$, …,$G_i$) is a conjunction of literals. To simplify the user-defined constraints H' means to replace them by B if H' matches the head (H) of a simplification rule and the guard G is satisfied. This kind of rule is used to compute the part '⊕' of formula (1).

In [Escrig and Toledo 98] CSP is used to compute the FIP by rewriting the formula (1) for each spatial aspect to be integrated.

Finally, in [Escrig and Toledo 02] the concept of velocity has been also integrated in the same model by following the three steps mentioned above and considering the problem as an instance of the CSP.
Chapter 3 and 4 presents how these three steps have been applied in order to get the integration of topology and motion in the same model.

## 1.5 OBJECTIVES OF THIS PHD THESIS.

This section presents the main objectives of our work together with the structure of the rest of the PhD thesis.

The first objective of our work is to make a comparative study of the existing approaches in QSR related with three spatial aspects: topology, movement and shape (chapter 2). These allow us to know the limitation of each approach in order to go a step further in the research work in QSR.

From this study, our first new contribution arises, presenting a motion model as a qualitative representational model for integrating qualitatively time and topological information for reasoning about dynamic worlds in which spatial relations between regions and between regions and objects may change with time. Therefore we have first developed a topological algebra, presented in chapter 3 (section 3.2 to 3.4), which has been integrated to the time algebra developed and presented in chapter 4 (section 4.1) in order to obtain a motion model, which is described in chapter 4, sections 4.2 to 4.4.

The topological algebra has been developed in a way which allows its integration with other spatial aspects such as orientation, distance, cardinal directions, velocity and so on. Moreover the integration of topology and time in order to obtain a motion model is also accomplished. The bases for these integrations have been inspired by the integration of several spatial aspects in the spatial reasoning field [Escrig and Toledo 98], bases that have been explained in section 1.4.

The second contribution of this thesis is the qualitative shape representation theory for non-regular polygonal object with or without holes and curves, and for completely curvilinear objects. This theory is described in detail in chapter 5.

Chapter 6 describes the applications of both contributions. First the motion model together with the simplest version of the qualitative shape description theory defined for non-polygonal closed objects have been applied to robotics (section 6.1) and then the complete qualitative shape description theory has been applied to industry (section 6.2).

Finally conclusions and ideas for future work are given in chapter 7.

# CHAPTER 2

## RELATED STATE OF THE ART

This chapter presents three studies of the state of the art: topological models, movement models and qualitative shape representations, which are the spatial aspects dealt in this PhD Thesis. They do not pretend to be an exhaustive overview of each one of them. What is intended in this chapter, is to introduce those approaches which have influenced our work, sometimes in a critical a comparative way.

We have taken advantage of some research work which appears in the literature as it will be explained in the corresponding chapters. Moreover, the deficiencies of these approaches for our goal have motivated the work presented in this thesis. What is already done in the field and those deficiencies are summarised in the conclusions of this chapter.

### 2.1. STATE OF THE ART ON TOPOLOGY

As we are interested in model motion as a combination of topological aspects and temporal ones, the first matter of study are the topological relations.

Topology is one fundamental aspect of space. Topology, which is closely related to geometry, is concerned only with those properties of geometric objects (such as number of holes, dimensionality, and boundaries) that remain unchanged when the objects are distorted in any way by such things as twisting, shrinking, or stretching. Because of this, topology is popularly known as *rubber sheet geometry*. For instance, if we used rubber sheets to write and draw on the picture in figure 2.1a, and if someone were to grab the right and left hand sides of this picture and pull (figure 2.1 b) then the image distorts from its original state. The same happens when someone were to squash the picture (figure 2.1c). And although the three images (figure 21.a to 2.1c) are different in many ways, the fact is that they are still a fish. This happens because there are some properties or relations–named topological relations- that always remain invariant under topological transformations, such as rotation, translation and resizing. It is important to remark that cutting (tearing or breaking), gluing together, or inserting, removing or merging holes of the objects are not topological transformations.

It is also interesting to differentiate the properties that are **always** preserved by the topological transformations such as connectivity, separation, intersection, order, and dimension, from the properties that are **not always** preserved by topological transformations, such as length, area, direction or shape.

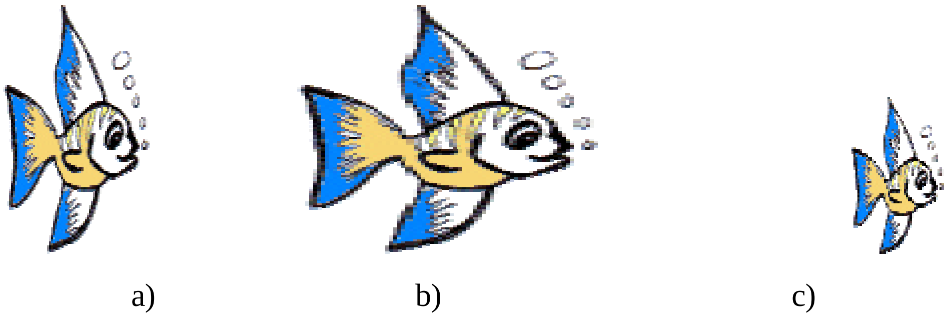


***Figure 2.1.*** *a) Original Image of a fish, b) Enlarged Image of a fish, c) Squeezed image of a fish. However topologically all of them are equal: a fish.*

Therefore, it is clear that topology must form a fundamental aspect of QSR since topology can only make qualitative distinctions. Although topology has been studied extensively within the mathematical literature, much of it is too abstract to be of relevance to those attempting to formalise common-sense spatial reasoning, then to QSR. Moreover, QSR is related to reasoning and not just representation, and this part has been paid little attention in mathematics. Of course, it might be possible to adapt the conventional mathematical formalisms, and indeed this strategy has been adopted [Egenghofer and Franzosa 91, 95].

Possibly topology contains the greatest body of research in spatial reasoning. It covers the work of [Egenhofer 91], [Ligozat 93], [Cohn et al. 93], [Hernández 94], [Mukerjee and Mittal 95], and others.

Therefore a great variety of approaches related to topological relations have been proposed. In these approaches we can find two principal trends:

1. Approaches studying the topological relations using as basic entity the spatial region, which are called *Spatial Region Approaches.*

2. Approaches studying the topological relations in terms of operations between sets of points, which are called *Point-Set Approaches.*

### 2.1.1. Spatial Region Approaches.

In these approaches a spatial region is a non-empty region, therefore points, lines and boundaries cannot be spatial regions ([Clarke 81],[Ciu, Cohn & Randell 92], [Gotts 96], [Bennett 94], [Renz & Nebel 98], [Asher and Vieu 95]). The basic relation that they define is C(x,y), that means x connects with y. This relation appears when the topological neighbouring regions x, y share a point. Using the basic relation, the approaches define a set of basic topological relations, which are mutually exclusive and complete. These two characteristics mean that between any two regions there is one and only one of the basic topological relations.

The primitive C(x,y) has been shown to be extremely powerful and has led to the development of a rich calculus of spatial predicates and relations, referred to as the RCC calculus. [Gotts et al. 96, Gotts 96] shows how the RCC calculus can describe and distinguish between complicated objects such as loops, and doughnuts with degenerate holes (Figure 2.2). However, this expressiveness is costly. And reasoning with the general RCC calculus is undecidable [Cohn et al. 97]. There has been some work on tractable subclasses. The best known is based on identifying a pair wise disjoint, jointly exhaustive set of eight spatial relations called RCC-8 calculus, which comes from Leeds (i.e.[Ciu et al. 93]).

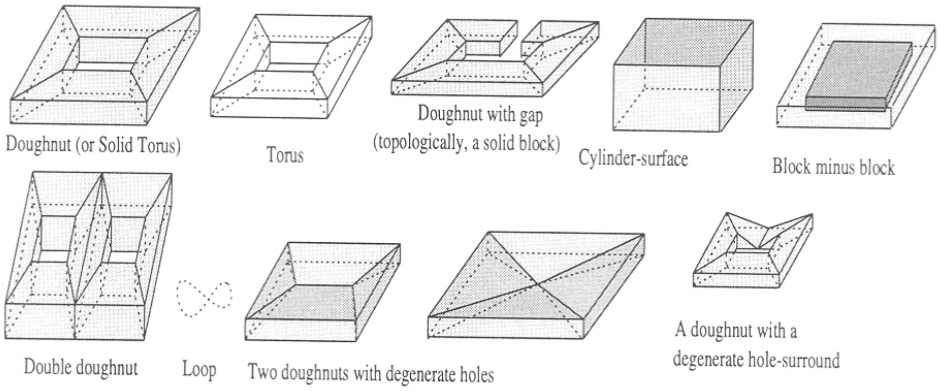


***Figure 2.2.*** *It is possible to distinguish these shapes using C(x,y) alone.*

[Asher and Vieu 95] is also an example of a theory on tractable subclasses of the RCC calculus. It develops a topological theory which is shown to be consistent and complete with respect to a certain class of models, and which has at least the same expressive power as RCC8. The main difference between them is that RCC8 does not distinguish between open and closed regions, whereas Asher and Vieu's theory does. This theory is built on a unique primitive: the relation connection C(x,y), interpreted here as connection between spatio-temporal regions, that it will be called sometimes "histories". Using C(x,y) the following relations can be defined: *part of (P), disconnection (DC), proper part of(PPO), overlap(O), partial overlap(PO), contact (EC), tangential part (TP), non tangential part (NTP), non tangential proper part (NTPP).*

Asher and Vieu's theory, and the RCC-8 theory ([Ciu et al. 93]) are both based upon Clarke's calculus ([Clarke 81, Clarke 85]) of individuals based on "connection". The basic part of the theory assumes the relation: C(x,y). However Clarke's interpretation of C(x,y) is that the two regions share at least one point whereas Asher and Vieu, and RCC-8 theories interpretation is that the topological closures of the two regions share at least one point. The topological closure of a region *A* is the smallest closed region containing *A*, therefore the closure of *A* will be the intersection of all closed sets containing *A*.

RCC-8 makes an exact definition of what counts as a region. In their first interpretation [Randell and Cohn 89], the regions may be of arbitrary dimension, however finally they must all be the same dimension and must not be of mixed dimension (for example, a region with a lower dimensional spike missing or sticking out is not intended). Such regions are termed as *regular.* They consider 3D regions and 2D regions (as is usual in GIS applications). Within the RCC theory it is not possible to distinguish between regions that are open, closed or neither, because they have the same closure. Moreover, they argue that distinctions are not necessary for QSR because such regions occupy the same amount of space and, moreover, there seems to be no reason to believe that some physical objects occupy closed regions and others occupy open regions. Moreover, regions are really spatially extended, i.e. they rule out the possibility of a region being null. Other than these restrictions, they allow any kind of region, in particular the regions may be multi-piece regions, have interior holes and tunnels.

Using C(x,y) a basic set of relations is defined. These relations describe different degrees of connection between regions from being disconnected, to being externally connected, allowing partial overlap, one region being a tangential part of the other, or a non tangential part, and so on. All degrees of connection from being externally connected to sharing mutual parts and thus being identical are formally defined. To define the relations of RCC-8 ([Ciu et al. 93], [Cohn 94], [Cohn et al. 97]) first other relations are needed in terms of C(x,y) relation, these relations are described bellow:

- DC(x,y) ↔ ¬C(x,y), which means that x and y are disconnected.
- P(x,y) ↔ ∀z[C(x,z) → C(y,z)], which means that x is part of y.
- PP(x,y) ↔ P(x,y) and ¬P(y,x), which means that x is proper part of y.
- EQ(x,y) ↔ P(x,y) and P(y,x), which means that x and y are equal.
- O(x,y) ↔ ∃z [P(z,x) and P(z,y)], which means that x and y overlap.
- DR(x,y) ↔ ¬O(x,y), x and y are discrete.
- PO(x,y) ↔ O(x,y) and ¬P(x,y) and ¬P(y,x), x and y partially overlap.
- EC(x,y) ↔ C(x,y) and ¬O(x,y), x and y are externally connected.
- TPP(x,y) ↔ PP(x,y) and ∃z [EC(z,y) and EC(z,x)], x is a tangential proper part of y.
- NTPP(x,y) ↔ PP(x,y) and ¬TPP(x,y), x is a non tangential proper part of y.
- Pi(x,y) ↔ P(y,x), Pi is the converse relation of P(x,y).
- PPi(x,y) ↔ PP(y,x), PPi is the converse relation of PP(x,y).
- TPPi(x,y) ↔ TPP(y,x), TPPi is the converse relation of TPP(x,y).
- NTPPi(x,y) ↔ NTPP(y,x), NTPPi is the converse relation of NTPPi(x,y).

From the relations before defined, RCC-8 reduces, by removing redundant relations, the set of relations to use to 8 topological relations which are jointly exhaustive pair wise disjoint, they are DC, EC, PO, TPP, NTPP, EQ, TPPi, NTPPi. Figure 2.3 shows graphic examples of these relations.

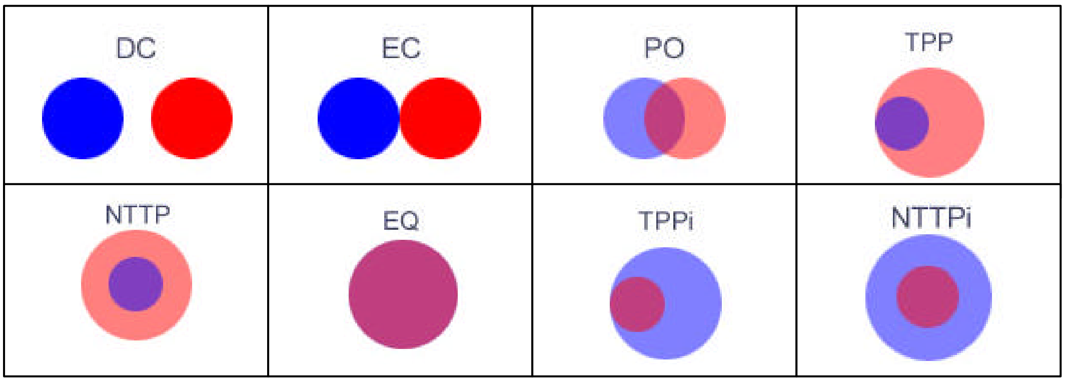


***Figure 2.3**. Graphic representation of the RCC-8 relations.*

## 2.1.2. Point-Set Approaches.

The approaches in this group are called point-set theories and consider set of points as entities, therefore points, lines and areas as the basic entities. Three sets of points are associated with every region- its interior, boundary and exterior. The topological relations are defined in terms of the intersections of the interiors and boundaries of the sets of points ([Egenhofer & Franzosa 91], [Pullar & Egenhofer 88], [Egenhofer 91], [Clementini & Di Felice 95]). Therefore relations are defined in terms of some properties of regions which are: the boundary, the interior, the exterior and the dimension of a set of points.

The *boundary* of a feature h is denoted by $\delta h$; it is defined for each of the feature types as follows:

$\delta P$: we consider the boundary of a point-like feature to be always empty.

$\delta L$: the boundary of a linear feature is the empty set in the case of a circular line, and the two distinct endpoints otherwise.

$\delta A$: the boundary of an area is the circular line consisting of all the accumulation points of the area.

The *interior* of a feature h is denoted by $h^{\circ}$. It is defined as $h^{\circ} = h - \delta h$. Note that the interior of a point feature is equal to the feature itself.

The *exterior* $h^{-}$ of a feature h is defined as: $h^{-}=\Re^{2} - h$.

The function *dim*, which returns the dimension of a feature of either of the types we consider, or of the intersection of two or more such features, is defined as follows (the symbol $\varnothing$ represents the empty set):

If S?∅ then

$$\text{dim(S)}=\begin{cases} 0 & \text{if S contains at least a point and no lines and no areas}\\ 1 & \text{if S contains at least a line and no areas}\\ 2 & \text{if S contains at least an area}\end{cases}$$

else dim(S) is undefined.

Each theory inside this group gives a set of basic topological relations that can exist between two spatial regions.

In this chapter we present some of the most relevant theories inside the point-set approaches. These approaches are presented ordered by expressiveness power, presenting first the less expressive approach and finally the more expressive one.

In [Pullar & Egenhofer 88] the **4-Intersection method (4IM)** for classifying topological relationships between one-dimensional intervals of $\Re^1$ is described. In [Egenhofer & Franzosa 91] the same method for classifying topological relationships between area features in $\Re^2$ is adopted. By considering also point and line features, we can distinguish among 6 major groups of binary relationships: area/area, line/area, point/area, line/line, point/line and point/point. In the 4IM, the classification of relationships is based on the intersection of the boundaries and interiors of two features $h_1$ and $h_2$ and b. Each intersection may be empty (∅) or non-empty (¬∅), resulting a total of $2^4$=16 combinations. Each case is represented by a matrix of values:

$$M=\begin{pmatrix} \partial h_1 \cap \partial h_2 & \partial h_1 \cap \partial h_2^{\circ} \\ h_1^{\circ} \cap \partial h_2 & h_1^{\circ} \cap h_2^{\circ} \end{pmatrix}$$

Such a matrix forms an equivalence class for topological relations, for instance the matrix $\begin{pmatrix} \neg\Phi & \Phi \\ \neg\Phi & \Phi \end{pmatrix}$, which means that the intersections $h1^{o}\cap h2^{o}$ and $\delta h1\cap h2^{o}$ are the only ones non empty, represents all the topological relations depicted in figure 2.4, which means that has a geometric interpretation. The geometric interpretations presented in figure 2.5 are topologically equivalent because the resulting 4IM matrix of them is the same.

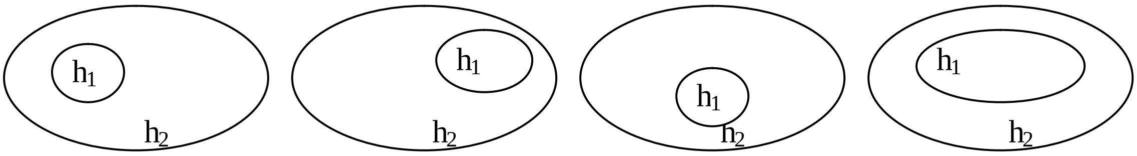


***Figure 2.4.*** *Geometrical Interpretation of the matrix* $\begin{pmatrix} \neg\Phi & \Phi \\ \neg\Phi & \Phi \end{pmatrix}$

It is possible to apply some simple geometric considerations to asses that not all combinations of values (∅ or ¬∅) of the matrix make sense for simple objects. These combinations are called the impossible cases. Removing the impossible cases, we get the valid matrices, which are shown together one possible geometrical interpretation which validates them are shown in figure 2.5.

The model also points out the converse relationships, which correspond to pairs of matrices $M_1$, $M_2$, such that $M_1=M_2^T$.

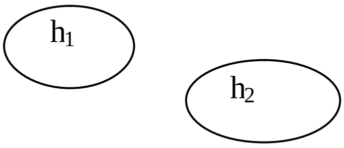
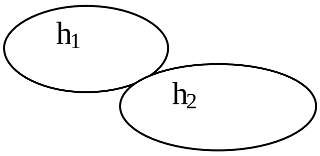
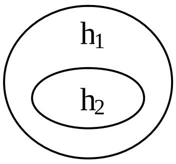
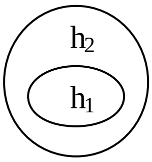
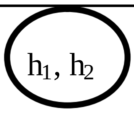
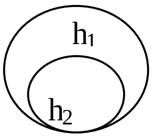
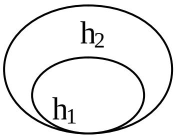
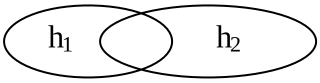
| Matrix | Graphical Representation |
|---|---|
| $\begin{matrix}\emptyset & \emptyset \\ \emptyset & \emptyset\end{matrix}$ |  |
| $\begin{matrix}\emptyset & \emptyset \\ \emptyset & \neg\emptyset\end{matrix}$ |  |
| $\begin{matrix}\neg\emptyset & \neg\emptyset \\ \emptyset & \emptyset\end{matrix}$ |  |
| $\begin{matrix}\neg\emptyset & \emptyset \\ \neg\emptyset & \emptyset\end{matrix}$ |  |
| $\begin{matrix}\neg\emptyset & \emptyset \\ \emptyset & \neg\emptyset\end{matrix}$ |  |
| $\begin{matrix}\neg\emptyset & \neg\emptyset \\ \emptyset & \neg\emptyset\end{matrix}$ |  |
| $\begin{matrix}\neg\emptyset & \emptyset \\ \neg\emptyset & \neg\emptyset\end{matrix}$ |  |
| $\begin{matrix}\neg\emptyset & \neg\emptyset \\ \neg\emptyset & \neg\emptyset\end{matrix}$ |  |

***Figure 2.5.*** *Validity of the 4-Intersection Matrices.*

By not considering the impossible cases and considering just one case for each pair of converse relationships, the model arrives at the result shown in Table 1, where there are in total 37 distinct and mutually exclusive relationships between features. In detail, in the area/area group, as there are 8 impossible cases and 2 pairs of converse relationships, the number of different types of relationships is 6. Line/area cases are 11 because there are 5 impossible cases; line/line cases are 12 because there are 14 pairs of converse relationships. The possible cases are only 3 for the point/area and point/line groups and 2 for the point/point group.

| relationships groups | Num. of possible cases | No. of real cases |
|---|---|---|
| area/area | 8 | 6 |
| line/area | 11 | 11 |
| point/area | 3 | 3 |
| line/line | 16 | 12 |
| point/line | 3 | 3 |
| point/point | 2 | 2 |
|  |  | Total: 37 |

***Table 2.1**. A summary of the 4IM. The number of real cases is obtained from possible cases by considering pairs of converse relationships as a single case.*

Finally, the significance of the 4-Intersection Matrix rests on the following: when two configurations have different 4-intersections matrices then these configurations are topologically different. But, if two configurations have the same 4-Intersection matrices then these configurations are topologically similar. However they are not necessarily the same. Figure 2.6 shows an example of topological similarity.

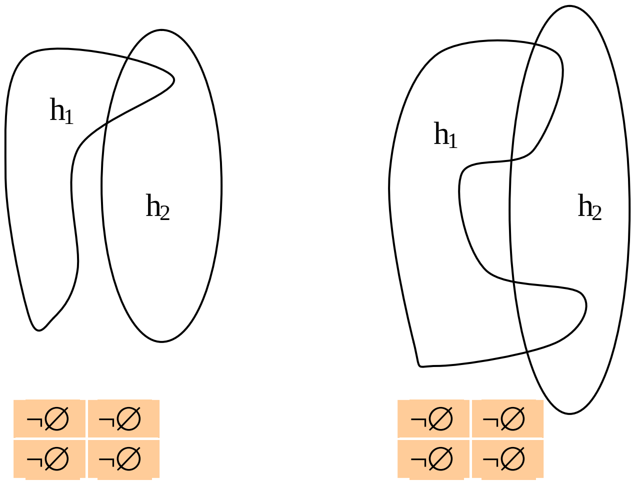


***Figure 2.6.** Example of Topological Similarity. Bellow the graphical representation of the values for the 4IM matrix are shown.*

The set of valid 4-Intersention matrices are complete in the sense that for any possible configuration of two regions, this set has always one topological relation that applies. And each matrix is mutually exclusive to other matrices because between two regions, exactly one of the matrices applies.

However the set is also redundant because some matrix situations can be expressed by means of other matrices. For instance, the case represented by the matrix $\begin{pmatrix} \neg\Phi & \Phi \\ \neg\Phi & \Phi \end{pmatrix}$ between $h_1$ and $h_2$ represents the same situation as the matrix $\begin{pmatrix} \neg\Phi & \neg\Phi \\ \Phi & \Phi \end{pmatrix}$ between $h_2$ and $h_1$.

Moreover with the 4-Intersection method we miss a part which is the exterior of a feature, and this means that there will be cases which cannot be described, as the ones in figure 2.7. In this figure both line relations have associated the same 4-Intersection Matrix. However in figure 2.7 a) part of the blue line's interior runs through the red line's exterior, and in figure 2.7b) part of the blue line's boundary is located in the red line's exterior. Therefore, using the exterior feature these cases could be differentiated. This is the reason why the **9-Intersection Method (9IM)** appears.

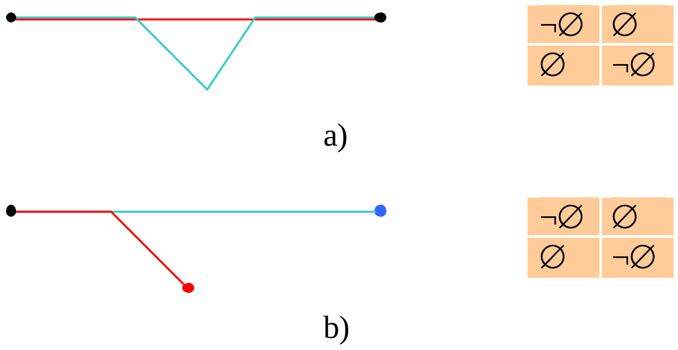


***Figure 2.7.*** *Line relations that could be differentiated using the concept of exterior.*

To solve this type of problem, the 9IM method appears. The 9IM is an extension of the 4IM based on considering also exterior of features, besides interior and boundary [Egenhofer and Herring 95]. Therefore, it is necessary to consider the following matrix of nine sets:

$$M = \begin{pmatrix} h_1^{\circ} \cap h_2^{\circ} & h_1^{\circ} \cap \partial h_2 & h_1^{\circ} \cap h_2^{-} \\ \partial h_1 \cap h_2^{\circ} & \partial h_1 \cap \partial h_2 & \partial h_1 \cap h_2^{-} \\ h_1^{-} \cap h_2^{\circ} & h_1^{-} \cap \partial h_2 & h_1^{-} \cap h_2^{-} \end{pmatrix}$$

By considering the empty or non-empty content of such nine sets, the total is $2^9$=512 theoretical combinations. As with the 4IM, excluding the impossible cases, we have 68 possible cases. Considering also the converse relationship, we can exclude another 2 cases for area/area group and 10 cases for line/line group, having a total of 56 real cases.

| relationships groups | No. of possible cases | No. of real cases |
|---|---|---|
| area/area | 8 | 6 |
| line/area | 19 | 19 |
| point/area | 3 | 3 |
| line/line | 33 | 23 |
| point/line | 3 | 3 |
| point/point | 2 | 2 |
| | | Total: 56 |

***Table 2.2***. *A summary of the 9IM*

One can use this calculus to reason about regions, and about regions which have holes by classifying the relationship not only between each pair of regions, but also the relationship between each hole of each region and the other region and each of its holes [Egenhofer, Clementini and Di Felice 94].

However, with the 9IM we still miss a part which is the dimension of a feature, and this means that there will be also cases which cannot be described. This is the point where the **Dimension Extended Method (DEM)** appears [Clementini et al. 93], which takes into account also the dimension of the intersection. The DEM can be considered as an extension of the 4IM, in which the dimension of the four intersection sets assume the values:-,0,1,2, meaning that the dimension is undefined (-), is set to 0 (dimension of points), 1 (dimension of lines) or 2 (dimension of areas).

Theoretically, these 4 possibilities result into $4^4$=256 different cases. However they adopted geometric criteria to reduce the number of cases by referring to specific groups of relationships. In the line/area group, only the following results are possible:

$$\begin{pmatrix} \dim(\partial A \cap \partial L) & \dim(\partial A \cap L^{\circ}) \\ \dim(A^{\circ} \cap \partial L) & \dim(A^{\circ} \cap L^{\circ}) \end{pmatrix} = \begin{pmatrix} \{-,0\} & \{-,0,1\} \\ \{-,0\} & \{-,1\} \end{pmatrix}$$

This is due to the fact that the dimension of the intersection cannot be higher than the lowest dimension of the two operands of the intersection: dim(∂A)=1, dim(A$^{o}$)=2, dim(∂L)=0, and dim(L$^{o}$)=1. Further, the definitions of line and area features exclude the option that dim(A$^{o}$∩L$^{o}$)=0. Following this discussion, the number of cases decreases from 256 to 24. Other geometric considerations bring the number of possible cases to 17.

In the area/area group of relationships, the following results of the intersections are possible:

$$\begin{pmatrix} \dim(\partial A_1 \cap \partial A_2) & \dim(\partial A_1 \cap \overset{\circ}{A_2}) \\ \dim(\overset{\circ}{A_1} \cap \partial A_2) & \dim(\overset{\circ}{A_1} \cap \overset{\circ}{A_2}) \end{pmatrix} = \begin{pmatrix} \{-,0,1\} & \{-,1\} \\ \{-,1\} & \{-,2\} \end{pmatrix}$$

After a detailed analysis, it is possible to identify 12 impossible cases and 3 pairs of converse relationships, resulting in 9 real topological relationships.

In the line/line group, the four sets may be equal to the following results:

$$\begin{pmatrix} \dim(\partial L_1 \cap \partial L_2) & \dim(\partial L_1 \cap \overset{\circ}{L_2}) \\ \dim(\overset{\circ}{L_1} \cap \partial L_2) & \dim(\overset{\circ}{L_1} \cap \overset{\circ}{L_2}) \end{pmatrix} = \begin{pmatrix} \{-,0\} & \{-,0\} \\ \{-,0\} & \{-,0,1\} \end{pmatrix}$$

It is possible to find 24 different cases and distinguish 6 pairs of converse relationships, resulting in 18 real cases.

Finally, with regard to groups involving point features, since the result of the intersections may be empty or zero-dimensional, we do not have more cases than in the standard 4IM. Table 3 is a summary of the analysis for all the groups of topological relationships, totalling 52 real cases.

| **relationships groups** | **No. of possible cases** | **No. of real cases** |
|---|---|---|
| area/area | 12 | 9 |
| line/area | 17 | 17 |
| point/area | 3 | 3 |
| line/line | 24 | 18 |
| point/line | 3 | 3 |
| point/point | 2 | 2 |
| | | Total: 52 |

***Table 2.3.*** *A summary of the DEM.*

However, the number of 52 relationships is still far too many for humans to use in a reasonable manner. It is better to have an overloaded set of just a few basic relationships which the user understands well. The DEM uses various results of feature intersections together with the boundary and interior operators to describe the required relationships. It may be clear that it is not a very user-friendly method, as the user is not interested in the intersections of the boundaries and the interior. Moreover, the concept of interior is less understood than the concept of boundary, because it is based on the mathematical point-set theory (open/closed sets). Therefore Clementini, Di Felice and Oosterom in [Clementini et al. 93] developed the **Calculus Based Method (CBM)** taking into account the above considerations by making available to the users only the boundary operators together with five topological relationships: *touch, in, cross, overlap,* and *disjoint* (figure 2.8*).* These topological relations are general in the sense that they are applied to point, lines and areas. They stated that this is the smallest set of relationships capable of representing all cases of the DEM under the condition that only the additional boundary operators are available. Furthermore, the set of topological relationships is close to the normal human use of these concepts and still powerful enough to represent a wide variety of cases. They also prove that the relationships are mutually exclusive and they constitute a full covering of all topological situations. [Clementini & Di Felice 94] proves that CBM is even more expressive than the DEM, because we need to define a combination of the DEM and the 9IM in order to find an equivalent method which the same expressive power of the CBM.

It is important to remark that the CBM is not sensitive to line orientation.

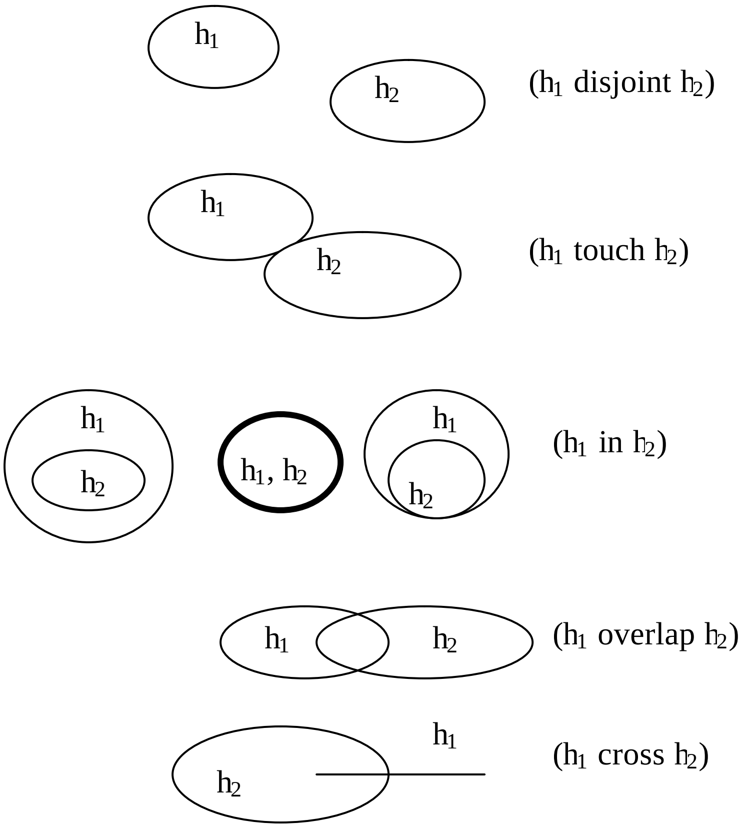


***Figure 2.8.*** *Graphical representation of the topological relationships defined in CBM.*

As we see in figure 2.8 there are several topological situations described under the relationship in. This has been one of the reasons why we have developed a new topological algebra with the same expressive power as the CBM but which is able to describe all the topological situations inside the *in*-relationship by using three more atomic relationships.

Figure 2.8 also points out that all the relationships defined in CBM can not be applied to all types of features, for instance the relation cross cannot be applied between two areas. We have defined a 3x3 matrix showing all the possible topological relationships of the CBM that can exist between each pair of regions (points, lines and areas).

The matrix is shown in the table 2.4. The regions (points, lines or areas) are situated in the first line and first column. In each entry we get the different topological relations that can hold between the entities that denote this entry.

The symbol * denotes that the relations in the entry are the same as the relations in its symmetrical entry, i. e. the topological relationships between a linear entity and a point-like entity are the same as the topological relationships between a point-like entity and a linear entity.

| | **Points** | **Lines** | **Areas** |
|---|---|---|---|
| **Points** | *In, Disjoint* | *Touch, In, Disjoint* | *Touch, In, Disjoint* |
| **Lines** | * | *Touch, In, Cross, Overlap, Disjoint* | *Touch, In, Cross, Disjoint* |
| **Areas** | * | * | *Touch, In, Overlap, Disjoint* |

***Table 2.4.*** *3x3 Matrix of possible topological relations between Points, Lines and Areas in the CBM.*

Finally, comparing all the point-set theories presented we can say that 9IM is potentially more expressive that 4IM, and as it considers the exterior of features, it considers the relationship between the region and its embedding space. However the more expressive method is the CBM calculus, which is more expressive than DEM or 9IM alone. CBM could be considered as the minimal set to represent all 9IM and DEM relations.

The next table presents a summary of topological cases for all these methods, excluding the converse relations, which denotes the expressiveness of the methods, showing the above conclusion. Each entry of the table shows the number of different topological relationships defined by each method.

| | A/A | L/A | P/A | L/L | P/L | P/P | Total |
|---|---|---|---|---|---|---|---|
| **4IM** | 6 | 11 | 3 | 12 | 3 | 2 | 37 |
| **9IM** | 6 | 19 | 3 | 23 | 3 | 2 | 56 |
| **DEM** | 6 | 17 | 3 | 18 | 3 | 2 | 52 |
| **DEM + 9IM or CBM** | 9 | 31 | 3 | 33 | 3 | 2 | 81 |

***Table 2.5.*** *Comparison of all point-set methods.*

## 2.2. STATE OF THE ART ON QUALITATIVE MOTION

Movement can be seen as a form of spatial change. For any quantity or quality that can be represented at an instant in time, we can also imagine it changing across time. Topological relations between entities can change as the entities move. The distance between two objects, the orientations of two lines, and the shape of two objects can all change with time.

We are going to examine in this section various approaches towards the representation of motion.

The Newtonian conception of space and time has exerted a strong influence on views about motion even outside physics. However, a lot of the approaches close to our concerns depart somewhat from the conception of motion as a continuous function from time (seen as the real line) to space.

The different approaches can be distinguished with respect to a few key choices about the ontology of space, time, and thus motion:

1. The choice of an absolute space (persisting through time and existing independently of the objects in it) versus a relative space, where only physical objects have an existence and are located with respect to one another. This leads to the distinction between Euclidean geometry (absolute location) and topological properties (relative position).
2. The choice of extended regions as primitive objects versus the choice of dimensionless points, either for time, space or both (hybrid solutions are not uncommon in literature).
3. The choice of expressing motion as relative to other entities or as absolute in a coordinate system.

4. The choice of a discrete or dense or continuous time and/or space; this leads to the distinction between measured time (absolute chronology) and ordered events (historical sequences). A fully, explicit, discrete model of motion is rare, but see a proposal in [Forrest 95].
5. The choice of a primitive space-time versus two separate domains for space and time.

Most of those choices can be done independently; therefore it is difficult to classify movement approaches in a clearly separated division. However, here we have made a classification, presenting two main groups of approaches:

1. Approaches which have chosen to represent two separated domains for space and time.
2. Approaches which have chosen to represent a primitive space-time domain.

Inside each group, we will find approaches considering the other choices that have to be made. Therefore, in each group we can find approaches dealing with absolute movement, or relative movement, with absolute or relative space or time, and with point, extended objects or both types as primitive either for time, space or both. Table 2.6 summarises all the approaches presented in both groups considering all the choices to be done.

The classification has been stated in this way, because the decision of considering two separated domains for space and time, or to consider space-time as an homogeneous domain has been one of the most debated ones when developing our contribution. However, since modern physics does not treat space and time as independent dimensions, but combines the two concepts into a single manifold (space-time domain), physicists are able to significantly simplify the form of most physical laws, as well as to describe the workings of the universe in a more uniform way, for us it seems rather natural to consider a space-time domain.

In fact, models which consider a space-time primitive domain combine three dimensions of space and one dimension of time into a single construction called the space-time continuum (in which time plays the role of the 4th dimension). This way to integrate space and time is made according to Euclidean space perception, in which our universe has three dimensions of space, and one dimension of time.

### 2.2.1. Movement Approaches with Separated Domains for Space and Time.

As table 2.6 shows, this is the most extensive group of approaches. In fact, most of the approaches in motion modelled it as a sequence of changes of positions, taking into account conceptual neighbourhood ([Zimmermann and Freksa 93]), but without integrating the concept of time into the same model. They consider space and time independent and express motion as a relation between the two domains, space and time.

Below we give a survey of the most relevant works in this group by showing if they deal with relative or absolute space, time, or movement, and if they deal with points or extended objects for space, time or both.

Absolute, Euclidean space and a separate continuous time, form the basis of pre-relativistic physics (the primitive objects being points in space and instants in time). This conception is also at work in robotics ([Stein and Musto 00]) and in studies grouped under the "qualitative physics" label ([Forbus 83], [Forbus 95], [Faltings 90], [Davis 88]). [Rajagopalan and Kuipers 94] can be classified inside the group of "qualitative physics" for extended objects using a Newtonian/Cartesian framework for representing motion.

The relative nature of space is on the contrary advocated for in most cognitively oriented approaches, as in linguistics for instance ([Asher and Sablayrolles 95]).

Finally, among the proponents of extended objects as primitives (either regions of space or intervals for time) can be found supporters of an absolute pre-existing space ([Galton 97], [Borgo et al. 96]) or of a relative space ([Bennett, Cohn, Torrini and Hazarika 00a], [Cohn and Hazarika 01b], [Clarke 81], [Van de Weghe et al. 04]). Some of these are in fact hybrid as they admit also points as objects ([Galton 97], [Clarke 81]).

The work of [Van de Weghe et al. 04] also expresses motion as relative because it expresses the motion of two objects that are moving towards each other or away from each other.

### 2.2.2. Movement Approaches with a Homogeneous Primitive Space-Time Domain.

This group covers the approaches which incorporate the notion of time into spatial representations having a combination of spatial and temporal logics ([Muller 98a, 98b], [Hayes 85], [Wolter and Zakharyaschev 00], [Claramunt et al. 97], [Hornsby and Egenhofer 02]). Since the work of [Muller 98a, 98b] has been the one which most influenced our work we will explain it in more detail.

[Hornsby and Egenhofer 02] and [Claramunt et al. 97] propose extended objects as primitives (regions of space and intervals for time) in a relative space.

[Hornsby and Egenhofer 02] considers objects as primitives and develops a changed-based approach in the sense that it does not consider time and space as separate dimensions, and it concentrates on recording changes or facts that are valid at a certain time.

The approach by [Claramunt et al. 97] is a hybrid approach in the sense that it admits also points as objects and in the sense that it develops a system to operate simultaneously on absolute and relative views of space and time.

On the other hand, Wolter and Zakharyaschev [Wolter and Zakharyaschev 00] propose languages that are combinations of two well-known and well-understood formalism in temporal and spatial reasoning. The spatial component is the RCC-8. As temporal component they chose the point based prepositional temporal logic **PTL** with the binary operators "Since" (*S*) and "Until"(*U*) based on the flow of time. Thus, the problem of constructing effective spatio-temporal formalism is viewed as designing decidable two-dimensional modal logics one dimension of which is a topological space and another one the flow of time. Therefore, RCC8 is temporalised using *S* and *U*, and other standard operators that are defined via *S* and *U*, as ? (at the next moment), $^{+}$ (always in the future), $\Diamond^{+}$ (some time in the future), etc. They stated that there are different ways of introducing a temporal dimension into the syntax of RCC8, ranging them from the less expressive one (or the simplest one, $ST_0$) to the most expressive ($ST_2$).

For instance, $ST_0$ is enough to express, for instance, the assumption that change is continuous, or the notion of conceptual neighbourhoods, as the next example illustrates:

$^{+}$(DC(X,Y) → ? (DC(X,Y) $\vee$ EC(X,Y))),

$^{+}$(EC(X,Y) → ? (DC(X,Y) $\vee$ EC(X,Y) $\vee$ PO(X,Y))),

$^{+}$(PO(X,Y) → ? (EC(X,Y) $\vee$ PO(X,Y) $\vee$ TPP(X,Y) $\vee$ EQ(X,Y) $\vee$ (TPP$^{-1}$(X,Y))), etc.

The first of these formulas, for instance, says that if two regions are disconnected at some moment, then at the next moment they either will remain disconnected or will be externally connected, but will not overlap.

Moreover, they make all these languages still more expressive defining $ST^{+}_{i}$. $ST^{+}_{i}$ make all the languages *STi* for *i*=0,1,2 more expressive by allowing applications of the Boolean operations to regions. Then it would be possible to write EQ(UK, Great_Britain U Northern_Ireland). Thus, the most expressive language is $ST^{+}_{2}$, which makes it possible to form unions, intersections, and complementation of spatial regions, and to apply temporal operators to both formulas and region terms (for instance, ? X denotes the state of region X "tomorrow").

With respect to the work of [Muller 98a, 98b], he presented a formal theory for reasoning about motion by enriching the topological concepts presented in [Asher and Vieu 95] to achieve a theory whose intended models are spatio-temporal entities. Therefore, the primitive objects of his theory are extended in space and time and knowledge about these entities is only expressed in terms of relations. However, the topological theory that Muller uses needs further structural specifications to be regarded as a proper spatio-temporal theory. Therefore, Muller selected an appropriate logic for temporal relations taking into account that his primitive objects are extended both in space and time. So, Muller develops a temporal theory, close to event logics which are quite close to interval-based temporal logics (as Allen's ones) with the difference that two objects can be different and still be contemporaneous. Therefore, Muller introduces a primitive of temporal connection noted as $\bowtie$, which has more or less the same behaviour as C (connection), only on a temporal level. With this temporal connection relation Muller defines next intuitive relations: *temporal inclusion* ($\subseteq_t$),*temporal overlap* ($\sigma$) and *temporal equivalence* ($\equiv_t$).

The figure bellow (figure 2.9) illustrates the temporal relations between spatio-temporal entities.

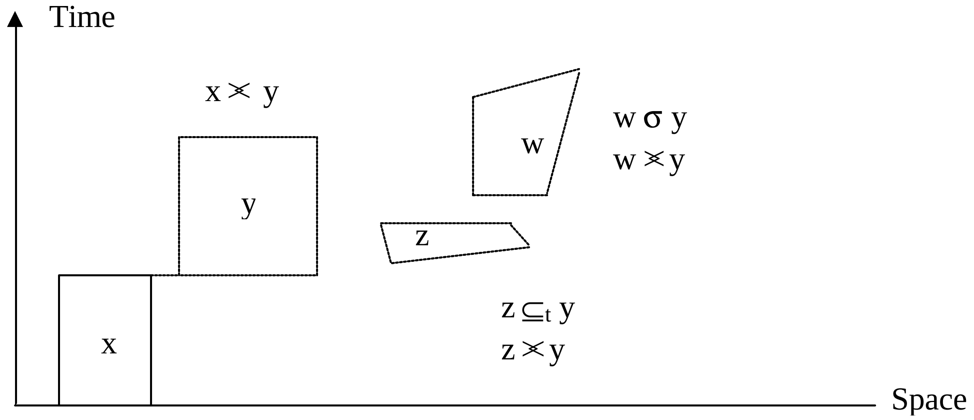


***Figure 2.9.*** *Examples of Temporal Relations under [Muller 98a, 98b] approaches.*

Given the topological theory and the temporal relations defined, Muller links them by stating that two connected entities are also time-connected.

Muller in [Muller 98a,b] gives a formal characterisation of the conceptual neighbourhood graph of RCC8 defining a continuity theory of motion using his theory, therefore using a unified framework. The continuity theory is close to intuition and can be defined without stating separately the possible transitions (as it is done when using separated domains for space and time).

Finally, in order to summarise all the approaches presented in the whole section 2.2 the next table presents the classification of all of them. The next notation is followed: A.S, R.S, A.M, R.M, A.T. and R.T. means Absolute Space, Relative Space, Absolute Motion, Relative Motion, Absolute Time and Relative Time respectively.

| **Approaches** | **A.S.** | **R.S** | **Points** | **Extended Regions** | **A.M .** | **R.M.** | **A. T.** | **R. T.** | **Space +Time Domain** | **Separate Space +Time Domains** |
|---|---|---|---|---|---|---|---|---|---|---|
| [Musto and Stein 00], [Forbus et al. 87], [Forbus 95], [Faltings 90], [Davis 90]. | X | | X | | X | | X | | | X |
| [Rajagopalan & Kuipers 94] | X | | | X | X | | X | | | X |
| [Asher and Sablayrolles 95] | | X | X | | X | | X | | | X |
| [Galton 97] | X | | X | X | X | | X | | | X |
| [Borgo et al. 96] | X | | | X | X | | X | | | X |
| [Bennett, Cohn, Torrini and Hazarika 00a], [Cohn and Hazarika 01] | | X | | X | X | | X | | | X |
| [Van de Weghe et al. 04] | | X | | X | | X | X | | | X |
| [Clarke 81] | | X | X | X | X | | X | | | X |
| [Claramunt et al. 97] | X | X | X | X | X | | X | X | X | |
| [Hornsby and Egenhofer 02] [Muller 98a, 98b], [Hayes 85], [Wolter and Zakharyaschev 00] | | X | | X | X | | X | | X | |

***Table 2.6.*** *Summary of the approaches classification*

## 2.3. STATE OF THE ART ON QUALITATIVE SHAPE REPRESENTATION

Human beings rely on qualitative descriptions of shape in many of their daily activities. Shape is an aspect of space that describes sets of points, in fact the shape of an object is the description of the properties of the boundary of the object. A ***single*** point has neither dimension nor shape. However a one-dimensional curve has a shape that can be described.

Shapes can be described in many different ways, ranging from quantitative to qualitative representations. A purely quantitative representation is when figures are described as mathematical functions of space coordinates. For instance a 2 dimensional round disk can be described by the following mathematical function:

$$x^2+y^2=r^2.$$

For more complex shapes, it is generally difficult to find a numerical function for the curve or surface describing the boundary of the figure. Piecewise interpolation methods are often used as a simplification. This means that the object to be described is approximated as consisting of many small parts, for instance straight lines or flat surfaces, for which it is possible to find numerical functions. The set of functions then makes up the quantitative description of the shape of the object. An alternative quantitative representation is to approximate the shape of the object by the pixels it occupies. Depending on the resolution, this gives a more or a less coarse result, since some pixels may be only partially filled. Furthermore, the description of the shape may be quite different if it is rotated or transfered within the grid.

One field of AI in which shape description is an important issue is artificial vision. In the artificial vision field a high computational cost for image processing is necessary. Moreover, object recognition from image processing is an unsolved problem, i.e. it is not yet possible to distinguish the same chair from different points of view or partially hidden by using quantitative image processing.

Since the **recognition** of an object is often possible only with partial information of its shape, as long as the key features of the object are available, we strongly believe that the use of qualitative techniques for object recognition would be very useful and could solve this problem. For example, in order to recognize a cat, it is often enough to see a pair of pointed ears and a furry tail. Therefore, the use of a qualitative theory for shape description and recognition will increase the efficiency in vision recognition because the recognition of a shape or an environment will be carried out by looking only for the distinguish features and not analysing each pixel of the image.

Moreover, it is strange to have two objects which represent the same object and which have the exact quantitative data. For instance, no two tiles are exactly identical. Due to the manufacturing process, they always have slight differences.

Once more the use of a qualitative method which would be able to recognise and match objects using only relevant features and managing uncertainty seems to be the most suitable one.

A purely qualitative representation may describe shapes by linguistic terms, such as "round", "straight" and so on. For instance, the example of a 2 dimensional round disk can be described qualitatively by the word *"round."* (figure 2.10)

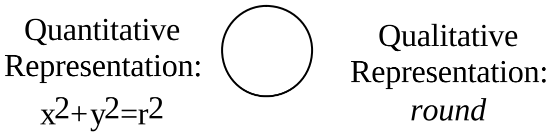


***Figure 2.10.*** *Examples of quantitative and qualitative representation of a round disk.*

Most of the qualitative approaches to shape description can be classified as follows:

1. *Axial representations:* approaches based on a description of the axes of an object, representing the shape qualitatively by reducing it to a "skeleton" or "axis". The "axis" is a planar arc reflecting some global or local symmetry or regularity within the shape.

2. *Primitive-based representations:* approaches where complex objects are described as combinations of more primitive and simple objects.

3. *Orientation and Projection-based representations or reference points representations*: in these approaches different aspects of the shape of an object are represented by either looking at it from different angles or by projecting it onto different axe.

4. *Topology and logic-based representations*: these approaches rely on topology and/or logics representing shapes.

5. *Cover-based representations*: in these approaches the shape of an object is described by covering it with simple figures, as rectangles or spheres.

Next sections detail each one of these classes.

## 2.3.1. Axial Representations.

**Axial representations** include approaches based on a description of the axes of an object ([Leyton 88], [Brady 83], [Galton and Meathrel 99], [Meathrel and Galton 00], [Meathrel and Galton 01]). A qualitative way of describing a shape is to reduce it to a “skeleton” also called “axis”. The axis is a 2 dimensional arc reflecting some global or local symmetry or regularity within the shape. The shape can be generated from the axis by letting a geometric figure, called the generator, move along the axis and sweep out the boundary of the shape. The generator is assumed to be of constant shape and to keep a specified point, for instance its centre, fixed to the axis as it moves. However, the generator can change both its size and its inclination with respect to the axis. Axial representations are qualitative since many different shapes can be generated from a given axis, using generators of different shape and varying sizes and angles. For two-dimensional shapes generally either a circle or a straight line segment is used as a generator. Figure 2.11 illustrates this explanation.

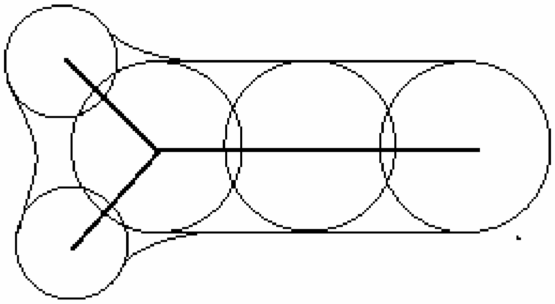


***Figure 2.11.*** *The axis of a 2D shape.*

One important type of approache that can be classified within Axial representations approaches, are the Curvature Extrema approaches [Hoffman and Richards 85], [Leyton 88]. They are based on the idea which comes from the fact that when asked to indicate the most salient points of a contour many people choose the points of extrema curvature, which means the points where the curve bends the most. Curvature is a mathematical function that indicates how fast the tangent of the contour rotates with respect to arc length. Five qualitative points are selected as the interesting ones to indicate the extrema curvature points (figure 2.12). These five points are: the maxima/minima of positive curvature (M+ and m+), the maxima / minima of negative curvature (M- and m-) and the zero crossing. These points correspond to psychologically salient points on the corresponding contour.

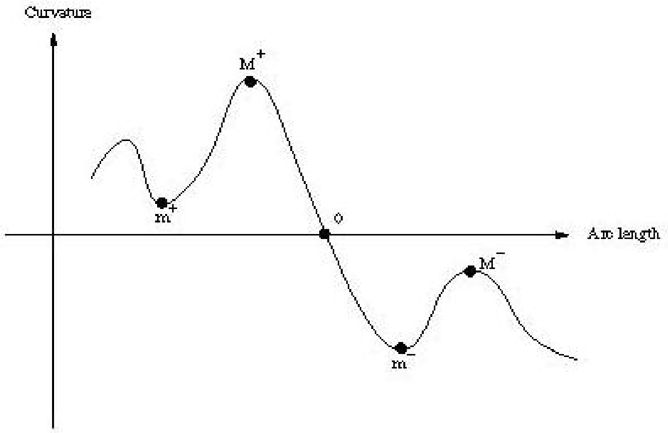


***Figure 2.12.*** *Curvature as a function of arc length*

Then, in order to describe parts of a contour, Richards and Hoffman [Hoffman and Richards 85] define codons, which are line segments composed of two or three curvature extrema. Three types of curvature extrema are considered in their representation: minimal negative (m-), Maximal positive (M+) and zero curvature.

Leyton [Leyton 88] carried further this idea by considering not only the sharpest points around a contour but also its flattest points. The latter correspond to points of minimal positive curvature (m+) and maximal negative curvature (M-).

Meathrel and Galton in ([Galton and Meathrel 99], [Meathrel and Galton 00], [Meathrel and Galton 01]) follow this trend of axial representations, characterising the shape by means of a string of tokens, recording salient curvature-based features encountered during a traversal of the outline of the shape. By considering variations in curvature as a starting point, they derived two sets of *atomic tokens* for describing curves, and presented *token ordering graphs* for verifying the syntax of atomic curve descriptions. The atomic tokens are obtained combining discretisations of tangent bearing and curvature. For curvature they use the discrete quantity space {+,0,-,U}, since at any point along a curve the curvature may be positive, zero, negative, or undefined. For tangent bearing they use the space {D,U}, whether the tangent bearing is defined or undefined respectively. Points where tangent bearing is undefined correspond to angles and cusps. Therefore, at any point *p* along a curve, its *curve state* is being specified by the pair $<b_p,c_p>$, where $b_p$, $c_p$ are the qualitative values of the tangent bearing and curvature at *p*, respectively. With this definition, they obtain the set of *atomic tokens* by considering for each valid pair $<b_p,c_p>$.

### 2.3.2. Primitive-Based Approaches.

**Primitive-based approaches** include approaches where complex objects are described as combinations of more primitive and simple objects. These approaches have some overlap with axial representations. Primitive-based approaches in 3D shape representation describe an object in terms of solid primitives covering its volume. Basically primitive-based schemes can be classified into:

6. *Generalized cylinder and geon-based representations*, which describe an object as a set of primitives plus a set of spatial connectivity relations among them ([Biederman 87], [Flynn & Jain 91]), [Shokoufandeh, Marsic and Dickinson 99], [Shokoufandeh, Dickinson et al. 02]).
7. *Constructive representations*, which describe an object as the Boolean combination of primitive point sets ([Requicha 80], [Brisson 89], [Ferrucci & Paoluzzi 91]).

These schemes basically differ in the type of description they provide and the application field in which they are used. Generalized cylinders provide a qualitative description of an object and then the information embedded in such models can be used to distinguish between different objects. However it cannot be used to generate synthetic images. And on the other hand, constructive representations have been used as quantitative description in CAD, where the primitives are specified in terms of numerical parameters, thus they can be used to generate a synthetic image of an object. Figure 2.13 shows examples of these two trends.

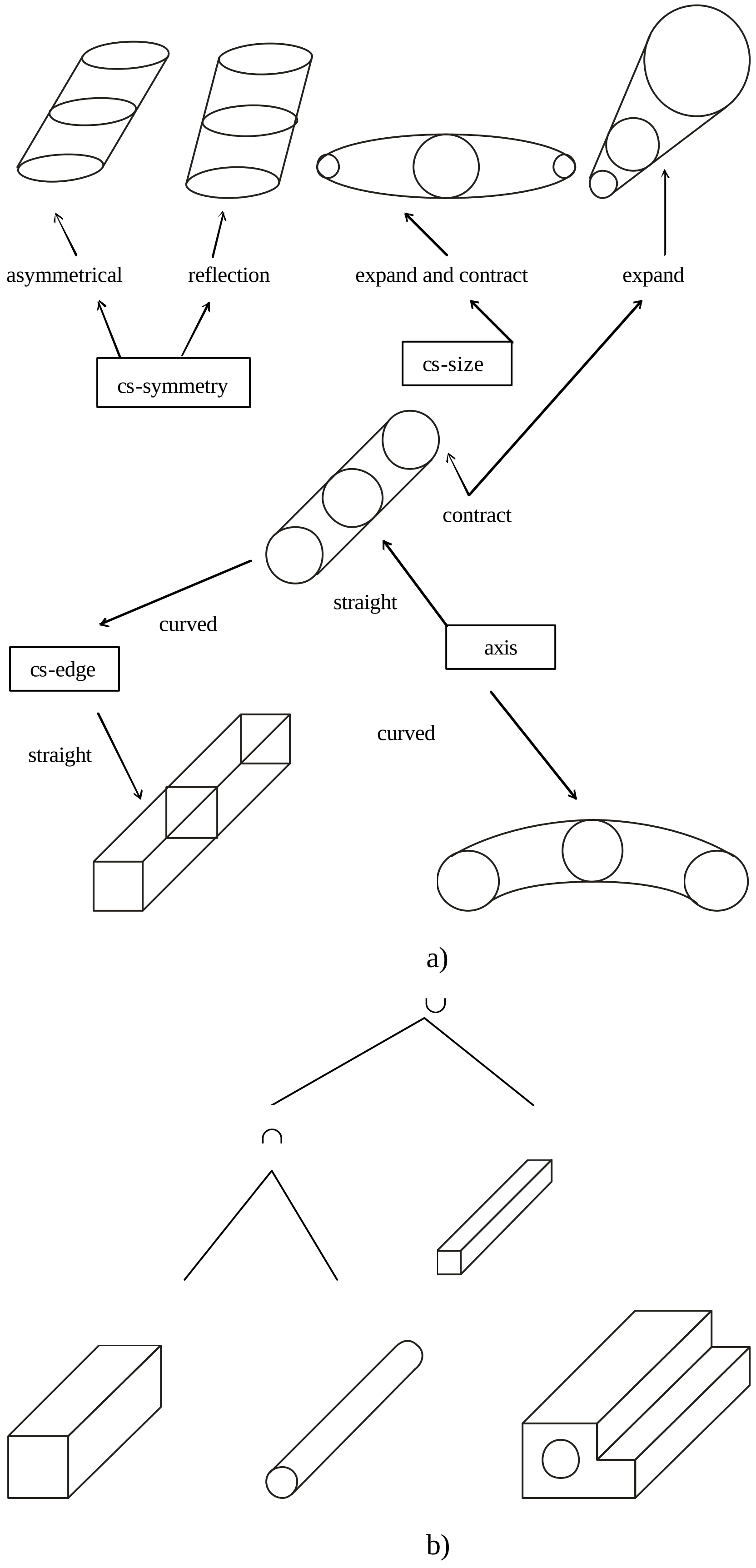


***Figure 2.13.*** *Examples of primitive based representations: a) Generalised Cylinders Example and b) Constructive Representations.*

### 2.3.3. Orientation, projection-based and reference-points representations.

**Orientation and projection-based representations or reference-points representations** include approaches where different aspects of the shape of an object are represented either by looking at it from different angles or by projecting it onto different axis ([Jungert 94], [Schlieder 96], [Freeman & Chakravarty 80], [Chen & Freeman 90], [Park & Gero 99], [Damski & Gero 96]). Most of these approaches are suitable for object recognition in image understanding. In these types of representations a fundamental issue is whether to store the representations of objects in an object-centred or in a viewer-centred coordinate. The problem is that, if the objects can be positioned in a scene with any possible orientation then descriptions of its surfaces from any viewpoint are required, in this case we have a viewer-centred description of the object. Then we have to consider multiview representations which describe objects through a finite set of viewer-centred descriptions, storing a set of projections or views of all admissible 3D objects in a scene.

An important study within this group, is the work of Jungert [Jungert 94]. Jungert's work is based on symbolic projections. One main aspect of symbolic projections is that the objects and the interrelations between them are formalized as strings. These symbolic projection strings of the description of objects and their relations are generated from projections of the objects down to the coordinate axis (figure 2.14). The string description of a shape represents qualitatively some feature of the vertices of the shape. These features are: if they are convex or concave vertices, their relative angles as acute, right-angled or obtuse, and in the case of being ended points if they are in the north, south, west or east.

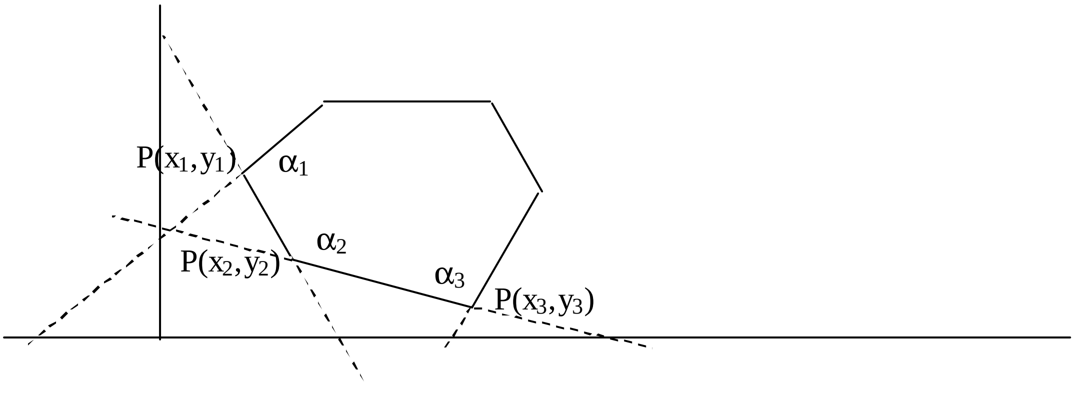


***Figure 2.14.*** *Example of Jungert's slope projections of an object down to the coordinate axis.*

Furthermore, curvatures of primitive sub-forms of the objects can be determined as well. Jungert uses slope projections and not perpendicular projections, in order to be able to determine whether a point is to the right or left of a line that is not perpendicular to the coordinate axes. Then, to determine the symbolic shape description of an object, Jungert projects all object sides down both to the x- and y- axes as figure 2.14 illustrates. Reasoning about the vertices to determine their features is based on the incoming and outgoing lines of each vertex. Figure 2.15 illustrates the basic principles of this projection technique, where the incoming line is $P_1$-$P_2$ and the outgoing is $P_2$-$P_3$. Projection is made down to the x-axis along or parallel to $P_1$-$P_2$ and to the y-axis perpendicular to that line. The following projections can be identified from figure 2.15:

$$U{:}x'_1x'_2<x'_3$$

$$V{:}y'_3<y'_2<y'_1$$

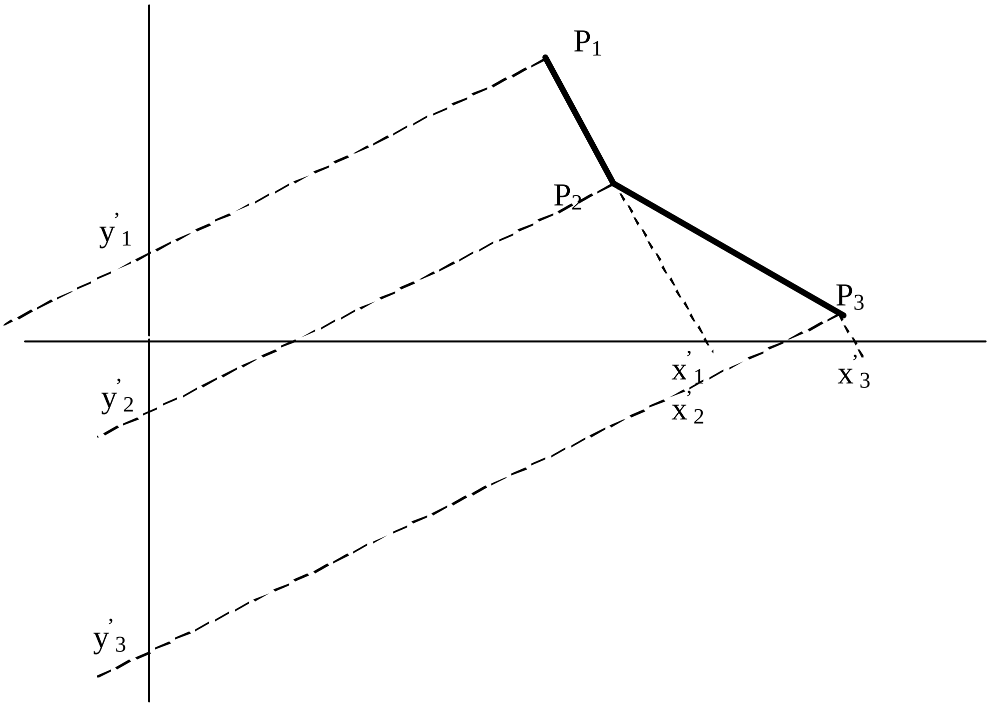


***Figure 2.15.*** *An incoming contour line $P_1$-$P_2$ and an outgoing $P_2$-$P_3$ and their projection lines.*

Therefore, for instance, the features for the point $P_2$ can be determined from the following reasoning technique: since the inside of the object is to the right when the contour is traversed from $P_1$ to $P_3$, a left turn at $P_2$ means that the object is convex. Furthermore, a right turn means a concave point, while a straight line cannot occur. Consequently, since $P_1$ and $P_2$ have projections in x ($x'_1$ and $x'_2$) with values less than the value of x'$_3$ of $P_3$, this can be interpreted as $P_2$ is convex. To determine the angle, Jungert studies the V string, if the value y'$_2$ ($P_2$) and the value y'$_3$ ($P_3$) are equal, a right-angle can be inferred. If y'$_3$ lies to the right of y'$_2$, then the angle is obtuse, otherwise acute. To determine the extreme points Junge rt uses next definitions:

north/south extreme points:

Vp: $y'_{j+1}<y'_j$ and $y'_{j+1}<y'_{j+2}$ → south

Vp: $y'_j<y'_{j+1}$ and $y'_{j+2}<y'_{j+1}$ → north

east/west extreme points:

Up: $x'_j<x'_{j+1}$ and $x'_{j+2}<x'_{j+1}$ → east

Up: $x'_{j+1}<x'_j$ and $x'_{j+1}<x'_{j+2}$ → west

By applying the above rules to all points in the object, it is possible to create a list of three different features which provide a qualitative description of the objects. Studying the concavities and convexities of objects, Jungert defines the primitive sub-forms. He states that there exist just two basic sub-contour types of which the first one is a curve that can be either convex or concave and the second is a zigzag contour. The simple sub-convexities and concavities can be combined, thus creating a class of sub-forms with twelve different instances. The simple contour parts can have a point that merges two parts so that the angle of that point is either obtuse (O), or acute (A) and in some cases also a right angle (R) as well. This point is called a passage. Hence, if the R-passage is excluded, the most common instances are depicted in figure 2.16.

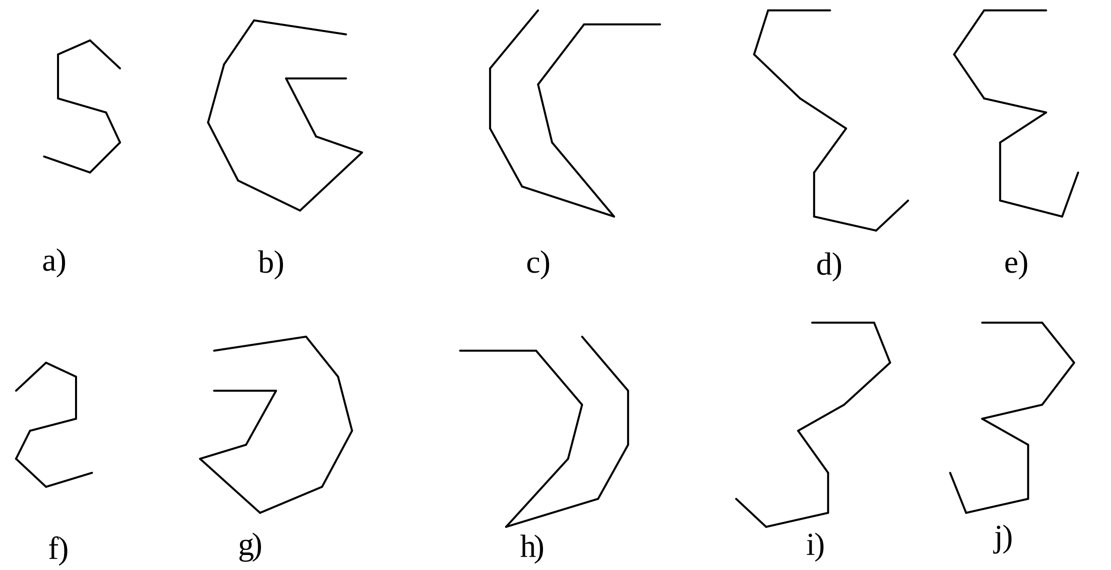


***Figure 2.16.*** *Sub-contours of type convex to concave with O-passage (a) and (b), convex to concave with A-passage (c), convex to convex with O-passage (d), convex to convex with A-passage (e), concave to convex with O-passage (f) and (g), concave to convex with A-passage (h), concave to concave with O-passage (i) and concave to concave with A-passage (j).*

## 2.3.4. Topology and Logic-based Approaches.

**Topology and logic-based approaches** are that ones that rely on topology and/or logic in representing shapes ([Bennett 94], [Cohn 95], [Randell & Cui & Cohn 92], [Clementini & Di Felice 97], [Chase 96, 97]). Benett ([Bennett 94]) uses the concept of convex-hull of a region to describe shapes. By convex hull of a region he means the smallest convex region of which it is a part. If one were to stretch an elastic membrane round a region then the convex-hull would be the whole of the region enclosed as figure 2.17 shows. By combining the 8 basic RCC relations with the convex hull operation we can specify 8 powered 4 relations of the form R1(X,Y) and R2(X, conv(Y)) and R3(Conv(X), Y) and R4(conv(X), conv(Y)), and these relations are used to describe the shape. He classifies shapes as similar (in the same class) if their convex hulls are similar.

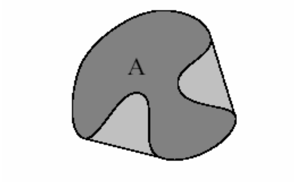


***Figure 2.17***. *Convex-hull in 2-dimensions.*

Scott C. Chase ([Chase 96, 97]), defines a method for describing designs by the combination of the paradigms of shape algebras and predicate logic representations. He models designs using spatial relations based upon a algebraic shape representation defined by him. This algebraic representation assumes that shapes are finite arrangements of basic geometric elements, which are points, lines, 2D regions and 3D solids. Each type of basic element has its own algebra by which it can be manipulated. For instance, in the case of lines, and shapes composed of lines, he states that in a shape all lines are maximal, which means that no line contains any parts which are part of another line. Therefore as shapes are simply finite sets of maximal lines, the subshape relation ≤ and the operations + (sum),- (difference) and • (product) are defined for shapes (figure 2.18). With this concept, he defines shape algebras and spatial relations. Therefore, shapes are composed of finite *basic elements* which are manipulated in algebras $U_{ij}$ indicating elements of dimension *i* in a space of dimension *j*. For example, $U_{02}$ and $U_{12}$ describe respectively points and lines in the plane. Each basic element in $U_{ij}$ is finite and can be distinguished by its *boundary* and *descriptor*. The boundary divides the design space between an element's interior (finite) and its exterior (infinite). It consists of a set of elements in the algebra of the next lowest dimension, for example, point bound lines, line bound planes, and so forth. Whilst the descriptor provides additional information about an element, for example the infinite element in which it is embedded, which in the case of a line segment would be the equation of its infinite line and in the case of an arc would be the equation of its circle. The operators +, -, •, and the relation ≤ operate only on elements with equal descriptors. Using the basic definition of shape and shape operations Chase defines a large number of geometric relations, as for instance *share_boundary(A,B)* and *surrounded_by(A,B).* Two basic elements A and B share a boundary if the product of their boundaries is non empty, and A is surrounded by B if and only if A is a part of B and they do not share a boundary.

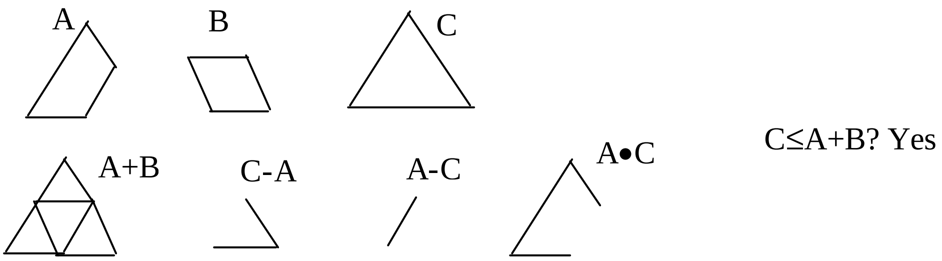


***Figure 2.18***. *Shapes A,B, and C (represented as sets of maximal lines) and results of operations upon them.*

## 2.3.5. Cover-Based Approaches.

**Cover-based approaches** are those approaches where the shape of an object is described by covering it with simple figures such as rectangles and spheres ([Del Pobil & Serna 95]). The sphere is the simplest of all geometric objects, and using only spheres they develop the next theory to describe a shape:

1. The elements that take part in the representation are only spheres, therefore any shape is reduced always to sets of spheres.
2. The model is twofold, it is composed of two sets of spheres: the spheres in the first set, which is the exterior representation, and cover the outer surface of the object, while the spheres in the second set, which is the interior representation, and are contained in the object.
3. A hierarchy of sets is defined for both representations so that the approximations for the object (for its outer bound and for its interior) are made better and better by using more spheres. Hierarchy is a rank of detail which means that the decomposition of the object and the approximation of its shape are obtained simultaneously and at different levels of detail which are used only when needed.
4. Finally a quality measure is used that permits them to quantify and control the degree of accuracy of the representation, both locally and globally. This problem is a NP-complete problem then they use a heuristic approach to solve it using what they call a rule-based expert named the spherizer. A remarkable feature of the approach is its stability with respect to shape changes.

Of course, there are some other representations that are not included in the above division and that are also interesting, for instance there are qualitative approaches that combine some of the techniques explained in the above characterisation, such as the work of Dugat, Gambarotto and Larvor [Dugat, Gambarotto and Larvor 02]. They provide a representation of shape of an object combining a notion of skeleton (describing the object's "architecture") and the thickness of the object over it, by a cover-based approach.

## 2.4. CONCLUSIONS OF THE STATE OF THE ART.

In this section we will comment on the approaches that have most influenced our work, showing the drawbacks we found in order to justify the development of our new theories for topology, movement and shape description.

With respect to topology we are interested in developing a calculus suitable for reasoning about topological relations between points, lines and areas as basic entities, therefore we cannot follow the trend established by the *Spatial Region Approaches* since this trend does not consider points and lines as regions. However, we will follow the trend established by the *Point-Set Approaches*. Specifically our topological calculus has been inspired by the CBM calculus of Clemetini [Clementini et al. 93]. The objects (features) manipulated by the CBM calculus consist of points, lines and areas commonly used in geographic information systems (GIS); that means that all kinds of features consist of closed (contain all their accumulation points) and connected (do not consist of the union of two or more separated features) sets. The knowledge about such entities is represented in the calculus as facts describing topological relations on pairs of the entities. The relation of two entities could be *disjoint, touch, overlap, cross, or in*.

When navigating using a map, it is often the case that one has to distinguish between a situation where a region is completely inside another region, the situation in which it touches it from inside, and the situation in which the two regions are equal. For instance, a driver might want to know whether Hamburg is strictly inside Germany or at (i.e., touches) the boundary of the country with the rest of the earth. The CBM calculus makes use of a boundary operator which, when combined with the in relation, allows for the distinction to be made. A close look at the calculus shows that, indeed, it is suited for conjunctive-fact queries of geographic databases (i.e. queries consisting of facts describing either of the five atomic relations or a boundary operator on a pair of features). In our topological contribution, we provide an Allen style approach [Allen 83] to the calculus, which means that we provide a constraint-based approach to the CBM calculus. Specifically, we present an algebra which will have nine atomic relations resulting from the refinement of the *"in"* relation, together with the other four atomic relations of the CBM calculus. Our main motivation is that we can then benefit from Allen style reasoning in the following aspects:

1. We can make use of Allen's constraint propagation algorithm to reason about knowledge expressed in the algebra. This means that composition tables recording the composition of every pair of the atomic relations have to be provided for the algebra, as well as a converse table. Therefore the calculus will be suitable for integration with other spatial aspects in order to reason about all of them within the same model as it has been done in [Escrig and Toledo 98]. We have to remember that we have defined three steps (section 1.4) in order to accomplish the integration of several spatial aspects in the same model:

   - The definition of the algebra of the spatial aspect to be integrated.
   - The definition of the Basic Step of the Inference Process (BSIP) for each spatial aspect to be integrated.
   - The definition of the Full Inference Process (FIP) for each spatial aspect to be integrated.

In fact, the definition of the topological algebra as an instance of the Constraint Satisfaction Problem (CSP) represents the first step. The definition of the converse and composition operations allows the implementation of the second step, the BSIP, which given two relationships which relate three objects (A relationship B), and (B relationship C), defines how to find the third relationship between objects A and C. The topological algebra defines the composition tables needed to compute this operation (BSIP), and in order to reduce the number of composition tables needed, the converse operation is defined too (see chapter 3 for more details).

2. The algebra will benefit from the incrementality of the propagation algorithm: knowledge may be added without having to revise all the processing steps achieved so far.

3. Disjunctive knowledge will be expressed in an elegant way, using subsets of the set of all nine atomic relations: such a disjunctive relation holds on a pair of features if and only if either of the atomic relations holds on the features. This is particularly important for expressing uncertain knowledge, which is closely related to the notion of conceptual neighbourhoods [Freksa 92].

Finally it should be remarked that the objects manipulated by our calculus are points, lines and areas; contrary to most constraint-based frameworks in the qualitative spatial and temporal reasoning literature, which deal with only one type of feature (for instance, intervals in [Allen 83] or regions [Cohn et al. 93]).

In the perspective of the representation of movement, we should ask ourselves what structures we think relevant in the following list: absolute versus relative space, extended regions or points for space and time, absolute motion versus relative motion, absolute chronology versus historical sequences, and a primitive space-time versus two separate domains for space and time. We have made the following choices:

- We want to represent knowledge with an axiomatic approach, by developing a constraint-based movement theory. Developing a constraint-based calculus of movement will allow us to integrate it with other spatial aspects following the three steps defined by [Escrig and Toledo 98] (section 1.4), as we do with topology.
- As with the case of the topological algebra, we want to develop a movement theory suitable for reasoning about the movement of points, lines and areas.
- As we consider that it seems rather natural to consider space-time as a homogeneous domain by considering regions of space-time, therefore the works of Muller and Wolter and Zakharyaschev have been the ones inspiring our movement theory. However these works consider as basic entities only regions, and we once more are interested in having a calculus powerful enough to manage also points and lines. Therefore, following these works, we have defined a constraint-based algebra for time suitable for integration with our topological algebra. In our movement calculus we represent topological information as a function of the point of the time in which it occurs as an instance of the Constraint Satisfaction Problem, obtaining the same advantages, mentioned above, for our topological calculus.

So, the movement theory developed uses a relative space because it deals with topological properties, it manages points and extended regions, it models absolute motion and absolute chronology, and finally it represents a primitive space-time instead of two separated domains. The resulting method can be applied to qualitative navigation of autonomous agents, for instance during the path planning task by describing the sequence of topological situations that the agent should find during its way to the target objective.

Finally, with respect to the Qualitative Shape Description theory, this part of our work has been inspired by the work of [Jungert 94]. [Jungert 94] is one of the approaches within the *Orientation and projection-based representations or reference-points representations* group. Therefore our work can be classified inside this group because we classify different aspects of shape by analysing its reference-points, which in our case as in the case of Jungert are the vertices. Jungert's work is based on symbolic projections which are generated from projections of the objects down to the coordinate axis. The use of an external reference system for describing a shape is not suitable for the recognition of shapes by an autonomous agent during its movement. In fact the use of external reference systems for describing the environment or the shape of objects seems to be not cognitive, since people do not project the features of objects to an exterior reference system when recognising them. Therefore we have developed a new theory inspired by Jungert's theory but without using an external reference system. The description of the shape is made as a function of its vertices and the relations between them. Moreover, Jungert describes curves as a sequence of straight lines, and we describe the curves themselves by describing its point of maximum curvature.

In fact, our approach consists of a reference-point information approach of the qualitative description of shapes considering qualitatively their angles, relative side length, concavities and convexities, and types of curvatures of the boundary. The shapes recognised are regular and non-regular closed polygons that can have curved segments and curvilinear shapes. Moreover, the shapes can contain holes. To describe shapes with holes, topological and qualitative spatial orientation aspects are considered in order to relate the hole with its container. The colour is also a feature which can be managed qualitatively with our theory. Managing objects with holes as well as the colour, is a further innovation of the theory with respect to the bibliography.

# CHAPTER 3

## THE ALGEBRA ON TOPOLOGY: INTEGRATION WITH OTHER SPATIAL ASPECTS

In this chapter we will develop a topological algebra to reason with regions that can be points, lines and areas which follows the Point-Set Approaches trend.

As our purpose is to integrate this algebra with other spatial aspects, we are going to follow the three steps explained in section 1.4. Therefore the chapter is divided in several sections as follows:

- The representation of topology, which means the algebra on topology we have developed is presented in section 3.1.
- The definition of the Basic Step of the Inference Process (BSIP) for topology is presented in section 3.2.
- And the definition of the Full Inference Process (FIP) for topology is defined in section 3.3.

Finally section 3.4 presents an example of reasoning about topological relations following the model developed.

### 3.1. THE REPRESENTATION OF TOPOLOGY: THE ALGEBRA DEVELOPED.

For the representation of topology we need a formal data model for topological relations and methods to combine topological knowledge. Moreover, in order to reason about topological information we need to define the BSIP and the FIP.

We have developed an algebra of the Calculus Based Method (CBM) calculus (see chapter 2 for details) as the one Allen ([Allen 83]) presented for temporal intervals.

> Specifically, we present an algebra which will have nine atomic relations resulting from the combination of the five atomic relations and the boundary operator of the CBM calculus.

Therefore, we have developed an algebra of which the atomic relations will be the three relations resulting from the refinement of the in relation, together with the other four atomic relations of the CBM calculus. We will provide the result of applying the converse and the composition operations to the atomic relations: this will be given as a converse table and composition tables. These tables in turn will play the central role in propagating knowledge expressed in the algebra using Allen's constraint propagation algorithm ([Allen 83]).

We will use the topological concepts of boundary, interior, and dimension of a (point-like, linear or areal) feature defined in section 2.1.2.

To define the topological relations of our algebra we use the original relations *touch, overlap, cross* and *disjoint* of the CBM calculus. As we cannot use the boundary operator which allows the CBM calculus to distinguish between the three subrelations equal, completely-inside, and touching-from-inside of the in relation (which are not explicitly present in the calculus) the three subrelations will replace the superrelation in the list of atomic relations. In addition, the new relations completely-inside and touching-from-inside being asymmetric, we need two other atomic relations corresponding to their respective converses, namely *completely-inside$_i$* and *touching-from-inside$_i$*. The definitions of the relations are given below. The topological relation r between two features $h_1$ and $h_2$, denoted by ($h_1$, r, $h_2$), is defined on the right hand side of the equivalence sign in the form of a point-set expression.

**Definition 3.1.** The *touch* relation:

$$(h_1, \text{touch}, h_2) \leftrightarrow \overset{\circ}{h}_1 \cap \overset{\circ}{h}_2 = \emptyset \wedge h_1 \cap h_2 \ ?\ \emptyset$$

The figure 3.1 shows graphical examples of the touch relation. The notation in this figure and in subsequent figures is as follows: the areas are denoted by a the letter "A", the lines by a "L" and the points by a "P", and in the case of having several entities of the same type (for instance two areas), then these letters are followed by a subindex to differentiate both entities.

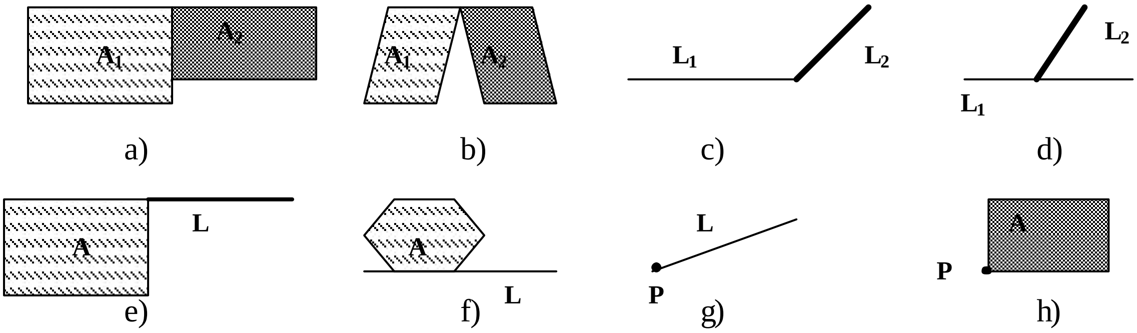


***Figure 3.1.*** *Graphical examples of the touch relation between two areas (figures 3.1a) and 3.1b)), between two lines (figures 3.1 c) and 3.1 d)), between an area and a line (figures 3.1 e) and 3.1 f)) and between a point and an area (figures 3.1 g) and 3.1 h)).*

**Definition 3.2.** The *cross* relation:

$(h_1, \text{cross}, h_2) \;\leftrightarrow$

$$\dim(\overset{\circ}{h}_1 \cap \overset{\circ}{h}_2)=\max(\dim(\overset{\circ}{h}_1), \dim(\overset{\circ}{h}_2)) - 1 \wedge h_1 \cap h_2 \ ?\ h_1 \wedge h_1 \cap h_2 \ ?\ h_2$$

Figure 3.2 shows graphical examples of the *cross* relation.

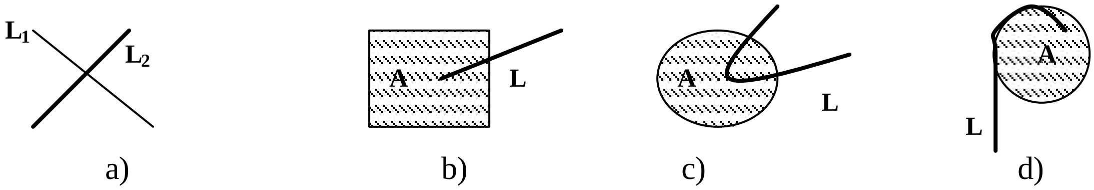


***Figure 3.2.*** *Graphical examples of the cross relation between two lines (figure 3.2 a)), between an area and a line (figures 3.2 b), 3.2 c) and 3.2 d)).*

**Definition 3.3.** The *overlap* relation:

$(h_1, \text{overlap}, h_2) \;\leftrightarrow$

$$\dim(\overset{\circ}{h}_1) = \dim(\overset{\circ}{h}_2) = \dim(\overset{\circ}{h}_1 \cap \overset{\circ}{h}_2) \wedge h_1 \cap h_2 \ ?\ h_1 \wedge h_1 \cap h_2 \ ?\ h2$$

Figure 3.3 shows graphical examples of this relation.

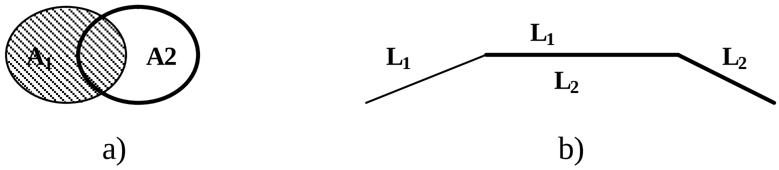


*Figure 3.3. Graphical examples of the overlap relation between two areas(figure 3.3 a)) and between two lines (figure 3.3 b)).*

**Definition 3.4.** The *disjoint* relation:

($h_1$, disjoint, $h_2$) $\leftrightarrow$ $h_1 \cap h_2 = \emptyset$

Figure 3.4 shows some graphic representations of the *disjoint* relation.

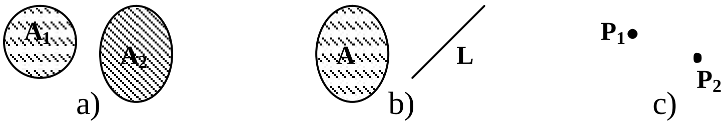


*Figure 3.4. Graphical examples of the disjoint relation between two areas (figure 3.4 a)), between an area and a line (figure 3.4 b)), and between two points (figure 3.4 c)).*

**Definition 3.5.** We define the *equal*, *completely-inside*, and *touching-from-inside* relations using the formal definition of the *in* relation:

($h_1$, in, $h_2$) $\leftrightarrow$ $h_1 \cap h_2 = h_1 \land \mathring{h}_1 \cap \mathring{h}_2$ ? $\emptyset$

Given that ($h_1$, in, $h_2$) holds, the following algorithm distinguishes between the completely-inside, the touching-from-inside, and the equal relations:

```
if ($h_2$, in, $h_1$) then ($h_1$, equal, $h_2$)
else if $h_1 \cap \delta h_2$ ? $\emptyset$ then ($h_1$, touching-from-inside, $h_2$)
else ($h_1$, completely-inside, $h_2$)
```

Figure 3.5 shows graphic examples of the *equal* relation.

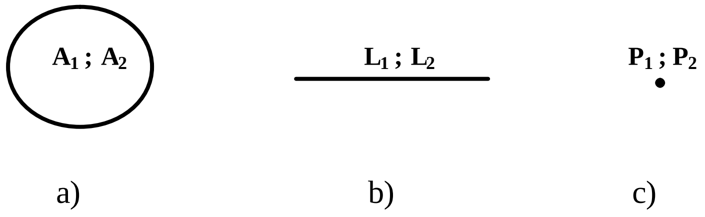


***Figure 3.5.*** *Examples of the equal relation. Figure 3.5 a) represents two equal areas, figure 3.5 b) represents two equal lines, and figure 3.5 c) represents two equal points.*

**Definition 3.6.** The *completely-inside$_i$* relation:

($h_1$, completely-inside$_i$, $h_2$) ↔ ($h_2$, completely-inside, $h_1$)

**Definition 3.7.** The *touching-from-inside$_i$* relation:

($h_1$, touching-from-insidei, $h_2$) ↔ ($h_2$, touching-from-inside, $h_1$)

Figure 3.6 shows graphic examples of the relations *complete-inside* and *touching-from-inside* and their converse relations (*completely-inside$_i$* and *completely-inside$_i$* ).

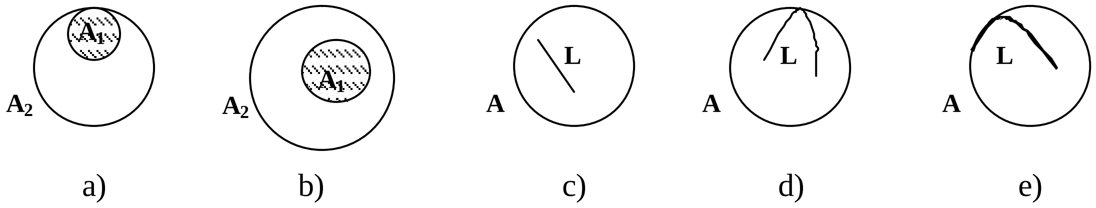


***Figure 3.6.*** *Examples of the completely-inside, touching-from-inside and their converse relations situations: examples with two areas, figure 3.6 a) representing (A1, touching-from-inside, A2) and (A2, touching-from-inside$_i$, A1), and figure 3.6 b) representing (A1, completely-inside, A2), and (A2, completely-inside$_i$, A1). Examples of situations with a line and an area, figure 3.6 c) representing (L, completely-inside, A), and (A, completely-inside$_i$, L),and d) and e) representing (L, touching-from-inside, A), and (A, touching-from-inside$_i$, L)situations.*

At this point we have defined the atomic relations of the new calculus. Now, we will prove that these relations are mutually exclusive, that is, it cannot be the case that two different relations hold between two features. Furthermore, we will prove that they form a full covering of all possible topological situations, that is, given two features, the relation between them must be one of the nine defined here. To prove these two characteristics we construct the topological relation decision tree depicted in figure 3.7.

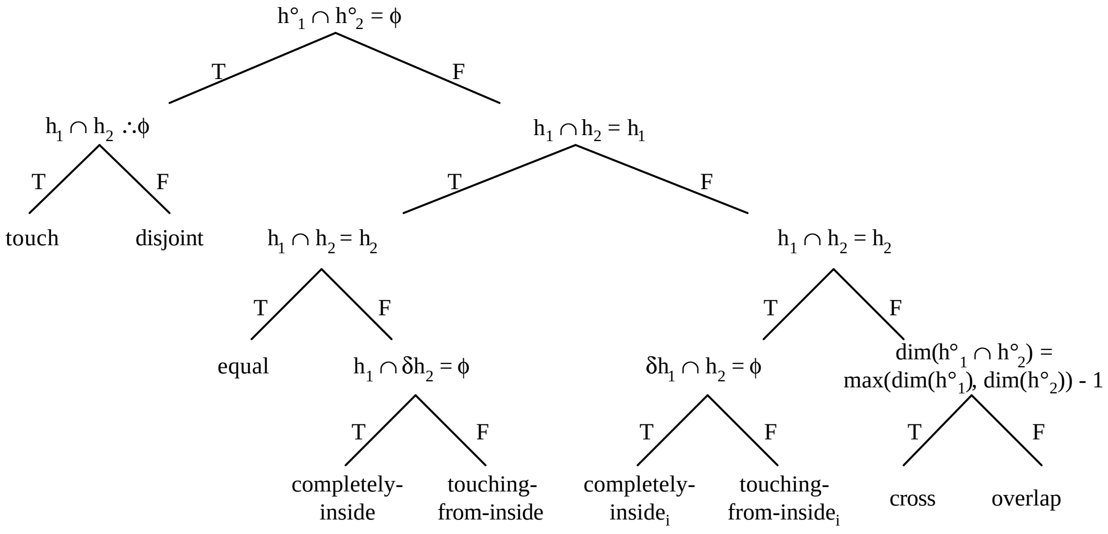


***Figure 3.7.*** *Topological relation decision tree*

**Proof.** Every internal node in this topological relation decision tree represents a Boolean predicate of a certain topological situation. If the predicate evaluates to true then the left branch is followed, otherwise the right branch is followed. This process is repeated until a leaf node is reached that will indicate which of the atomic topological relations this situation corresponds to. Two different relations cannot hold between two given features, because there is only one path to be taken in the topological relation decision tree to reach a particular topological relation. And there can be no cases outside the new calculus, because every internal node has two branches, so for every Boolean value of the predicate there is an appropriate path and every leaf node has a label that correspond to one of the atomic topological relations.

**Definition 3.8.** A general relation of the calculus is any subset of the set of all atomic relations. Such a relation, named R, is defined as follows:

$$(\forall h_1,h_2)\ ((h_1, R, h_2) \Leftrightarrow \bigvee_{r \in R} (h_1, r, h_2))$$

Given the original relationship ($h_1$, r, h2), by permuting $h_1$ and $h_2$ we can obtain one relationship more, which is the *converse* operation. Below we give the formal definition of the *converse* operation.

**Definition 3.9.** The *converse* of a general relation R is denoted as $R^{\cup}$. It is defined as:

$(\forall h_1, h_2)\ ((h_1, R, h_2) \Leftrightarrow (h_2, R^{\cup}, h_1))$, where

$R^{\cup} = \bigcup_{r \in R} \{r^{\cup}\}$, being r one of the atomic relationships.

Table 3.1 provides the converse for the atomic relations of the algebra.

| **r** | $\mathbf{r}^{E}$ |
|---|---|
| Overlap | Overlap |
| Touch | Touch |
| Cross | Cross |
| Disjoint | Disjoint |
| Completely-inside | Completely-inside$_i$ |
| Touching-from-inside | Touching-from-inside$_i$ |
| Completely-inside$_i$ | Completely-inside |
| Touching-from-inside$_i$ | Touching-from-inside |
| Equal | Equal |

***Table 3.1***. *The converse table*

Finally, we denote the universal relation by XY-U, with the U denoting the term "*universal*", and with X and Y belonging to {P, L, A} (we use P for a point, L for a line, and A for an area). The universal relation XY-U represents the set of all possible atomic relations, between a feature $h_1$ of type X and a feature $h_2$ of type Y. For instance, PP-U is the set of all possible topological relations between two points. These universal relations are as follows:

PP-U = {*equal, disjoint*}

PL-U = {*touch, disjoint, completely-inside*}

PA-U = {*touch, disjoint, completely-inside*}

LP-U = {*touch, disjoint, completely-inside$_i$*}

LL-U = {*touch, disjoint, overlap, cross, equal, touching-from-inside, completely-inside, touching-from-inside$_i$, completely-inside$_i$*}

LA-U = {*touch, cross, disjoint, touching-from-inside, completely-inside*}

AP-U = {*touch, disjoint, completely-inside$_i$*}

AL-U = {*touch, cross, disjoint, touching-from-inside$_i$, completely-inside$_i$*}

AA-U = {*touch, overlap, disjoint, equal, touching-from-inside, completely- inside, touching-from-inside$_i$, completely-inside$_i$*}

Note that the sets LP-U, AP-U, and AL-U are the converse sets of PL-U, PA-U, and LA-U, respectively.

## 3.2. THE BASIC STEP OF THE INFERENCE PROCESS FOR TOPOLOGICAL RELATIONS.

The Basic Step of the Inference Process (BSIP) for topological information consists of *iven two topological relationships between three objects in the space, ($h_1$, $r_1$, $h_2$) and ($h_2$, $r_2$, $h_3$), we want to infer the relationship ($h_1$, $r_3$, $h_3$)"*. To infer such relationship we need to define and then use the composition operation for two general relations $R_1$ and $R_2$.

**Definition 3.10.** The *composition* $R_1 \otimes R_2$ of two general relations $R_1$ and $R_2$ is the most specific relation R such that:

$(\forall h_1, h_2, h_3)\ ((h_1, R_1, h_2) \wedge (h_2, R_2, h_3) \Rightarrow (h_1, R, h_3))$ , where

$R_1 \otimes R_2 \;=\; \bigcup\limits_{\substack{r_1 \in R_1 \\ r_2 \in R_2}} r_1 \otimes r_2$ , being $r_1$ and $r_2$ atomic relationships.

Using this definition we have to construct the composition tables for each combination of the three types of features (points, lines, and areas) considered in the algebra. Therefore, if we consider all possibilities of three features named $h_1$, $h_2$, and $h_3$ being a point-like feature, a linear feature, or an areal feature, we would need 27 ($3^3$) tables. However, we construct only 18 of these tables from which the other 9 can be obtained. The 18 tables to be constructed split into 6 for $h_2$=point-like feature, 6 for $h_2$=linear feature and 6 for $h_2$=areal feature: when feature h1 is of type X, feature $h_2$ of type Y, and feature $h_3$ of type Z, with X, Y, and Z belonging to {P, L, A}, the corresponding composition table will be referred to as the XYZ table. In tables 3.2, 3.3, and 3.4 we show the tables constructed and their numbers of entries.

| TABLE | NUMBER OF ENTRIES |
|---|---|
| PPP table | \|PP-U\| x \|PP-U\|= 4 |
| PPL table | \|PP-U\| x \|PL-U\|= 6 |
| PPA table | \|PP-U\| x \|PA-U\|= 6 |
| LPL table | \|LP-U\| x \|PL-U\|= 9 |
| LPA table | \|LP-U\| x \|PA-U\|= 9 |
| APA table | \|AP-U\| x \|PA-U\|= 9 |
| TOTAL NUMBER OF ENTRIES: | **43** |

***Table 3.2.*** *Number of entries of the constructed tables for h$_2$=point-like entity*

| TABLE | NUMBER OF ENTRIES |
|---|---|
| PLP table | \|PL-U\| x \|LP-U\|= 9 |
| PLL table | \|PL-U\| x \|LL-U\|= 27 |
| PLA table | \|PL-U\| x \|LA-U\|= 15 |
| LLL table | \|LL-U\| x \|LL-U\|= 81 |
| LLA table | \|LL-U\| x \|LA-U\|=45 |
| ALA table | \|AL-U\| x \|LA-U\|=25 |
| TOTAL NUMBER OF ENTRIES: | **202** |

***Table 3.3.*** *Number of entries of the constructed tables for h2=linear entity*

| TABLE | NUMBER OF ENTRIES |
|---|---|
| PAP table | \|PA-U\| x \|AP-U\|= 9 |
| PAL table | \|PA-U\| x \|AL-U\|=15 |
| PAA table | \|PA-U\| x \|AA-U\|=24 |
| LAL table | \|LA-U\| x \|AL-U\|=25 |
| LAA table | \|LA-U\| x \|AA-U\|=40 |
| AAA table | \|AA-U\| x \|AA-U\|=64 |
| TOTAL NUMBER OF ENTRIES: | **177** |

***Table 3.4***. *Number of entries of the constructed tables for h$_2$=areal entity*

Let us consider the case h$_2$=linear entity. The six tables to be constructed for this case are the PLP, PLL, PLA, LLL, LLA, and ALA tables. From these six tables, we can get the other three, namely the LLP, ALP, and ALL tables. We illustrate this by showing how to get the $r_1 \otimes r_2$ entry of the LLP table from the PLL table. This means that we have to find the most specific relation R such that for any two linear features L$_1$ and L$_2$, and any point-like feature P, if (L$_1$, r$_1$, L$_2$) and (L$_2$, r$_2$, P) then (L$_1$, R, P). We can represent this as:

$$\begin{tikzcd} L_1 \arrow[r, "r_1"] \arrow[dr, "R"'] & L_2 \arrow[d, "r_2"] \\ & P \end{tikzcd}$$

From the converse table we can get the converses $r_1^{\cup}$ and $r_2^{\cup}$ of $r_1$ and $r_2$, respectively. The converse $R^{\cup}$ of R is clearly the composition $r_2^{\cup}\otimes r_1^{\cup}$ of $r_2^{\cup}$ and $r_1^{\cup}$, which can be obtained from the PLL table:

$$\begin{tikzcd} L_1 & L_2 \arrow[l, "r1^{\cup}"'] \\ & P \arrow[u, "r2^{\cup}"'] \arrow[ul, "R^{\cup}"] \end{tikzcd}$$

Now R is the converse of $R^{\cup}$: $R = (R^{\cup})^{\cup}$.

Below we present the composition tables, in which the relation *touch* is denoted by T, *cross* by C, *overlap* by O, *disjoint* by D, *completely-inside* by CI, *touching-from-inside* by TFI, *equal* by E, *completely-inside$_i$* by CIi, and *touching-from-inside$_i$* by TFIi.

| $r_2$<br>$r_1$ | **E** | **D** |
|---|---|---|
| **E** | E | D |
| **D** | D | {E, D} |

***Table 3.5.*** *The PPP composition table*

| $r_2$<br>$r_1$ | **T** | **D** | **CI** |
|---|---|---|---|
| **E** | T | D | CI |
| **D** | PL-U | PL-U | PL-U |

***Table 3.6.*** *The PPL composition table*

| $r_2$<br>$r_1$ | **T** | **D** | **CI** |
|---|---|---|---|
| **E** | T | D | CI |
| **D** | PA-U | PA-U | PA-U |

***Table 3.7.*** *The PPA composition table*

| r$_2$<br>r$_1$ | **T** | **D** | **CI** |
|---|---|---|---|
| **T** | {T, C, TFI} | {T, C, D} | {C, TFI, CI} |
| **D** | LA-U | LA-U | LA-U |
| **CI$_i$** | {T, C} | {T, C, D} | {C, CI, TFI} |

***Table 3.8.*** *The LPA composition table*

| r$_2$<br>r$_1$ | **T** | **D** | **CI** |
|---|---|---|---|
| **T** | {T, O, C, E, TFI, TFI$_i$} | {T, D, O, C , TFI$_i$, CI$_i$} | {T, O, C, TFI, CI} |
| **D** | {T, D, O, C, CI, TFI} | LL- U | {T, D, O, C, TFI, CI} |
| **CI$_i$** | {T, O, C, TFI$_i$, CI$_i$ } | {T, D, O, C, TFI$_i$, CI$_i$} | {O, C, E, TFI, CI, TFI$_i$, CI$_i$} |

***Table 3.9.*** *The LPL composition table*

| r$_2$<br>r$_1$ | **T** | **D** | **CI** |
|---|---|---|---|
| **T** | {T, O, E, TFI, TFI$_i$} | {T, O, D, CI$_i$, TFI$_i$} | {O, TFI, CI} |
| **D** | {T, O, D, TFI, CI} | AA-U | {T, O, D, CI, TFI} |
| **CI$_i$** | {O, TFI$_i$, CI$_i$} | {T, O, D, CI$_i$, TFI$_i$} | {O, E, TFI, CI, TFI$_i$, CI$_i$} |

***Table 3.10.*** *The APA composition table*

| r$_2$<br>r$_1$ | **T** | **D** | **CI$_i$** |
|---|---|---|---|
| **T** | PP-U | D | D |
| **D** | D | PP-U | D |
| **CI** | D | D | PP-U |

***Table 3.11.*** *The PLP composition table*

| r$_2$<br>r$_1$ | **T** | **C** | **D** | **CI** | **TFI** |
|---|---|---|---|---|---|
| **T** | {T, D} | PA-U | D | CI | {T, CI} |
| **D** | PA-U | PA-U | PA-U | PA-U | PA-U |
| **CI** | {T, D} | PA-U | D | CI | {T, CI} |

***Table 3.12****. The PLA composition table*

| r$_2$<br>r$_1$ | **T** | **C** | **O** | **D** | **E** | **CI** | **TFI** | **CI$_i$** | **TFI$_i$** |
|---|---|---|---|---|---|---|---|---|---|
| **T** | PL-U | PL-U | PL-U | D | T | CI | {T, D} | D | {T, D} |
| **D** | PL-U | PL-U | PL-U | PL-U | D | PL-U | PL-U | D | D |
| **CI** | {T, D} | PL-U | PL-U | D | CI | CI | CI | PL-U | PL-U |

***Table 3.13.*** *The PLL composition table*

| $r_2$ $r_1$ | **T** | **C** | **D** | **TFI** | **CI** |
|---|---|---|---|---|---|
| **T** | {T, C, D, TFI} | LA-U | {T, C, D} | {T, C, TFI, CI} | {C, TFI, CI} |
| **D** | LA-U | LA-U | LA-U | LA-U | LA-U |
| **O** | {T, C, D} | LA-U | {T, C, D} | {C, TFI, CI} | {C, TFI, CI} |
| **C** | {T, C, D} | LA-U | {T, C, D} | {C, TFI, CI} | {C, TFI, CI} |
| **E** | T | C | D | TFI | CI |
| **TFI** | {T, D} | LA-U | D | {TFI, CI} | CI |
| **CI** | {T, D} | LA-U | D | {TFI, CI} | CI |
| **TFI$_i$** | {T, C} | C | {T, C, D} | {TFI, CI} | {C, TFI, CI} |
| **CI$_i$** | {T, C} | C | {T, C, D} | {TFI, CI} | {C, TFI, CI} |

***Table 3.14.*** *The LLA composition table*

| $r_2$ $r_1$ | **T** | **C** | **D** | **CI** | **TFI** |
|---|---|---|---|---|---|
| **T** | AA-U\{CI, CI$_i$} | AA-U\{E, CI, CI$_i$} | AA-U\{E, CI, TFI} | AA-U\{T, D, E, CI$_i$, TFI$_i$} | AA-U\{D, E, CI$_i$, TFI$_i$} |
| **C** | AA-U\{E, CI, CI$_I$} | AA-U | AA-U\{E, CI, TFI} | {O, TFI, CI} | {O, TFI, CI} |
| **D** | AA-U\{E, CI$_i$, TFI$_i$} | AA-U\{E, CI$_i$, TFI$_i$} | AA-U | AA-U\{E, CI$_i$, TFI$_i$} | AA-U\{E, CI$_i$, TFI$_i$} |
| **CI$_i$** | AA-U\{T, D, E, CI, TFI} | {O, TFI$_i$, CI$_i$} | AA-U\{E, CI, TFI} | AA-U\{T, D} | {O,TFI$_i$, CI$_i$} |
| **TFI$_i$** | AA-U\{D, E, CI, TFI} | {O, TFI$_i$, CI$_i$} | AA-U\{E, CI, TFI} | {O, TFI, CI} | {O, E, TFI, TFI$_i$} |

***Table 3.15.*** *The ALA composition table*

| $r_2$ $r_1$ | **T** | **C** | **O** | **D** | **E** | **CI** | **TFI** | **CI$_i$** | **TFI$_i$** |
|---|---|---|---|---|---|---|---|---|---|
| **T** | LL-U\ {CI$_i$} | {T, C, O, D, TFI, CI} | {T, C, O, D, TFI, CI} | {T, C, O, D, TFI$_i$, CI$_i$} | T | {T, C, O, TFI, CI} | {T, C, O, TFI, CI} | D | {T, D} |
| **C** | {T, C, O, D, TFI$_i$, CI$_i$} | LL-U | LL-U\ {E, CI$_i$, TFI$_i$} | LL-U\ {E, CI, TFI} | C | {C, O, TFI, CI} | {C, O, TFI, CI} | {T, C, D} | {T, C, D} |
| **O** | {T, C, O, D, TFI$_i$, CI$_i$} | LL-U\ {E, CI, TFI} | LL-U | LL-U\ {E, CI, TFI} | O | {O, TFI, CI} | {O, TFI, CI} | {T, O, D, TFI$_i$, CI$_i$} | {T, O, D, TFI$_i$, CI$_i$} |
| **D** | {T, C, O, D, TFI, CI} | LL-U\ {E, CI$_i$, TFI$_i$} | LL-U\ {E, CI$_i$, TFI$_i$} | LL-U | D | LL-U \{E, CI$_i$, TFI$_i$} | LL-U \{E, CI$_i$, TFI$_i$} | D | D |
| **E** | T | C | O | D | E | CI | TFI | CI$_i$ | TFI$_i$ |
| **CI** | D | {T, C, D} | {T, O, D, TFI, CI} | D | CI | CI | CI | LL-U \{C} | {T, O, D, TFI, CI} |
| **TFI** | {T, D} | {T, C, D} | {T, O, D, TFI, CI} | D | TFI | LL-U \{C, E} | {TFI, CI} | {T, O, D, TFI$_i$, CI$_i$} | LL-U \{C, CI, CI$_i$} |
| **CI$_i$** | {T, C, O, TFI$_i$, CI$_i$} | {C, O, TFI$_I$, CI$_i$} | {O, TFI$_i$, CI$_i$} | LL-U\ {E, CI, TFI} | CI$_i$ | LL-U\{ T, C, D} | {O, TFI$_I$, CI$_i$} | CI$_i$ | CI$_i$ |
| **TFI$_i$** | {T, C, O, TFI$_i$, CI$_i$} | {C, O, TFI$_i$, CI$_i$} | {O, TFI$_i$, CI$_i$} | LL-U\ {E, CI, TFI} | TFI$_i$ | {O, TFI, CI} | {O, E, TFI, TFI$_i$} | CI$_i$ | {CI, CI$_i$} |

***Table 3.16.*** *The LLL composition table*

| $r_2$<br>$r_1$ | **T** | **D** | **CI$_i$** |
|---|---|---|---|
| **T** | PP-U | D | D |
| **D** | D | PP-U | D |
| **CI** | D | D | PP-U |

***Table 3.17.*** *The PAP composition table*

| $r_2$<br>$r_1$ | **T** | **C** | **D** | **CI$_i$** | **TFI$_i$** |
|---|---|---|---|---|---|
| **T** | PL-U | PL-U | D | D | {T, D} |
| **D** | PL-U | PL-U | PL-U | D | D |
| **CI** | D | PL-U | D | PL-U | PL-U |

***Table 3.18.*** *The PAL composition table*

| $r_2$<br>$r_1$ | **T** | **O** | **D** | **E** | **CI** | **TFI** | **CI$_i$** | **TFI$_i$** |
|---|---|---|---|---|---|---|---|---|
| **T** | {T, D} | PA-U | D | T | CI | {T, CI} | D | {T, D} |
| **D** | PA-U | PA-U | PA-U | D | PA-U | PA-U | D | D |
| **CI** | D | PA-U | D | CI | CI | CI | PA-U | PA-U |

***Table 3.19.*** *The PAA composition table*

| $r_2$<br>$r_1$ | **T** | **C** | **D** | **CI$_i$** | **TFI$_i$** |
|---|---|---|---|---|---|
| **T** | LL-U | {T, D, C, O, TFI, CI} | {T, D, C, O, TFI$_i$, CI$_i$} | D | {T, D, O} |
| **C** | {T, D, C, O, TFI$_i$, CIi} | LL-U | {T, D, C, O, TFI$_i$, CI$_i$} | {T, D, C, O, TFI$_i$, CI$_i$} | {T, D, C, O, TFI$_i$, CI$_i$} |
| **D** | {T, D, C, O, TFI, CI} | {T, D, C, O, TFI, CI} | LL-U | D | D |
| **CI** | D | {T, D, C, O, TFI, CI} | D | LL-U | {T, D, C, O, TFI, CI} |
| **TFI** | {T, D, O} | {T, D, C, O, TFI, CI} | D | {T, D, C, O, TFI$_i$, CI$_i$ } | {T, D, C, O, E, TFI, TFI$_i$} |

***Table 3.20.*** *The LAL composition table*

| $r_2$<br>$r_1$ | **T** | **O** | **D** | **E** | **CI** | **TFI** | **CI$_i$** | **TFI$_i$** |
|---|---|---|---|---|---|---|---|---|
| **T** | {T, D, C, TFI} | LA-U | {T, D, C} | T | {C, CI, TFI} | {T, C, CI, TFI} | D | {T, D} |
| **C** | {T, D, C} | LA-U | {T, D, C} | C | {C, CI, TFI} | {C, CI, TFI} | {T, C, D} | {T, C, D} |
| **D** | LA-U | LA-U | LA-U | D | LA-U | LA-U | D | D |
| **CI** | D | LA-U | D | CI | CI | CI | LA-U | LA-U |
| **TFI** | {T, D} | LA-U | D | TFI | CI | {CI, TFI} | {T, C, D} | {C, T, D, TFI} |

***Table 3.21.*** *The LAA composition table*

| $r_2$ $r_1$ | **T** | **O** | **D** | **E** | **CI** | **TFI** | **CI$_i$** | **TFI$_i$** |
|---|---|---|---|---|---|---|---|---|
| T | {T, D, O, E, TFI, TFI$_i$} | {T, D, O, TFI, CI} | {T, D, O, TFI$_i$, CI$_i$} | T | {O, CI, TFI} | {O, T, CI, TFI} | D | {D, T} |
| **O** | {T, D, O, TFI$_i$, CI$_i$} | AA-U | {T, D, O, TFI$_i$, CI$_i$} | O | {O, CI, TFI} | {O, TFI, CI} | {T, D, O, TFI$_i$, CI$_i$} | {T, D, O, TFI$_i$, CI$_i$} |
| **D** | {T, D, O, TFI, CI} | {T, D, O, TFI, CI} | AA-U | D | {T, D, O, TFI, CI} | {T, D, O, TFI, CI} | D | D |
| **E** | T | O | D | E | CI | TFI | CI$_i$ | TFI$_i$ |
| **CI** | D | {T, D, O, TFI, CI} | D | CI | CI | CI | AA-U | {T, D, O, CI, TFI} |
| **TFI** | {T, D} | {T, D, O, CI, TFI} | D | TFI | CI | {CI, TFI} | {T, D, O, TFI$_i$, CI$_I$} | {T, D, O, E, TFI$_i$, TFI} |
| **CI$_I$** | {O, CI$_i$, TFI$_i$} | {O, CI$_i$, TFI$_i$} | {T, D, O, CI$_i$, TFI$_i$} | CI$_i$ | {E, O, CI, TFI, CI$_i$, TFI$_i$} | {O, TFI$_i$, CI$_i$} | CI$_i$ | CI$_i$ |
| **TFI$_i$** | {T, O, CI$_i$, TFI$_i$} | {O, CI$_i$, TFI$_I$} | {T, D, O, CI$_i$, TFI$_i$} | TFI$_i$ | {O, CI, TFI} | {O, E, TFI, TFI$_i$} | CI$_i$ | {TFI$_i$, CI$_i$} |

***Table 3.22.*** *The AAA composition table*

Using these 27 composition tables plus the converse table defined in section 3.1, we can implement the BSIP. We have to realise that the composition tables proposed are *complete*, that is, we can infer the composition between any combination of point, line and area features. The result of the inference is always one of the topological relations or a disjunction of them.

To implement the BSIP, the composition tables and converse table of the topological calculus are defined as facts of a PROLOG database, for example the fact composition_table_PLA(t,tfi,[t,ci]) means that being h$_1$ a point, h$_2$ a line and h$_3$ an area, and knowing that (h$_1$,touch,h$_2$) and (h$_2$,touch_form_inside,h$_3$), then h$_1$ and h$_3$ can hold one of the relations given between brackets, i. e. (h$_1$,touch,h$_3$) or (h$_1$,completely inside,h$_3$). Figures 3.8 to 3.11 show the converse table and some examples of how the composition tables explicitly constructed have been implemented in our approach as facts of our PROLOG database. Figure 3.12 shows how the rest of tables not explicitly constructed are implemented in our PROLOG database.

```
conv(o,[o]).
conv(t,[t]).
conv(c,[c]).
conv(d,[d]).
conv(ci,[cii]).
conv(tfi,[tfii]).
conv(cii,[ci]).
conv(tfii,[tfi]).
conv(e,[e]).
```

***Figure 3.8.*** *PPP-Table*

```
composition_table_PPL(e,t,[t]).
composition_table_PPL(e,d,[d]).
composition_table_PPL(e,ci,[ci]).
composition_table_PPL(d,t,[t,d,ci]).
composition_table_PPL(d,d,[t,d,ci]).
composition_table_PPL(d,ci,[t,d,ci]).
```

***Figure 3.9.*** *PPL-Table*

```
composition_table_PPA(e,t,[t]).
composition_table_PPA(e,d,[d]).
composition_table_PPA(e,ci,[ci]).
composition_table_PPA(d,t,[t,d,ci]).
composition_table_PPA(d,d,[t,d,ci]).
composition_table_PPA(d,ci,[t,d,ci]).
```

***Figure 3.10.*** *PPA-table.*

```
composition_table_LPA(t,t,[t,c,tfi]).
composition_table_LPA(t,d,[t,c,d]).
composition_table_LPA(t,ci,[c,tfi,ci]).
composition_table_LPA(d,t,[t,c,d,tfi,ci]).
composition_table_LPA(d,d,[t,c,d,tfi,ci]).
composition_table_LPA(d,ci,[t,c,d,tfi,ci]).
composition_table_LPA(cii,t,[t,c]).
composition_table_LPA(cii,d,[t,c,d]).
composition_table_LPA(cii,ci,[c,ci,tfi]).
```

***Figure 3.11.*** *LPA-Table*

```
/********* LPP composition table **************/
composition_table_LPP(R1,R2,Rsdo):-              conv(R1,R11),
conv(R2,R22),            composition_table_PPL(R22,R11,Rconv),
conv_op(Rconv,Rsdo).
/********* APP composition table **************/
composition_table_APP(R1,R2,Rsdo):-              conv(R1,R11),
conv(R2,R22),            composition_table_PPA(R22,R11,Rconv),
conv_op(Rconv,Rsdo).
/************* APL composition table ****************/
composition_table_APL(R1,R2,Rsdo):-              conv(R1,R11),
conv(R2,R22),            composition_table_LPA(R22,R11,Rconv),
conv_op(Rconv,Rsdo).
/*********** LLP composition table *******************/
composition_table_LLP(R1,R2,Rsdo):-              conv(R1,R11),
conv(R2,R22),            composition_table_PLL(R22,R11,Rconv),
conv_op(Rconv,Rsdo).
/*************** ALP composition table***************/
composition_table_ALP(R1,R2,Rsdo):-              conv(R1,R11),
conv(R2,R22),            composition_table_PLA(R22,R11,Rconv),
conv_op(Rconv,Rsdo).
/*********** ALL composition table ******************/
composition_table_ALL(R1,R2,Rsdo):-              conv(R1,R11),
conv(R2,R22),            composition_table_LLA(R22,R11,Rconv),
conv_op(Rconv,Rsdo).
/**************** LAP composition table ************/
composition_table_LAP(R1,R2,Rsdo):-              conv(R1,R11),
conv(R2,R22),            composition_table_PAL(R22,R11,Rconv),
conv_op(Rconv,Rsdo).
/************* AAP composition table *************/
composition_table_AAP(R1,R2,Rsdo):-              conv(R1,R11),
conv(R2,R22),            composition_table_PAA(R22,R11,Rconv),
conv_op(Rconv,Rsdo).
/********* AAL composition table *******************/
composition_table_AAL(R1,R2,Rsdo):-              conv(R1,R11),
conv(R2,R22),            composition_table_LAA(R22,R11,Rconv),
conv_op(Rconv,Rsdo).
```

***Figure 3.12. Part I**. PROLOG Implementation of the tables not explicitly constructed.*

```
/*******Auxiliar Prédicats *************/
conv_op(R1,R2):- conv_op(R1,[],R2).
conv_op([],R,R).
convoy([R1|R2],R3,R4):-   conv(R1,R11), append(R11,R3,R31),
conv_op(R2,R31,R4).
```

***Figure 3.12. Part II.*** *PROLOG Implementation of the tables not explicitly constructed**.***

The definition of the BSIP is the second step defined in section 1.4 in order to allow reasoning about the topological information represented by the algebra developed. In fact, reasoning about knowledge expressed in the presented calculus can be done using a constraint propagation algorithm similar to the one in ([Allen 83]), guided by the 18 composition tables and the converse table. Such an algorithm has the advantage of being incremental: knowledge may be added without having to revise the processing steps achieved so far. Moreover, this allows the integration of topology with other spatial aspects in the same model as it has been done with the concepts of orientation, cardinal directions, and absolute and relative distances, thanks to considering the representation and the reasoning process of each aspect as an instance of the Constraint Satisfaction Problem (CSP) as it has been done in [Escrig and Toledo 1998, 2001]. Therefore, the last step to allow reasoning about topological knowledge is to define the Full Inference Process (FIP).

## 3.3. THE FULL INFERENCE PROCESS FOR TOPOLOGICAL RELATIONS.

For computing the full inference process (FIP) of topological information we consider that:

1. each topological relationship between two objects is seen as a constraint;

2. the set of topological relationships forms a constraint graph, where the nodes are spatial objects (points, lines and areas) and the arcs are the binary constraints between objects. This constraint graph is not complete at the beginning, that is, all the nodes are not bi-directional connected, because there is no initial topological relationship between all the objects in the space;

3. the fact of propagating the constraints for making explicit the topological relationships between all the nodes in the graph, is seen as an instance of the CSP.

The formula (1), defined in section 1.4, which approximated the solution for temporal objects, is rewritten for topological relations between spatial objects as follows:

$$c_{a,c} \coloneqq c_{a,c} \oplus c_{a,b} \otimes c_{b,c} \qquad (2)$$

In our approach, the constraint $c_{a,b}$ (which represents the topological relationship holding between objects a, b) is represented by the predicate ctr_comp_top(TB,TA,A,B,Rel), where A and B are the spatial objects which holds the set of atomic topological relationships included in the set Rel; TB and TA represents the types of the objects A and B, which can be point (p), line (l) or area (a).

The intersection ('$\oplus$') and composition ('$\otimes$') parts of formula (2) are implemented with simplification and propagation CHRs (see section 1.4 for details), respectively.

The part of the intersection ($c_{a,b}\oplus$....) is implemented by the following simplification CHR:

*ctr_comp_top(TB,TA,B,A,R1),ctr_comp_top(TB,TA,B,A,R2) <=> intersection(R1,R2,R3)|ctr_comp_top(TB,TA,B,A,R3).*

For supplying the lack of completeness of the constraint graph (because there is not a topological relation between every object in the graph), two CHRs more are defined, by applying the converse operation to the first and second constraints, respectively.

*ctr_comp_top(TB,TA,B,A,R1), ctr_comp_top(TA,TB,A,B,R2) <=> conv_op(R2,R22), intersection(R1,R22,R3) | ctr_comp_top(TB,TA,B,A,R3).*

*ctr_comp_top(TA,TB,A,B,R1), ctr_comp_top(TB,TA,B,A,R2) <=> conv_op(R1,R11), intersection(R1,R11,R3) | ctr_comp_top(TB,TA,B,A,R3).*

The part of the basic operation (2) related with the composition ($c_{a,b} \otimes c_{b,c}$) corresponds to the BSIP defined in the previous section. It is implemented by propagation CHRs:

*ctr_comp_top(TB,TA,B,A,R1), ctr_comp_top(TC,TB,C,B,R2) ==> composition(R1,R2,R3) | newctr_comp_top(TC,TA,C,A,R3).*

where composition/3 refers to the set of facts of the PROLOG database which defines the composition operation between 3 spatial objects.

As before, for the case in which the constraint graph is not complete, two other CHRs are defined by applying the converse operation to the first and second constraints.

*ctr_comp_top(TA,TB,A,B,R1), ctr_comp_top(TC,TB,C,B,R2) ==> ( conv_op(R1,R11), composition(TA,TB,TC,R11,R2,R3) | ctr_comp_top(TC,TA,C,A,R3).*

*ctr_comp_top(TB,TA,B,A,R1), ctr_comp_top(TB,TC,B,C,R2) ==> (conv_op(R2,R22), composition(TA,TB,TC,R1,R22,R3), | ctr_comp_top(TC,TA,C,A,R3).*

*ctr_comp_top(TA,TB,A,B,R1), ctr_comp_top(TB,TC,B,C,R2) ==>(conv_op(R1,R11), conv_op(R2,R22), composition(TA,TB,TC,R11,R22,R3), | ctr_comp_top(TC,TA,C,A,R3).*

In order to test the integration of the topological relations with other spatial aspects in the same model, as has been done with the rest of spatial aspects, a PROLOG algorithm has been implemented to infer new spatial information from a given set of spatial relations. Therefore, as the BSIP, the FIP has been implemented in a PROLOG program that can work together the PROLOG program already defined in [Escrig and Toledo 1998, 2001]. Bellow we show this algorithm (algorithm 3.1).

```
% Constraint declarations and definitions
(3.1a) constraints (ctr_comp_top)/5, (ctr_comp_top)/7.
(3.1b) label_with ctr_comp_top(N,TB,TA,B,A,Rel,I) if N>1.
(3.1c) ctr_comp_top(N,TB,TA,B,A,Rel,I):-
member(R,Rel),ctr_comp_top(1,TB,TA,B,A,[R],I).

%Initialize
(3.2) ctr_comp_top(TB,TA,B,A,Rel) <=>
length(Rel,N),ctr_comp_top(N,TB,TA,B,A,Rel,1).

%Special cases
(3.3a) ctr_comp_top(N,TB,TA,B,A,Rel,I) <=> empty(Rel) | false.
(3.3b) ctr_comp_top(N,TA,TA,A,A,Rel,I) <=> true.
(3.3c) ctr_comp_top(N,p,p,A,B,Rel,I) <=> N=2 | true.
(3.3d) ctr_comp_top(N,p,l,A,B,Rel,I) <=> N=3 | true.
(3.3e) ctr_comp_top(N,p,a,A,B,Rel,I) <=> N=3 | true.
(3.3f) ctr_comp_top(N,l,p,A,B,Rel,I) <=> N=3 | true.
(3.3g) ctr_comp_top(N,l,l,A,B,Rel,I) <=> N=9 | true.
(3.3h) ctr_comp_top(N,l,a,A,B,Rel,I) <=> N=5 | true.
(3.3i) ctr_comp_top(N,a,p,A,B,Rel,I) <=> N=3 | true.
(3.3j) ctr_comp_top(N,a,l,A,B,Rel,I) <=> N=5 | true.
(3.3k) ctr_comp_top(N,a,a,A,B,Rel,I) <=> N=8 | true.
```

***Algorithm 3.1. Part I.*** *Path consistency algorithm to propagate compositions of disjunctive qualitative orientation relationships.*

```
%Intersections
(3.4a) ctr_comp_top(N1,TB,TA,B,A,R1,I),
ctr_comp_top(N2,TB,TA,B,A,R2,J) <=>
intersection(R1,R2,R3), length(R3,N3), K is min(I,J)+1 |
ctr_comp_top(N3,TB,TA,B,A,R3,K).

(3.4b) ctr_comp_top(N1,TB,TA,B,A,R1,I),
ctr_comp_top(N2,TA,TB,A,B,R2,J) <=>
conv_op(R2,R22), intersection(R1,R22,R3), length(R3,N3), K is
min(I,J)+1 |
ctr_comp_top(N3,TB,TA,B,A,R3,K).

(3.4c) ctr_comp_top(N1,TA,TB,A,B,R1,I),
ctr_comp_top(N2,TB,TA,B,A,R2,J) <=>
conv_op(R1,R11), intersection(R1,R11,R3), length(R3,N3), K is
min(I,J)+1 |
ctr_comp_top(N3,TB,TA,B,A,R3,K).

%Compositions
(3.5a) ctr_comp_top(N1,TB,TA,B,A,R1,I),
ctr_comp_top(N2,TC,TB,C,B,R2,J) ==>
((I=1,J<6);(J=1;I<6)), composition_op(TA,TB,TC,R1,R2,R3),
length(R3,N3), K is I+J |
ctr_comp_top(N3,TC,TA,C,A,R3,K).

(3.5b) ctr_comp_top(N1,TA,TB,A,B,R1,I),
ctr_comp_top(N2,TC,TB,C,B,R2,J) ==>
((I=1,J<6);(J=1,I<6)), singleton(R1), conv_op(R1,R11),
composition_op(TA,TB,TC,R11,R2,R3), length(R3,N3),
K is I+J | ctr_comp_top(N3,TC,TA,C,A,R3,K).

(3.5c) ctr_comp_top(N1,TB,TA,B,A,R1,I),
ctr_comp_top(N2,TB,TC,B,C,R2,J) ==>
((I=1,J<6);(J=1,I<6)), singleton(R2), conv_op(R2,R22),
composition_op(TA,TB,TC,R1,R22,R3),
length(R3,N3), K is I+J | ctr_comp_top(N3,TC,TA,C,A,R3,K).

(3.5d) ctr_comp_top(N1,T,ATB,A,B,R1,I),
ctr_comp_top(N2,TB,TC,B,C,R2,J) ==>
((I=1,J<6);(J=1,I<6)), singleton(R1), singleton(R2),
conv_op(R1,R11), conv_op(R2,R22),
composition_op(TA,TB,TC,R11,R22,R3),
length(R3,N3), K is I+J | ctr_comp_top(N3,TC,TA,C,A,R3,K).
%Auxiliary predicates
singleton([_]).
empty([]).
choose([R],[R|_]).

/*****************************************************************
***      PREDICATES TO DO COMPOSITION OF DISJUNCTIVE
RELATIONSHIPS       *********************************************/
composition_op(TA,TB,TC,R1,R2,R3):-
(  (singleton(R1),singleton(R2)) ->
    composition_simple(TA,TB,TC,R1,R2,R3);
    composition1_op(TA,TB,TC,R1,R2,[],R3)).

composition_simple(TA,TB,TC,[R1],[R2],R3):-
composition(TA,TB,TC,R1,R2,R3).
```

***Algorithm 3.1. Part II.*** *Path consistency algorithm to propagate compositions of disjunctive qualitative orientation relationships.*

```
composition(p,p,p,R1,R2,R3):- composition_table_PPP(R1,R2,R3).
composition(p,p,l,R1,R2,R3):- composition_table_PPL(R1,R2,R3).
composition(p,p,a,R1,R2,R3):- composition_table_PPA(R1,R2,R3).
composition(l,p,p,R1,R2,R3):- composition_table_LPP(R1,R2,R3).
composition(l,p,l,R1,R2,R3):- composition_table_LPL(R1,R2,R3).
composition(l,p,a,R1,R2,R3):- composition_table_LPA(R1,R2,R3).
composition(a,p,p,R1,R2,R3):- composition_table_APP(R1,R2,R3).
composition(a,p,l,R1,R2,R3):- composition_table_APL(R1,R2,R3).
composition(a,p,a,R1,R2,R3):- composition_table_APA(R1,R2,R3).

composition(p,l,p,R1,R2,R3):- composition_table_PLP(R1,R2,R3).
composition(p,l,l,R1,R2,R3):- composition_table_PLL(R1,R2,R3).
composition(p,l,a,R1,R2,R3):- composition_table_PLA(R1,R2,R3).
composition(l,l,p,R1,R2,R3):- composition_table_LLP(R1,R2,R3).
composition(l,l,l,R1,R2,R3):- composition_table_LLL(R1,R2,R3).
composition(l,l,a,R1,R2,R3):- composition_table_LLA(R1,R2,R3).
composition(a,l,p,R1,R2,R3):- composition_table_ALP(R1,R2,R3).
composition(a,l,l,R1,R2,R3):- composition_table_ALL(R1,R2,R3).
composition(a,l,a,R1,R2,R3):- composition_table_ALA(R1,R2,R3).

composition(p,a,p,R1,R2,R3):- composition_table_PAP(R1,R2,R3).
composition(p,a,l,R1,R2,R3):- composition_table_PAL(R1,R2,R3).
composition(p,a,a,R1,R2,R3):- composition_table_PAA(R1,R2,R3).
composition(l,a,p,R1,R2,R3):- composition_table_LAP(R1,R2,R3).
composition(l,a,l,R1,R2,R3):- composition_table_LAL(R1,R2,R3).
composition(l,a,a,R1,R2,R3):- composition_table_LAA(R1,R2,R3).
composition(a,a,p,R1,R2,R3):- composition_table_AAP(R1,R2,R3).
composition(a,a,l,R1,R2,R3):- composition_table_AAL(R1,R2,R3).
composition(a,a,a,R1,R2,R3):- composition_table_AAA(R1,R2,R3).

composition1_op(A,B,C,[],_,R3,R3).
composition1_op(A,B,C,[R1|R11],R2,Raux,Rdo):-
   composition1(A,B,C,R1,R2,R_parcial),
   union(Raux,R_parcial,R),
   composition1_op(A,B,C,R11,R2,R,Rdo).

composition1(A,B,C,R1,R2,R):-
   composition11(A,B,C,R1,R2,[],R).

composition11(A,B,C,R1,[],Rdo,Rdo).
composition11(A,B,C,R1,[R21|R22],Raux,Rdo):-
   composition(A,B,C,R1,R21,R3),
   union(Raux,R3,R),
   composition11(A,B,C,R1,R22,R,Rdo).

/* The following file contains the converse table for
topological information as facts of the PROLOG database*/
:-['inversas.pl'].
/* The following file contains the composition tables for
topological information as facts of the PROLOG database*/
:-['composit.pl'].
```

***Algorithm 3.1. Part III.*** *Path consistency algorithm to propagate compositions of disjunctive qualitative orientation relationships.*

This algorithm is based on the algorithm developed in [Escrig and Toledo 98, 01] for integrating other spatial aspects such as orientation, cardinal and absolute directions and distances. Escrig and Toledo's algorithm uses the optimisation introduced in the algorithm [Mackworth and Freuder 85] named PC-2. This optimisation is based on the idea that the constraint $c_{a,b}$ can be computed as the converse $c_{b,a}$ if it is needed (by applying the converse operation to the corresponding relationship), which saves half of the computation. Therefore, this optimisation is also included in our algorithm. In algorithm 3.1 no queue of modified constraints is needed because the new constraint goal itself will trigger new applications of the propagation CHRs.

Two predicates, ctr_comp_top of arity 5 and 7, are declared in rule(3.1a). Predicates ctr_comp_top/5 are the kind of constraints introduced initially as topological information. The predicates of type ctr_comp_top/5 are translated into the predicates ctr_comp_top/7 by rule (3.2) where length (N) of the relationship is added as well as the length of the shortest path (I) from which the constraint is derived. A length of the shortest path (I) equal to 1 means that the constraint is direct, that is, it is user-defined, not obtained from derivation. Both arguments are included to increase efficiency. The first one will avoid compositions between constraints which do not give more information (rules 3.3c to 3.3k) because all the topological atomic relationships (they will form a different set of constraints depending on the types of the objects related) are included in the disjunction. The last argument is used to restrict the propagation CHRs to involve at least one direct constraint and to avoid that a new constraint exactly equal to other one already existent, will trigger again a composition rule causing an infinite execution. Therefore in order to guarantee the termination we use the last argument to involve at least one direct constraint and the other constraint has to be of a length of the shortest path (I) (path from which it has been derived) equal to a value that we specify in the algorithm. This value is established depending on the quantity of information we want to infer, when it is bigger, we can get more new information but the propagation will be longer.

It should also be observed that termination is guaranteed because the simplification rules replace R1 and R2 by the result of intersecting R1 and R2 which is R3 (and R3 is the same as R1 or R2 or smaller), and because propagation CHRs are never repeated for the same constraint goals more than twice.

Following the explanation of the algorithm 3.1, the constraints will be treated by the CLP clause (3.1c) if the relation, Rel¸ represents a disjunction of primitive relationships. In predicate (3.1c), member(R,Rel) non-deterministically chooses one primitive constraint, R, from the disjunctive constraint Rel which implements the backtrack search part of the algorithm. Special cases are simplification CHRs. (3.3a) detects inconsistent constraints. When the constraints relate three spatial objects with an empty relationship, the constraint in substituted by the built-in predicate *false* and the full predicate fails. (3.3b) deletes constraints which contains only one region which is related with itself.

Simplification CHRs (3.4a) to (3.4c) perform intersections which permit the simplification of redundant information. Rule (3.4a) implements intersection in the same way as it is originally defined, that is, given two constraints that relate the same two spatial objects, the more restricted relationship between both constraints is calculated by the predicate *intersection (R1,R2,R3)* and these constraints are substituted by a new one which relates the same two objects with the new relationship R3 among them.

By applying the converse operation to the first (rule 3.4b) or second constraint (rule 3.4c) of the two which are in the head of the original intersection rule, it is possible to obtain the topological information among the same two objects. Therefore it is possible to calculate the intersection if the converse operation is applied to the relationship or disjunction of relationships in the guard part of the rules. This is possible because we have to notice that the application of the converse operation to a disjunction of relationships is equivalent to the application of these operations to each relationship included in the disjunction of relations as it has been shown in section 3.1

Propagation CHRs (3.5a) to (3.5d) perform compositions. (3.5a) implements the composition as originally defined previously in this section. In a similar way to that which it happens to the simplification rule, the application of the converse operation to the first constraint, to the second one or to both constraints of the two which define the head of the original composition rule define the CHRs (3.5b) and (3.5d) respectively.

In CHRs (3.5b) and (3.5d) another optimisation is introduced. It consists of restricting one of the two constraints involved in the propagation to be disjunction-free by adding to the guard a check which guarantees that its corresponding relationship is singleton. This not only reduces the average size of the resulting constraint but also makes composition more efficient.

Therefore, a total of 3 simplification rules and 4 propagation rules define the algorithm which solves the FIP for topological relations.

## 3.4. EXAMPLE USING TOPOLOGICAL INFORMATION WITH CONSTRAINTS HANDLING RULES.

The topological information between the entities involved in the example of figure 3.13 might be expressed in natural language such that:

*"We are interested in knowing all the topological relations between five elements, which are a park, named Ribalta Park, a public building named Pergola, a shop, a track and the Bus Station. We know that the Pergola is in the middle of the Ribalta Park, that the track is disjoint of the Ribalta Park but that it touches the shop and crosses the Bus Station".*

Depending on the granularity we choose we can represent each one of this objects as a point, a line or an area. We have chosen for this example the next distribution: the Ribalta Park and the Bus Station due its dimensions compared with the other elements are considered as areas, the track is considered as a line and the shop and the Pergola are considered as points.

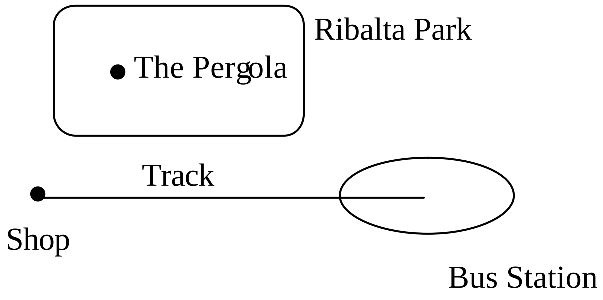


***Figure 3.13.*** *Graphic representation of the elements of this example.*

Therefore we can represent the relations that we know using the topological algebra developed by the next constraints:

- ctr_comp_top(a,p,Ribalta,Pergola,[ci]),
- ctr_comp_top(a,l,Ribalta,Track,[d]),
- ctr_comp_top(p,l,Shop,Track,[t]),
- ctr_comp_top(l,a,Track,Buspark,[c]);

In these constraints *ci*, *d*, *t* and *c* represents the topological relations completely inside, disjoint, touch and cross respectively and the symbols *a*, *p* and *l* represent the types of the elements in the constraints. If these constraints are provided as entry to the previous defined CSP for topological information, all the information which can be inferred from the original relationships has been obtained, which are related below. The information inferred is obtained with path length of limit equal to 3 for the inference process.

1. ctr_comp_top(1,l,p,Track,Pergola,[d],6)
2. ctr_comp_top(1,a,p,Ribalta,Shop,[d],6)
3. ctr_comp_top(1,a,l,Ribalta,Track,[d],6)
4. ctr_comp_top(1,p,p,Shop,Pergola,[d],4)

The first argument of the predicate in the inferred constraints corresponds to the argument of primitive relationships in the disjunctive relations. The last number corresponds to the length of the shortest path from which the constraint was derived.

This is a simple example to show that the algorithm obtains the desired results, and the results are consistent, that is, the results are correctly calculated. If we use a bigger limit of the path length for the inference process we will obtain more topological relations between the five entities of the example in figure 3.13.

# CHAPTER 4

## INTEGRATION OF TOPOLOGY AND TIME: THE MOTION MODEL.

Our aim in this chapter is to formalize the intuitive notion of spatio-temporal continuity for a qualitative theory of motion. Motion can be seen as a form of spatio-temporal change, the chapter presents a qualitative representation model for integrating qualitative time and topological information for modelling motion and reasoning about dynamic worlds in which spatial relations between regions may change with time. Therefore, the integration of the concept of time will allow us to represent and reason about topological changes and not only about snapshots of a changing world. It is therefore important to develop a model which combines space and time in an integrated fashion.

Assuming that change is continuous, as is the case in standard qualitative reasoning, it is important to know which qualitative values or relations, are neighbours in the sense that if a value or predicate holds at one time, then there is some continuous change possible such that the next value or predicate to hold will be a neighbour. Continuity networks defining such as neighbours are often called *conceptual neighbourhoods* in the literature following the use of the term defined by Freksa [Freksa 92] to describe the Allen's 13 JEPD (jointly exhaustive and pair wise disjoint) temporal relations [Allen 83] according to their conceptual closeness or closest topological distance (e.g. *meets* is a neighbour of both *overlaps* and *before*) as it has been seen in chapter 1.2. Therefore, we are going to define the conceptual neighbourhoods of the topological algebra presented in chapter 3 in section 4.3.

Moreover, as we also want to integrate the motion model with other spatial aspects, we are going to follow once more the three steps defined in section 1.4 in order to allow the integration, which are;

- To define the representational model for the motion model (section 4.2), for which we need to define a suitable time algebra (section 4.1) ;
- To define the BSIP for motion (section 4.3);
- And to define the FIP (sections 4.4).

To define the representational model, we have developed an algebra which integrates topological information and time. The topological information integrated is the one presented in chapter 3. But, we also have to develop a time algebra suitable for that integration. This time algebra is presented in section 4.1.

## 4.1. TIME ALGEBRA.

We define a time algebra in which variables represent time points. There are five primitive constraints: prev, next, <<, >>, ==. These primitive constraints are defined as follows:

**Definition 4.1.** Given two time points, t and t', t **==** t' iff it has not occurred a topological change between t and t' (or between t' and t) on any relation.

**Definition 4.2.** Given two time points, t and t', t' **next** t iff t' > t and some topological relation or relations have changed to a neighbour relation between t and t'.

**Definition 4.3**. Given two time points, t and t', t' **prev** t iff t' < t and some topological relation or relations have changed to a neighbour relation between t and t'.

**Definition 4.4.** Given two time points, t and t', t' **>>** t iff t' > t and a topological relation has changed strictly more than once to a neighbour relation.

**Definition 4.5.** Given two time points, t and t', t' **<<** t iff t' < t and a topological relation has changed strictly more than once to a neighbour relation.

According to these definitions, time is represented by disjunctive binary constraints of the form X{$r_1$, ..., $r_n$}Y, where each $r_i$ is a relation that is applicable to X and Y. X{$r_1$, ..., $r_n$}Y is a disjunction of the way (X $r_1$ Y) $\vee$ .... $\vee$ (X $r_n$ Y) and $r_i$ is also called primitive constraints.

We have chosen to represent time as time points because we are only interested in the point of time in which one region is transformed into its topological neighbourhood, and this will occur just in one point of time.

**Definition 4.6.** A **general relation** R of the calculus is any subset of the set of all atomic relations.

Given the original relationship (X r Y), by permuting X and Y we can obtain one relationship more, which is the *converse* operation. Below we give the formal definition of the *converse* operation.

**Definition 4.7.** The **converse** of a general relation R, called $R^{\cup}$ is defined as:

$$\forall(X,Y)\ ((X,R,Y) \Leftrightarrow (Y,R^{\cup},X)) \qquad (1)$$

In this definition $R^{\cup}$ is:

$$R^{\cup} = \bigcup_{r \in R_1} \{r^{\cup}\}$$

Table 4.1shows the converse operation for the time algebra.

| **r** | $\mathbf{r}^{E}$ |
|---|---|
| == | == |
| << | >> |
| >> | >> |
| next | prev |
| prev | next |

***Table 4.1*** *The converse table for the time algebra.*

### 4.1.1 The Basic Step of the Inference Process for Time Relations.

The Basic Step of the Inference Process (BSIP) for time information consists in *"given two time relationships between three objects in the space, ($h_1$, $r_1$, $h_2$) and ($h_2$, $r_2$, $h_3$), we want to infer the relationship ($h_1$, $r_3$, $h_3$)"*. To infer such relationship we need to define and then use the composition operation for two general relations $R_1$ and $R_2$.

**Definition 4.8.** The **composition** R1 ⊗ R2 of two general relations R1 and R2 is the most specific relation R such that:

$$\forall (h1,h2,h3)\ ((h1,R1,h2) \wedge (h2,R2,h3) \Rightarrow (h1,R,h3)) \qquad (2)$$

Note that for general relations $R_1$ and $R_2$ we have:

$$R_1 \otimes R_2 = \bigcup_{\substack{r_1 \in R_1 \\ r_2 \in R_2}} r_1 \otimes r_2$$

The composition table for time relations is shown in table 4.2.

| $r_1$ / $r_2$ | << | **prev** | == | **next** | >> |
|---|---|---|---|---|---|
| << | {<<} | {<<} | {<<} | {prev,<<} | {<<,prev,==,next,>>} |
| prev | {<<} | {<<,prev} | {prev} | {==,prev,next} | {next,>>} |
| == | {<<} | {prev} | {==} | {next} | {>>} |
| next | {<<,prev} | {prev,==,next} | {next} | {>>,next} | {>>} |
| >> | {<<,prev,==,next,>>} | {>>,next} | {>>} | {>>} | {>>} |

***Table 4.2.*** *The composition table for the time algebra.*

If we want to reason only about time information then we should define the FIP for time information, but since we want to reason about motion information as the integration of topology and time information we have only defined the aspects of the time algebra needed for this integration. Section 4.2 to 4.4 presents the motion model one more following the three steps defined in section 1.4: the representational model, the BSIP and the FIP.

## 4.2. THE REPRESENTATIONAL MODEL OF MOTION.

The representational model of topology and qualitative time points follows the formalism used by Allen for temporal interval algebra [Allen 83]. The Allen style formalism will provide to our approach the possibility of reasoning about topology in dynamic worlds by applying the Allen's constraint propagation algorithm.

In the representational model for motion, binary relations between two objects, which can be points, lines or areas, called $h_1$ and $h_2$, in a point of time t are defined as tertiary constraints or propositions where the topological relation r between $h_1$ and $h_2$ in the point of time t is denoted by $(h_1,r,h_2)_t$. That is, we define a general relation on motion and the corresponding converse operation such as:

**Definition 4.9.** A **general relation R** of the algebra during **time t** is defined as:

$$\forall(h_1,h_2)\ ((h_1,R,h_2)_t \Leftrightarrow \cup_{r\in R}\ (h_1,r,h_2)_t) \qquad (3)$$

**Definition 4.10.** The **converse of a general relation R in time t**, denoted as $R^{\cup}$, is defined as follows:

$$\forall(h_1,h_2)\ ((h_1,R,h_2)_t \Leftrightarrow (Y,R^{\cup},X)_t) \qquad (4)$$

From this definition we observe that the converse of the integration of topology and time algebra, which means the converse of the motion algebra is the same as the converse defined only for topological relations because the converse is calculated in the same point of time, that is, time does not affect the converse operation. Therefore the converse table is the same as the one explained in chapter 3.1.

## 4.3. THE BASIC STEP OF THE INFERENCE PROCESS (BSIP) ON MOTION.

The BSIP for motion consists of: *"given three objects A, B, C, if the topological relationships in time t between A and B and B and C are known, it is possible to obtain the topological relationship in time between objects A and C".* To infer such topological relationship in time t we are going to define the composition operation for two general relations R1 and R2.

We can distinguish four possible cases for solving the BSIP of the model which integrates topology and time:

**Definition 4.11.** The resulting general relation R obtained from the **composition** (⊗) operation could be calculated such as:

a) Composition of three regions in the same point of time :

$$(A,R1,B)t_0 \otimes (B,R2,C)t_0 \Rightarrow (A,R,C)t_0$$

b) Composition implementing Freksa's conceptual neighbourhood :

$$(A,R1,B)t_0 \otimes (t_0, \text{Reltime}, t_1) \Rightarrow (A,R,B)t_1$$

c) Composition of three regions in different points of time:

$$(A,R1,B)t_0 \otimes (B,R2,C)t_1 \,/\, (t_0, \text{Reltime}, t_1) \Rightarrow ((A,R1,B)t_0 \otimes (t_0, \text{Reltime}, t_1)) \otimes$$
$$(B,R2,C)t_1 \Rightarrow (A,R',B)t_1 \otimes (B,R2,C)t_1 \Rightarrow (A,R,C)t_1$$

d) Composition between the same regions in the same point of time.

$$(A,R1,B)_{t0} \otimes (A,R1,B)_{t0} \Rightarrow (A,R1,B)_{t0}$$

### 4.3.1 Composition of three regions in the same point of time.

The composition of three regions in the same point of time, shown in definition 4.11.a is the composition of the topological relations between three regions A, B and C, in the same point of time, where A, B, C belong to {point, line, area} is the usual topological composition, where time has no effect. To calculate this composition we will use the 18 composition tables and the converse table defined for the topological algebra in chapter 3.

### 4.3.2 Composition implementing Freksa's conceptual neighbourhood.

The second type of composition, in definition 4.11.b, is the composition which implements Freksa's conceptual neighbourhood notion [Freksa 91]. It looks for the possible topological relations which will appear between two regions as time changes. To reason about this type we need to construct 6 composition tables that will be referred to as XYt–table where the regions X and Y belong to {point (P), line (L), area (A)} and t represents the time dimension of the algebra. We would need 9 composition tables if we consider all possibilities with X and Y being a point-like, a linear or an areal entity. However, we construct only 6 tables from which the other three tables can be obtained, by using the converse operation. We construct the AAt-table, LAt-table, PPt-table, LLt-table, PLt-table and PAt-table. These tables are depicted in tables 4.3 to 4.8, respectively. We have depicted in the same column both "next" and "prev" cases, and "<<" and ">>" cases because the corresponding entries are the same.

| **Rtime**<br>**RTop** | **next** or **prev** | << or >> | == |
|---|---|---|---|
| T | {D,O,T} | {T,E,TFI,C,CIi,TFIi} | {T} |
| O | {T,TFI,E,O,TFIi} | {O,D,CI,CIi} | {O} |
| D | {T,D} | {D,O,E,TFI,CI,TFIi,CIi,TFIi} | {D} |
| E | {O,E} | {E,T,D,TFI,CI, TFIi,CIi} | {E} |
| TFI | {O,CI,TFI} | {TFI,T,D,E,CI, TFIi, CIi} | {TFI} |
| CI | {TFI,CI} | {CI,T,O,D,E,TFIi,CIi} | {CI} |
| TFIi | {O,CIi,TFi} | {TFIi,T,D,E,CI,TFI} | {TFIi} |
| CIi | {TFIi,CIi} | {CIi,T,O,D,E,TFI,CI} | {CIi} |

***Table 4.3.*** *$AA_t$-table.*

| **Reltime**<br>**Reltop** | **next** or **prev** | << or >> | == |
|---|---|---|---|
| T | {C,D,T} | {T,TFI,CI} | {T} |
| C | {T,TFI,C} | {C,D,CI} | {C} |
| D | {T,D} | {D,C,TFI,CI} | {D} |
| TFI | {C,CI,TFI} | {TFI,T,D} | {TFI} |
| CI | {TFI,CI} | {CI,T,C,D} | {CI} |

***Table 4.4.*** *$LA_t$-table.*

| **Reltime**<br>**Reltop** | **next** or **prev** | << or >> | == |
|---|---|---|---|
| E | {D,E} | {E} | {E} |
| D | {E,D} | {D} | {D} |

***Table 4.5.*** *$PP_t$-table.*

| **Reltime**<br>**Reltop** | **Next or prev** | **<< or >>** | **==** |
|---|---|---|---|
| T | {D,O,C,T, TFI, CI, TFIi,CIi} | {T,E} | {T} |
| D | {T,D} | {D,C,O,E,TFI,CI, TFIi,CIi} | {D} |
| O | {T,E,TFI, TFIi, O} | {O,C,D,CI,CIi} | {O} |
| C | {T,C} | {D,C,O,E,TFI,CI, TFIi, CIi} | {C} |
| E | {O,E} | {T,E,D,C,TFI,CI, TFIi,CIi} | {E} |
| TFI | {CI,T,O,TFI} | {C, D,TFI, E, TFIi,CIi} | {TFI} |
| CI | {TFI, T,CI} | {CI,C,D,O,E, TFIi,CIi} | {CI} |
| TFIi | {T,O,CIi,TFi} | {TFIi,D,C,E, TFI,CI} | {TFIi} |
| CIi | {C,TFIi,CIi} | {CIi,T,D,O,E, TFI,CI} | {CIi} |

***Table 4.6.*** *$LL_t$-table.*

| **Reltime**<br>**Reltop** | **next** or **prev** | << or >> | == |
|---|---|---|---|
| T | {D,CI,T} | {T} | {T} |
| D | {T,CI,D} | {D} | {D} |
| CI | {T,DCI} | {CI} | {CI} |

***Table 4.7.*** *$PL_t$-table.*

| **Reltime**<br>**Reltop** | **next** or **prev** | << or >> | == |
|---|---|---|---|
| T | {D,CI,T} | {T} | {T} |
| D | {T,D} | {D,CI} | {D} |
| CI | {T,CI} | {CI,D} | {CI} |

***Table 4.8.*** *$PA_t$-table.*

Notice that the "=" time relation is the identity.

As a relation t prev t' corresponds to a change of some topological relation to a neighbour relation, the tables always keep the possibility that a relation has not changed between time t and t'. This situation model the fact that the time changes from t to t' because another topological relationship has changed and the relationship between X and Y (RelTop) has not changed.

The other three tables which are not constructed can be obtained by applying the converse operation to the ones which are constructed. For example, the ALt-table can be obtained using the LAt-table and the converse operation. This means that we have to find the most specific relation R such that, if X and Y are an area and a linear entity respectively:

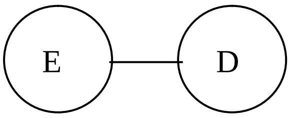
$$(X, \text{Reltop}, Y)t_0 \otimes (t_0, \text{Reltime}, t_1) \Rightarrow (X, R, Y)t_1 \quad (5)$$

From the LAt-table and using the converse operation we will get the relation R as follows:

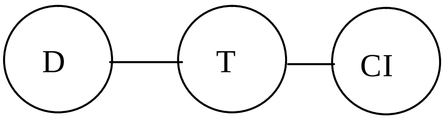
$$(Y, \text{Reltop}^{\cup}, X)t_0 \otimes (t_1, \text{Reltime}^{\cup}, t_0) \Rightarrow (Y, R', X)t_0 \qquad (6)$$

Then the relation R that we are looking for is $R=(R')^{\cup}$, which means that R is the converse of R' relation.

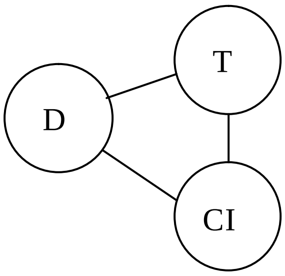
As we have already mentioned, these tables implement Freksa's notion of conceptual neighbourhood. Therefore using them, we can obtain the conceptual neighbourhood graph for each Universal Set of the topological calculus (described in chapter 3), which are shown in figure 4.1, in which the relation *touch* is denoted by T, *cross* by C, *overlap* by O, *disjoint* by D, *completely-inside* by CI, *touching-from-inside* by TFI, *equal* by E, *completely-inside*$_i$ by CIi, and *touching-from-inside*$_i$ by TFIi. In these graphs each relation in a node is the conceptual neighbour of the nodes connected to it by arcs. Then, the topological neighbourhood of a region is that region to which the original region can be transformed to by a process of gradual, continuous change which does not involve passage through any third region.



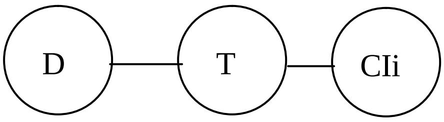
a) PP-U Conceptual Neighbourhood.



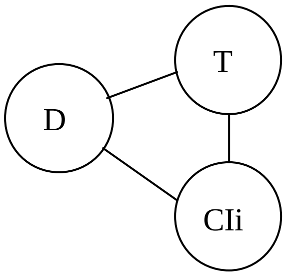
b) PA-U Conceptual Neighbourhood.



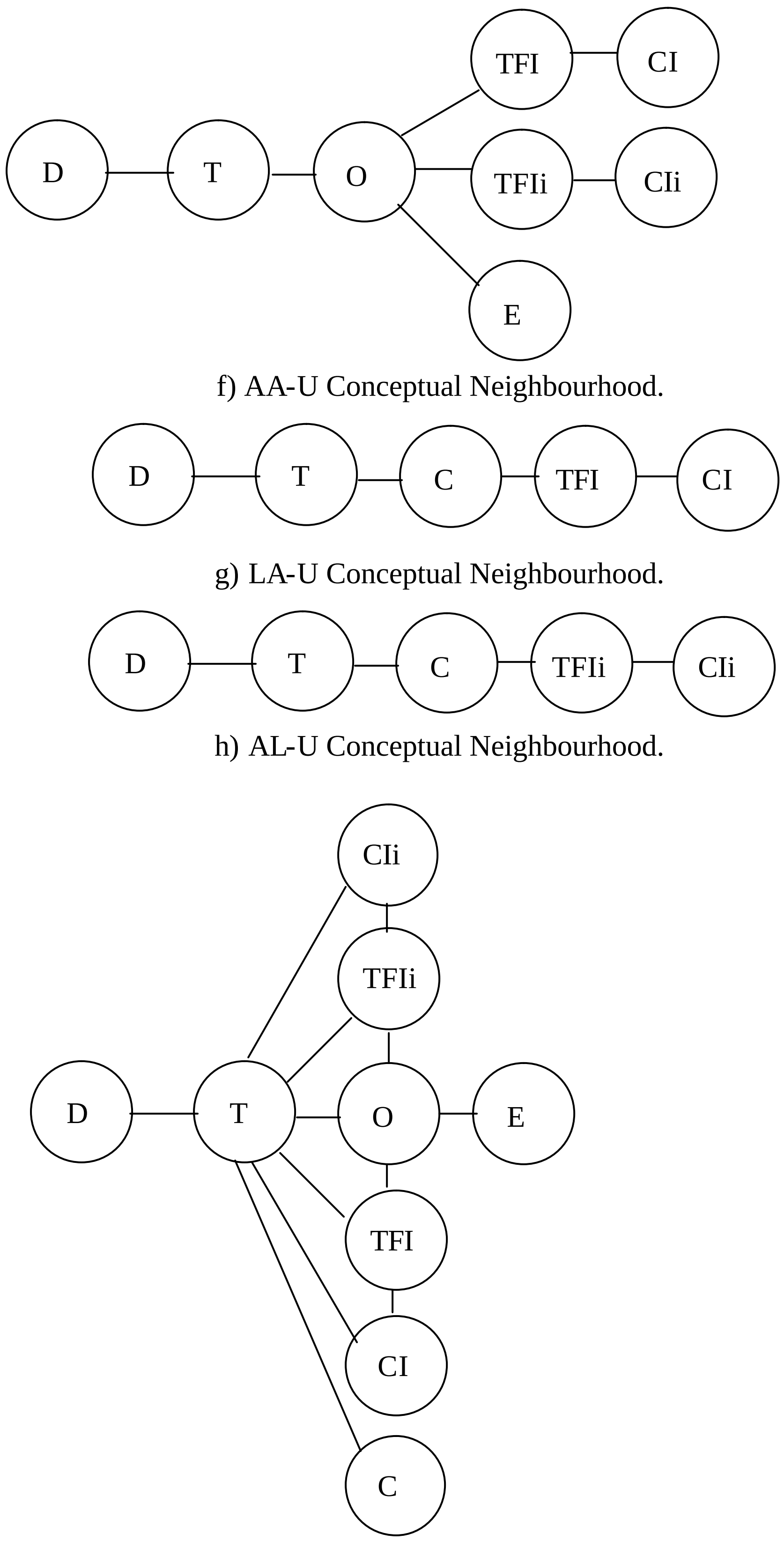
c) PL-U Conceptual Neighbourhood.



d) AP-U Conceptual Neighbourhood.



e) LP-U Conceptual Neighbourhood.



f) AA-U Conceptual Neighbourhood.

g) LA-U Conceptual Neighbourhood.

h) AL-U Conceptual Neighbourhood.

i) LL-U Conceptual Neighbourhood.

***Figure 4.1.*** *Conceptual Neighbourhoods Graphs for each Universal set of topological relations.*

### 4.3.3 Composition of three regions in different points of time.

The third case of composition is the composition of three regions in different points of time defined in definition 4.11.c as:

$$(A,R1,B)t_0 \otimes (B,R2,C)t_1 \,/\, (t_0, \text{Reltime}, t_1) \Rightarrow ((A,R1,B)t_0 \otimes (t_0, \text{Reltime}, t_1)) \otimes$$
$$(B,R2,C)t_1 \Rightarrow (A,R',B)t_1 \otimes (B,R2,C)t_1 \Rightarrow (A,R,C)t_1$$

In this case, we want to infer the composition R in time t1 between three regions, X, Y and Z having the topological relation in time $t_0$ between X and Y, the topological relation in time $t_1$ between Y and Z and the qualitative time relation between times $t_0$ and $t_1$. In order to get the composition relation R, first of all we have to obtain the topological relations that can appear between X and Y in time $t_1$ using the composition tables defined for the case of (4.11.b) above described. Therefore we have the general relation R' which appears between X and Y in $t_1$. This relation together with the general relation R2 between Y and Z in $t_1$ corresponds to the case (4.11.a).

### 4.3.4 Composition between the same regions in the same point of time.

The last composition case is the composition between the same regions in the same point of time, shown in definition 4.11.d. It is the composition between the same two regions in the same point of time, sharing the relation R1, and the result is the relation R1 itself. We do not infer more information from this case.

As we have done with the topological algebra presented in chapter 3, in order to test the integration of the motion algebra presented with other spatial aspects in the same model, a PROLOG algorithm has been implemented to infer new spatio-temporal information from a given set of spatio-temporal relations. Therefore the BSIP and the FIP have been implemented in a PROLOG program that can works together the PROLOG program already defined in [Escrig and Toledo 1998, 2001].

### 4.3.5 The Implementation of the BSIP for the Qualitative Motion Model

To implement the BSIP, the composition tables and converse table of the time algebra developed together with the converse and composition tables of the motion model are implemented as facts of our PROLOG database, for example the fact composition_table_time(<<,next,[prev,<<]) means that having three points of time $t_1$, $t_2$ and $t_3$, and knowing that ($t_1$,<<,$t_2$) and ($t_2$,next,$t_3$) then $t_1$ and $t_3$ can hold one of the time relations given between brackets, i.e. ($t_1$, prev, $t_3$) or ($t_1$,<<,$t_3$). For the case of the motion algebra, the fact composition_table_AAt(t,next,[d,o,t]) means that being two entities $h_1$ and $h_2$ areal entities, and having two points of time $t_1$ and $t_2$, knowing that in the point of time $t_1$ ($h_1$,touch,$h_2$)$_{t1}$, and that ($t_1$, next, $t_2$), then in the point of time $t_2$, $h_1$ and $h_2$ can hold one of the topological relations given between brackets, i.e. ($h_1$, disjoint, $h_2$)$_{t2}$, ($h_1$, overlap, $h_2$)$_{t2}$, or ($h_1$, touch, $h_2$)$_{t2}$.

Figures 4.2 to 4.3 show the converse table and composition table respectively of the time algebra as they have been implemented in our PROLOG database.

```
conv_time(==,[==]).
conv_time(<<,[<<]).
conv_time(>>,[>>]).
conv_time(next,[next]).
conv_time(prev,[prev]).
```

***Figure 4.2.*** *Converse table for the Time algebra as PROLOG facts.*

```
composition_table_time(<<,<<,[<<]).
composition_table_time(<<,prev,[<<]).
composition_table_time(<<,==,[<<]).
composition_table_time(<<,next,[prev,<<]).
composition_table_time(<<,>>,[<<,prev,next,==,>>]).
composition_table_time(prev,<<,[<<]).
composition_table_time(prev,prev,[<<,prev]).
composition_table_time(prev,==,[prev]).
composition_table_time(prev,next,[==,next,prev]).
composition_table_time(prev,>>,[next,>>]).
composition_table_time(==,<<,[<<]).
composition_table_time(==,prev,[prev]).
composition_table_time(==,==,[==]).
composition_table_time(==,next,[next]).
composition_table_time(==,>>,[>>]).
composition_table_time(next,<<,[<<,prev]).
composition_table_time(next,prev,[next,==,prev]).
composition_table_time(next,==,[next]).
composition_table_time(next,next,[>>,next]).
composition_table_time(next,>>,[>>]).
composition_table_time(>>,<<,[<<,prev,==]).
composition_table_time(>>,prev,[>>,next]).
composition_table_time(>>,==,[>>]).
composition_table_time(>>,next,[>>]).
composition_table_time(>>,>>,[>>]).
```

***Figure 4.3.*** *Composition table for the Time Algebra as PROLOG facts.*

Figures 4.4, 4.5 and 4.6 shows some examples of how the composition tables explicitly constructed for the motion algebra have been implemented in our approach as facts of the PROLOG database. Figure 4.7 shows how the rest of tables not explicitly constructed are implemented in our PROLOG database. We use the same notation as in chapter 3.

```
composition_table_AAt(t,next,[d,o,t]).
composition_table_AAt(t,prev,[d,o,t]).
composition_table_AAt(t,<<,[t,e,tfi,c,ci,tfii,cii]).
composition_table_AAt(t,>>,[t,e,tfi,c,ci,tfii,cii]).
composition_table_AAt(t,==,[t]).
composition_table_AAt(o,next,[t,tfi,e,tfii,o]).
composition_table_AAt(o,prev,[ t,tfi,e,tfii,o]).
composition_table_AAt(o,<<,[o,d,ci,cii]).
composition_table_AAt(o,>>,[o,d,ci,cii]).
composition_table_AAt(o,==,[o]).
composition_table_AAt(d,next,[t,d]).
composition_table_AAt(d,prev,[t,d]).
composition_table_AAt(d,<<,[d,o,e,tfi,ci,tfii,cii]).
composition_table_AAt(d,>>,[d,o,e,tfi,ci,tfii,cii]).
composition_table_AAt(d,==,[d]).
composition_table_AAt(e,next,[o,e]).
composition_table_AAt(e,prev,[o,e]).
composition_table_AAt(e,<<,[e,t,d,tfi,ci,tfii,cii]).
composition_table_AAt(e,>>,[e,t,d,tfi,ci,tfii,cii]).
composition_table_AAt(e,==,[e]).
composition_table_AAt(tfi,next,[o,ci,tfi]).
composition_table_AAt(tfi,prev,[o,ci,tfi]).
composition_table_AAt(tfi,<<,[tfi,t,d,e,ci,tfii,cii]).
composition_table_AAt(tfi,>>,[tfi,t,d,e,ci,tfii,cii]).
composition_table_AAt(tfi,==,[tfi]).
composition_table_AAt(ci,next,[tfi,ci]).
composition_table_AAt(ci,prev,[tfi,ci]).
composition_table_AAt(ci,<<,[ci,t,o,d,e,tfii,cii]).
composition_table_AAt(ci,>>,[ci,t,o,d,e,tfii,cii]).
composition_table_AAt(ci,==,[ci]).
composition_table_AAt(tfii,next,[o,cii,tfii]).
composition_table_AAt(tfii,prev,[o,cii,tfii]).
composition_table_AAt(tfii,<<,[tfii,t,d,e,ci,tfi]).
composition_table_AAt(tfii,>>,[tfii,t,d,e,ci,tfi]).
composition_table_AAt(tfii,==,[tfii]).
composition_table_AAt(cii,next,[tfii,cii]).
composition_table_AAt(cii,prev,[tfii,cii]).
composition_table_AAt(cii,<<,[cii,t,o,d,e,tfi,ci]).
composition_table_AAt(cii,>>,[cii,t,o,d,e,tfi,ci]).
composition_table_AAt(cii,==,[cii]).
```

***Figure 4.4. AAt-Table.***

```
composition_table_LAt(t,next,[c,d,t]).
composition_table_LAt(t,prev,[c,d,t]).
composition_table_LAt(t,<<,[t,tfi,ci]).
composition_table_LAt(t,>>,[t,tfi,ci]).
composition_table_LAt(t,==,[t]).
composition_table_LAt(c,next,[t,tfi,c]).
composition_table_LAt(c,prev,[t,tfi,c]).
composition_table_LAt(c,<<,[c,d,ci]).
composition_table_LAt(c,>>,[c,d,ci]).
composition_table_LAt(c,==,[c]).
composition_table_LAt(d,next,[t,d]).
composition_table_LAt(d,prev,[t,d]).
composition_table_LAt(d,<<,[d,c,tfi,ci]).
composition_table_LAt(d,>>,[d,c,tfi,ci]).
composition_table_LAt(d,==,[d]).
composition_table_LAt(tfi,next,[c,ci,tfi]).
composition_table_LAt(tfi,prev,[c,ci,tfi]).
composition_table_LAt(tfi,<<,[tfi,t,d]).
composition_table_LAt(tfi,>>,[tfi,t,d]).
composition_table_LAt(tfi,==,[tfi]).
composition_table_LAt(ci,next,[ci,tfi]).
composition_table_LAt(ci,prev,[ci,tfi]).
composition_table_LAt(ci,<<,[ci,t,c,d]).
composition_table_LAt(ci,>>,[ci,t,c,d]).
composition_table_LAt(ci,==,[ci]).
```

***Figure 4.5. LAt-table.***

```
composition_table_PAt(t,next,[d,ci,t]).
composition_table_PAt(t,prev,[d,ci,t]).
composition_table_PAt(t,<<,[t]).
composition_table_PAt(t,>>,[t]).
composition_table_PAt(t,==,[t]).
composition_table_PAt(d,next,[t,d]).
composition_table_PAt(d,prev,[t,d]).
composition_table_PAt(d,<<,[d,ci]).
composition_table_PAt(d,>>,[d.ci]).
composition_table_PAt(d,==,[d]).
composition_table_PAt(ci,next,[t,ci]).
composition_table_PAt(ci,prev,[t,ci]).
composition_table_PAt(ci,<<,[d,ci]).
composition_table_PAt(ci,>>,[d,ci]).
composition_table_PAt(ci,==,[ci]).
```

***Figure 4.6. PAt-table.***

```
/********* ALt composition table **************/
composition_table_ALt(R1,R2,Rsdo):-               conv(R1,R11),
conv_time(R2,R22),        composition_table_LAt(R11,R22,Rconv),
conv_time_op(Rconv,Rsdo).

/********* APt composition table **************/
composition_table_APt(R1,R2,Rsdo):-               conv(R1,R11),
conv_time(R2,R22),        composition_table_PAt(R11,R22,Rconv),
conv_time_op(Rconv,Rsdo).

/************* LPt composition table ****************/
composition_table_LPt(R1,R2,Rsdo):- conv(R1,R11),
conv_time(R2,R22), composition_table_PLt(R11,R22,Rconv),
conv_time_op(Rconv,Rsdo).

/*******Auxiliar Predicates *************/
conv_time_op(R1,R2):- conv_time_op(R1,[],R2).
conv_time_op([],R,R).
conv_time_op([R1|R2],R3,R4):-
   conv(R1,R11),
   append(R11,R3,R31),
  conv_time_op(R2,R31,R4).
```

***Figure 4.7.*** *PROLOG Implementation of the three tables not explicitly constructed.*

## 4.4. THE FULL INFERENCE PROCESS FOR MOTION.

In order to define a straightforward algorithm to solve the FIP, the concept of topology integrated with time is seen in our approach as an instance of the CSP.

The formula (1), defined in section 1.4, which approximated the solution for temporal objects, is rewritten for the motion model as follows:

**Case 1:** $c_{a,c,t} := c_{a,c,t} \oplus c_{a,b,t} \otimes c_{b,c,t}$ (7)

**Case 2:** $c_{a,b,t1} := c_{a,b,t1} \oplus c_{a,b,t0} \otimes c_{t0,t1}$ (8)

**Case 3:** $c_{a,b,t1} := c_{a,c,t0} \oplus c_{a,b,t0} \otimes c_{b,c,t1} / c_{t0,t1}$ (9)

Topology plus time relationships (named as $c_{a,b,t}$) are represented as tertiary constraints by the predicate ctr_comp_top_time(TB,TA,A,B,Rel,t), where A and B are the spatial objects which holds the set of atomic topological relationships included in the set Rel in the point of time t, TB and TA represents the types of the objects A and B, which can be point (p), line (l) or area (a). Time relationship ($c_{t0,t1}$) between points of time t0 and t1, (t0, Rtime,t1), is represented by the predicated ctr_comp_time(t0,t1,Rtime).

The path consistency algorithm is implemented with two kinds of CHRs as in the case of topological relations. The part of the intersection ($c_{a,b,t} \oplus$....) of formulas 7, 8, and 9 are implemented by a simplification CHR and the part of the composition ($c_{a,b,t} \otimes$ ) of formulas 7, 8, and 9 are implemented by propagation CHRs. Part of the FIP is presented in algorithm 4.1. Termination is guaranteed due to the simplification CHR and because propagation CHRs are never repeated for the same constraint goal.

The algorithm 4.1 is based on the algorithm 3.1 for topological relations explained in section 3.3. It also includes the optimisation introduced in [Escrig and Toledo 98, 01] which allows a constraint to be computed as its converse when it is needed. It saves half of the computation.

The simplification CHR used to implement the intersection of the three formulas (7), (8), and (9) is the next one:

*ctr_comp_top_time(TB,TA,B,A,R1,T), ctr_comp_top_time(TB,TA,B,A,R2,T) <=> intersection(R1,R2,R3) | ctr_comp_top_time(TB,TA,B,A,R3,T).*

For supplying the lack of completeness of the constraint graph (because there is not a topological relation in a point of time t between every object in the graph), two more CHRs are defined by applying the converse operation to the first and second constraints of the initial CHR, respectively:

*ctr_comp_top_time(TB,TA,B,A,R1,T), ctr_comp_top_time(TA,TB,A,B,R2,T) <=> conv_op(R2,R22), intersection(R1,R22,R3) | ctr_comp_top(N3,TB,TA,B,A,R3,T,K).*

*ctr_comp_top_time(TA,TB,A,B,R1,T), ctr_comp_top_time(TB,TA,B,A,R2,T) <=> conv_op(R1,R11), intersection(R1,R11,R3) | ctr_comp_top(N3,TB,TA,B,A,R3,T,K).*

In the case of propagation CHRs, we would need three different propagation CHRs for each case, this means that we would have a propagation CHRs for the formula (7), another for the formula (8) and finally another different one for the formula (9), which are the following ones respectively:

***%CASE A: (Formula (7))***

*ctr_comp_top_time(TB,TA,B,A,R1,T), ctr_comp_top_time(TC,TB,C,B,R2,T) ==>, composition_op(TA,TB,TC,R1,R2,R3) | ctr_comp_top_time(TC,TA,C,A,R3,T).*

***%CASE B: (Formula (8))***

```
ctr_comp_top_time(TB,TA,B,A,R1,T1), ctr_comp_time(T1,T2,R2) ==>
  composition_time_op(TA,TB,T1,T2,R1,R2,R3) |
  ctr_comp_top_time(TB,TA,B,A,R3,T2).
```

***%CASE C: (Formula (9))***

```
ctr_comp_top_time(TB,TA,B,A,R1,T1), ctr_comp_top_time(TC,TB,C,B,R2,T2),
  ctr_comp_time(T1,T2,R3) ==> composition_time_op(TA,TB,T1,T2,R1,R3,R4) |
  ctr_comp_top_time(TB,TA,B,A,R4,T2).
```

For each case of propagation CHRs, we need to define more CHRs by applying the converse operation to the first and second constraints, respectively. In the case of the composition (propagation CHRs) we need a different number of CHRs for each case to implement (A, B and C). Therefore, for the CHRs implementing the case A of the composition we will need 4 CHRs, the initial one and three more CHRs defined using the converse operation. For the case B, we need only one CHR more, because the converse is applied only to the second constraint. And finally for the case C we would need 8 CHRs, defining the seven CHRs which are not original by applying the converse operation to the 3 constraints in the head.

***%CASE A: (Formula (7))***

```
ctr_comp_top_time(TA,TB,A,B,R1,T), ctr_comp_top_time(TC,TB,C,B,R2,T) ==>
  conv_op(R1,R11), composition_op(TA,TB,TC,R11,R2,R3) |
  ctr_comp_top_time(TC,TA,C,A,R3,T).

ctr_comp_top_time(TB,TA,B,A,R1,T), ctr_comp_top(TB,TC,B,C,R2,T) ==>
  conv_op(R2,R22), composition_op(TA,TB,TC,R1,R22,R3) |
  ctr_comp_top_time(TC,TA,C,A,R3,T).

ctr_comp_top_time(TA,TB,A,B,R1,T), ctr_comp_top(TB,TC,B,C,R2,T) ==>
  conv_op(R1,R11), conv_op(R2,R22), composition_op(TA,TB,TC,R11,R22,R3) |
  ctr_comp_top_time(TC,TA,C,A,R3,T).
```

***%CASE B: (Formula (8))***

```
ctr_comp_top_time(N1,TB,TA,B,A,R1,T1,I), ctr_comp_time(N2,T2,T1,R2,J) ==>

conv_time_op(R2,R22), composition_time_op(TA,TB,T1,T2,R1,R22,R3) |
  ctr_comp_top_time(TC,TA,C,A,R3,T2), ctr_comp_time(T1,T2,R22).
```

```
%CASE C: (Formula (9))

ctr_comp_top_time(TB,TA,B,A,R1,T1), ctr_comp_top_time(TC,TB,C,B,R2,T2),
  ctr_comp_time(T2,T1,R3) ==> conv_time_op(R3,R33),
  composition_time_op(TA,TB,T1,T2,R1,R33,R4) |
  ctr_comp_top_time(,TB,TA,B,A,R4,T2), ctr_comp_time(T1,T2,R33).

ctr_comp_top_time(TA,TB,A,B,R1,T1), ctr_comp_top_time(TC,TB,C,B,R2,T2),
  ctr_comp_time(T1,T2,R3) ==> conv_op(R1,R11),
  composition_time_op(TA,TB,T1,T2,R11,R3,R4) |
  ctr_comp_top_time(TB,TA,B,A,R4,T2).

ctr_comp_top_time(TA,TB,A,B,R1,T1), ctr_comp_top_time(TC,TB,C,B,R2,T2),
  ctr_comp_time(T2,T1,R3) ==> conv_op(R1,R11), conv_time_op(R3,R33),
  composition_time_op(TA,TB,T1,T2,R11,R33,R4) |
  ctr_comp_top_time(TB,TA,B,A,R4,T2), ctr_comp_time(T1,T2,R33).

ctr_comp_top_time(TB,TA,B,A,R1,T1), ctr_comp_top_time(TB,TC,B,C,R2,T2),
  ctr_comp_time(T1,T2,R3) ==> composition_time_op(TA,TB,T1,T2,R1,R3,R4) |
  ctr_comp_top_time(TB,TA,B,A,R4,T2).

ctr_comp_top_time(TB,TA,B,A,R1,T1), ctr_comp_top_time(TB,TC,B,C,R2,T2),
  ctr_comp_time(T2,T1,R3) ==> conv_time_op(R3,R33),
  composition_time_op(TA,TB,T1,T2,R1,R33,R4) |
  ctr_comp_top_time(TB,TA,B,A,R4,T2), ctr_comp_time(T1,T2,R33).

ctr_comp_top_time(TA,TB,A,B,R1,T1), ctr_comp_top_time(TB,TC,B,C,R2,T2),
  ctr_comp_time(T1,T2,R3) ==> conv_op(R1,R11),
  composition_time_op(TA,TB,T1,T2,R11,R3,R4) |
  ctr_comp_top_time(TB,TA,B,A,R4,T2).

ctr_comp_top_time(TA,TB,A,B,R1,T1), ctr_comp_top_time(TB,TC,B,C,R2,T2),
  ctr_comp_time(T2,T1,R3) ==> conv_op(R1,R11), conv_time_op(R3,R33),
  composition_time_op(TA,TB,T1,T2,R11,R33,R4) |
  ctr_comp_top_time(TB,TA,B,A,R4,T2), ctr_comp_time(T1,T2,R33).
```

But the eight propagation CHRs described for implementing the case C (formula 9) are not needed explicitly, because using the constraints implemented for solving the cases A and B (formulas 7 and 8 respectively) we obtain the desired result. For instance imaging that we have the next three constraints: *ctr_comp_top_time(TA,TB,A,B,R1,T1), ctr_comp_top_time(TC,TB,C,B,R2,T2), ctr_comp_time(T2,T1,R3),* then the constraints *ctr_comp_top_time(TA,TB,A,B,R1,T1),*and *ctr_comp_time(T2,T1,R3)* will trigger the second propagation CHR defined for the case B, and we will obtain two new constraints which are: *ctr_comp_top_time(TA,TB,A,B,R4,T2)* and *ctr_comp_time(T1,T2,R33).*Then the constraints *ctr_comp_top_time(TA,TB,A,B,R4,T2)* (one of the new ones obtained) and *ctr_comp_top_time(TC,TB,C,B,R2,T2)* will trigger the second propagation CHR defined for the case A. Therefore, finally we obtain the relation we were looking for which is *ctr_comp_top_time(TC,TA,C,A,R5,T).* This is the reason why these eight CHRs are not implemented.

```
% Constraint declarations and definitions
(4.1a) constraints (ctr_comp_top_time)/6, (ctr_comp_top_time)/8.
(4.1b) label_with ctr_comp_top_time(N,TB,TA,B,A,Rel,T,I) if N>1.
(4.1c) ctr_comp_top_time(N,TB,TA,B,A,Rel,T,I):-
member(R,Rel),ctr_comp_top_time(1,TB,TA,B,A,[R],T,I).
(4.1d) constraints (ctr_comp_time)/3, (ctr_comp_time)/5.
(4.1e) label_with ctr_comp_time(N,T1,T2,Rel,I) if N>1.
(4.1f) ctr_comp_time(N,T1,T2,Rel,I):-
member(R,Rel),ctr_comp_time(1,T1,T2,[R],I).
%Initialize
(4.2a) ctr_comp_top_time(TB,TA,B,A,Rel,T)   <=>   length(Rel,N),
ctr_comp_top_time(N,TB,TA,B,A,Rel,T,1).
(4.2b) ctr_comp_time(T1,T2,Rel)        <=>        length(Rel,N),
ctr_comp_time(N,T1,T2,Rel,1).
% Special cases
(4.3a) ctr_comp_top_time(N,TB,TA,B,A,Rel,T,I) <=>
   empty(Rel)|false.
(4.3b) ctr_comp_top_time(N,TA,TA,A,A,Rel,T,I) <=> true.
(4.3c) ctr_comp_top_time(N,p,p,A,B,Rel,T,I) <=> N=2 | true.
(4.3d) ctr_comp_top_time(N,p,l,A,B,Rel,T,I) <=> N=3 | true.
(4.3e) ctr_comp_top_time(N,p,a,A,B,Rel,T,I) <=> N=3 | true.
(4.3f) ctr_comp_top_time(N,l,p,A,B,Rel,T,I) <=> N=3 | true.
(4.3g) ctr_comp_top_time(N,l,l,A,B,Rel,T,I) <=> N=9 | true.
```

***Algorithm 4.1 Part I.*** *Path consistency algorithm to propagate compositions of disjunctive motion (topology + time) relationships.*

```
(4.3h) ctr_comp_top_time(N,l,a,A,B,Rel,T,I) <=> N=5 | true.
(4.3i) ctr_comp_top_time(N,a,p,A,B,Rel,T,I) <=> N=3 | true.
(4.3j) ctr_comp_top_time(N,a,l,A,B,Rel,T,I) <=> N=5 | true.
(4.3k) ctr_comp_top_time(N,a,a,A,B,Rel,T,I) <=> N=8 | true.
(4.3l) ctr_comp_time(N,T1,T2,Rel,I) <=> empty(Rel) | false.
(4.3m) ctr_comp_time(N,T1,T1,Rel,I) <=> true.
(4.3n) ctr_comp_time(N,T1,T2,Rel,I) <=> N=5 | true.
% Intersection
%Intersections: The intersection is the same for all cases
%CASE A,B,C:
(4.4a) ctr_comp_top_time(N1,TB,TA,B,A,R1,T,I),
       ctr_comp_top_time(N2,TB,TA,B,A,R2,T,J) <=>
   intersection(R1,R2,R3), length(R3,N3), K is min(I,J)+1 |
   ctr_comp_top_time(N3,TB,TA,B,A,R3,T,K).
(4.4b) ctr_comp_top_time(N1,TB,TA,B,A,R1,T,I),
       ctr_comp_top_time(N2,TA,TB,A,B,R2,T,J) <=>
   conv_op(R2,R22), intersection(R1,R22,R3), length(R3,N3), K is
   min(I,J)+1 | ctr_comp_top(N3,TB,TA,B,A,R3,T,K).
(4.4c) ctr_comp_top_time(N1,TA,TB,A,B,R1,T,I),
       ctr_comp_top_time(N2,TB,TA,B,A,R2,T,J) <=>
   conv_op(R1,R11), intersection(R1,R11,R3), length(R3,N3), K is
   min(I,J)+1 | ctr_comp_top(N3,TB,TA,B,A,R3,T,K).
%Compositions
%CASE A:
(4.5a) ctr_comp_top_time(N1,TB,TA,B,A,R1,T,I),
   ctr_comp_top_time(N2,TC,TB,C,B,R2,T,J) ==>
   ((I=1,J<6);(J=1;I<6)), composition_op(TA,TB,TC,R1,R2,R3),
   length(R3,N3), K is I+J |
   ctr_comp_top_time(N3,TC,TA,C,A,R3,T,K).
(4.5b) ctr_comp_top_time(N1,TA,TB,A,B,R1,T,I),
   ctr_comp_top_time(N2,TC,TB,C,B,R2,T,J) ==>
   ((I=1,J<6);(J=1,I<6)), singleton(R1), conv_op(R1,R11),
   composition_op(TA,TB,TC,R11,R2,R3), length(R3,N3), K is I+J |
   ctr_comp_top_time(N3,TC,TA,C,A,R3,T,K).
```

***Algorithm 4.1 Part II.*** *Path consistency algorithm to propagate compositions of disjunctive motion (topology + time) relationships.*

```
(4.5c) ctr_comp_top_time(N1,TB,TA,B,A,R1,T,I),
   ctr_comp_top_time(N2,TB,TC,B,C,R2,T,J) ==>
   ((I=1,J<6);(J=1,I<6)), singleton(R2), conv_op(R2,R22),
   composition_op(TA,TB,TC,R1,R22,R3), length(R3,N3), K is I+J |
   ctr_comp_top_time(N3,TC,TA,C,A,R3,T,K).
(4.5d) ctr_comp_top_time(N1,TA,TB,A,B,R1,T,I),
   ctr_comp_top_time(N2,TB,TC,B,C,R2,T,J) ==>
   ((I=1,J<6);(J=1,I<6)), singleton(R1), singleton(R2),
   conv_op(R1,R11), conv_op(R2,R22),
   composition_op(TA,TB,TC,R11,R22,R3), length(R3,N3), K is I+J
   | ctr_comp_top_time(N3,TC,TA,C,A,R3,T,K).
%CASE B:
(4.5e) ctr_comp_top_time(N1,TB,TA,B,A,R1,T1,I),
   ctr_comp_time(N2,T1,T2,R2,J) ==> ((I=1,J<6);(J=1;I<6)),
   composition_time_op(TA,TB,T1,T2,R1,R2,R3), length(R3,N3), K
   is I+J | ctr_comp_top_time(N3,TB,TA,B,A,R3,T2,K).
(4.5f) ctr_comp_top_time(N1,TB,TA,B,A,R1,T1,I),
   ctr_comp_time(N2,T2,T1,R2,J) ==> ((I=1,J<6);(J=1,I<6)),
   singleton(R2), conv_time_op(R2,R22),
   composition_time_op(TA,TB,T1,T2,R1,R22,R3), length(R3,N3), K
   is I+J | ctr_comp_top_time(N3,TC,TA,C,A,R3,T2,K),
   ctr_comp_time(N2,T1,T2,R22,J).
%CASE C:It is implemented with the constraints in CASE A and B
%Auxiliary predicates
singleton([_]).
empty([]).
choose([R],[R|_).
/**PREDICATES TO DO COMPOSITION OF DISJUNCTIVE RELATIONS*******/
composition_time_op(TA,TB,T1,T2,R1,R2,R3):-
(    (singleton(R1),singleton(R2)) ->
     composition_time_simple(TA,TB,T1,T2,R1,R2,R3);
     composition1_time_op(TA,TB,T1,T2,R1,R2,[],R3) ).
composition_time_simple(TA,TB,T1,T2,[R1],[R2],R3):-
   composition_t(TA,TB,T1,T2,R1,R2,R3).
```

***Algorithm 4.1 Part III.*** *Path consistency algorithm to propagate compositions of disjunctive motion (topology + time) relationships.*

```
/***PREDICATES TO DO COMPOSITION OF DISJUNCTIVE RELATIONS.CONT.*/
composition_t(p,p,T1,T2,R1,R2,R3):-
   composition_table_PPt(R1,R2,R3).
composition_t(a,a,T1,T2,R1,R2,R3):-
   composition_table_AAt(R1,R2,R3).
composition_t(l,a,T1,T2,R1,R2,R3):-
   composition_table_LAt(R1,R2,R3).
composition_t(p,a,T1,T2,R1,R2,R3):-
   composition_table_PAt(R1,R2,R3).
composition_t(p,l,T1,T2,R1,R2,R3):-
   composition_table_PLt(R1,R2,R3).
composition_t(l,l,T1,T2,R1,R2,R3):-
   composition_table_LLt(R1,R2,R3).
composition_t(a,l,T1,T2,R1,R2,R3):-
   composition_table_ALt(R1,R2,R3).
composition_t(a,p,T1,T2,R1,R2,R3):-
   composition_table_APt(R1,R2,R3).
composition_t(l,p,T1,T2,R1,R2,R3):-
   composition_table_LPt(R1,R2,R3).

composition1_time_op(A,B,C,D,[],_,R3,R3).
composition1_time_op(A,B,C,D,[R1|R11],R2,Raux,Rdo):-
   composition1_t(A,B,C,D,R1,R2,R_parcial),
   union(Raux,R_parcial,R),
   composition1_time_op(A,B,C,D,R11,R2,R,Rdo).

composition1_t(A,B,C,D,R1,R2,R):-
   composition11_t(A,B,C,D,R1,R2,[],R).

composition11_t(A,B,C,D,R1,[],Rdo,Rdo).
composition11_t(A,B,C,D,R1,[R21|R22],Raux,Rdo):-
   composition_t(A,B,C,D,R1,R21,R3),
   union(Raux,R3,R),
   composition11_t(A,B,C,D,R1,R22,R,Rdo).
```

***Algorithm 4.1. Part IV.*** *Path consistency algorithm to propagate compositions of disjunctive motion (topology + time) relationships.*

```
/* The following files contain the converse table and
   composition tables of the topological information, the time
   information and the motion information*/
:-['inversas.pl'].
:-['composit.pl'].

:-['inversas_time.pl'].
:-['composit_time.pl'].
:-['composit_topo_time.pl'].
```

***Algorithm 4.1. Part V.*** *Path consistency algorithm to propagate compositions of disjunctive motion (topology + time) relationships.*

In algorithm 4.1 the predicate ctr*_comp_top_time/6* is the kind of constraint used to include to the topological information the time concept between two objects or regions. For example ctr_comp_top_time(A,A,Office$_1$,Office$_2$,[ci,tfi],T$_1$) means that between two regions named Office$_1$ and Office$_2$ which are of the type area holds the set of topological relations [ci, tfi] in the moment of time T$_1$, which means that at the moment of time T$_{1,}$ Oficce$_2$ is completely-inside or touching from inside Office$_1$. Whilst, the predicate *ctr_comp_time/3* is the kind of constraint used to represent time information between two points of time, for instance *ctr_comp_time(T$_1$,T$_2$,[prev])* means that between the two points of time T$_1$ and T$_2$ holds the time relation *prev*, which means that T$_1$ is previous to T$_2$.

The constraints ctr_comp_top_time/6 and ctr_comp_time/3 are the ones introduced to the system and both types are translated into ctr_comp_top_time/8 and ctr_comp_time/5 respectively by CHRs (4.2a) and (4.2b) where length (N) of the relationship is added as well as the length (I, which initially is 1) of the shortest path form which the constraint is derived in order to increase the efficiency of the algorithm as in the case of the algorithm 3.1 of section 3.3. Argument N avoids compositions between constraints which do not improve the motion information and argument I restricts one of the two constraints involved in the propagation to being a direct constraint, which means that it should not be derived from another propagation rule.

Constraints will be treated by the CLP clauses (4.1c) and (4.1f) if the motion relation (Rel in 4.1c) and the time relation (Rel in 4.1f) are a disjunction of primitive motion relations (4.1b) and a disjunction of primitive time relations (4.1e) respectively. The backtrack search part of the algorithm is implemented by the predicates (4.1c) and (4.1f) where *member(R,Rel)* chooses non-deterministically one primitive motion symbol (R in 4.1c) and time symbol (R in 4.1f) from the disjunctive motion relation (Rel in 4.1c) and the disjunctive time relation (Rel in 4.1f) respectively.

Special cases are simplification CHRs. (4.3a) and (4.3l) detect inconsistent constraints. Inconsistent constraints in this case provoke the failure of the algorithm. Another behaviour can be obtained be changing these rules. (4.3b) and (4.3m) delete constraints which contain equality. The CHR (4.3n) deletes constraints when the length of the time relations equals the length of the structure relations, which is equivalent to knowing nothing about the time information. The rest of special CHRs delete constraints when the length of each type of motion relations equals the length of its respective structure relations, as (4.3n) does for time relations.

Simplification CHRs (4.4a), (4.4b) and (4.4c) implement the intersection part of the three formulas (7), (8), and (9). (4.4a) is the original intersection CHR and CHRs (4.4b) and (4.4c) defines intersection applying the converse operation to the second and first constraint respectively.

Propagation CHRs perform composition of formulas (7), (8), and (9). (4.5a) implements composition as it is initially defined by the formula (7) for the case A for motion information. CHRs from (4.5b) to (4.5d) apply the converse operation to the first, the second or both constraints respectively for the formula (7) (case A). (4.5e) implements composition as it is initially defined by the formula (8) for the case B for motion information and (4.5f) applies the converse operation to the second constraint for the same formula (case B). Finally, no more propagation CHRs are implemented for the formula (9) (case C) because the propagation CHRs implemented also implement this case as it has been explained previously in this section.

Therefore, the algorithm defines a total of 19 simplification rules, and 6 propagation rules.

# CHAPTER 5

## QUALITATIVE THEORY FOR SHAPE REPRESENTATION AND MATCHING

This section describes the qualitative theory for shape representation and matching that we have developed. Following the classification given in chapter 2.3.2 the theory proposed in this section can be classified as a reference points based representation due to the fact that the theory uses the vertices (reference points) of the objects to give its description, and it does not segment complex shapes in primitive shapes, on the contrary it gives a unique and complete description of each shape. This theory considers qualitatively the angles, relative side length, concavities and convexities, types of curvatures of the boundary of the objects. And as we will see later, it also considers qualitatively, the colour of the objects for some applications. These aspects have not been considered in other approaches. The shapes recognised are regular and non-regular closed polygons that can have curve segments and curvilinear shapes. Moreover the shapes can contain holes. To describe shapes with holes, topological and qualitative spatial orientation aspects are considered in order to relate the hole with its container. Each object is described by a string containing its qualitative distinguishing features (symbolic representation), which is used to match the object against others.

Shape description using reference-points information will have to make use of some landmarks. As reference points (landmarks) we understand these points which completely specify the boundary. For polygonal boundaries we have chosen the vertices as reference points. For circular shapes and curvilinear segments in a shape we have chosen three points: the starting and the end point of each curve and its point of maximum curvature.

The qualitative description of a reference point, named j, is determined using the previous reference point, named i, and the following reference point, named k. The order of the reference points is given by the natural cyclic order of the vertices of closed objects. We only have to determine the sense in which we visit or describe each reference point, which should be the same for the description of all objects. We have chosen to visit the vertices in a counter-clockwise sense.

## 5.1. THE REPRESENTATION OF SHAPES OF IRREGULAR POLYGONAL OBJECTS.

The central idea of the qualitative shape representation consists of giving three reference points *i, j, k,* which are consecutive, the qualitative description of the reference point *j* is determined by positioning an oriented line from the point *i* to the point *k* as figure 5.1 shows. In figure 5.1 *i* is the vertex 1, *j* is the vertex 2 and *k* is the vertex 3 and the oriented line is placed from 1 to 3.

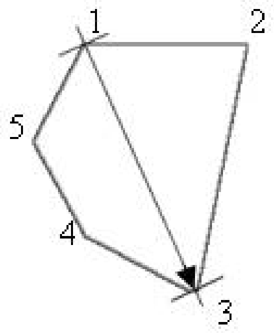


***Figure 5.1.*** *Example of a shape figure in which we are determining the qualitative description of vertex 2 using vertex 1 and 3 by placing and oriented line between them.*

In the case of a polygonal non-regular shape all its segments are straight segments. Therefore, the description of each reference point is given by a set of three elements (triple) which is <Aj,Cj,Lj>, where Aj means the angle for the reference point j, Cj means the type of convexity of point j and Lj means the relative length of the edges associated to reference point j (edge formed by vertices i and j versus edge formed by vertices j and k), where:

Aj ∈ {right-angled, acute, obtuse};

Cj ∈ {convex, concave} and

Lj belongs to LRS, where LRS = {smaller, equal, bigger}.

### 5.1.1 Determining the Convexity

The convexity of the point j is determined as follows: if the reference point *j* remains on the left of the oriented line from *i* to *k* then the point *j* is a convex vertex. Otherwise, if the point *j* remains on the right of the oriented line from *i* to *k* then the point *j* is concave. As we find a vertex when the orientation of the edge changes then it is not possible that the reference point *j* remains exactly above the oriented line from *i* to *k*. Formally, if *Vj* means vertex *j*, and *Vj wrt ViVk* means the relation of the vertex *j* with respect to the oriented line from vertex *i* to vertex *k*, we can formulate:

*If Vj wrt ViVk Î left then Vj is convex.*

*If Vj wrt ViVk Î right then Vj in concave.*

### 5.1.2 Determining the Angle

The qualitative description of an angle is determined using a new concept and some topological concepts as boundary, interior and exterior of an entity, defined in chapter 2. The new concept consists of giving the two reference points joined by the oriented line, i and k, we place a circle of diameter ik between these two reference points.

Therefore, the angle is determined as follows; if the reference point *j* remains exactly in the boundary of the circle of diameter *ik,* then the vertex *j* is right-angled. If j remains in the exterior of the circle then *j* is acute. And if *j* remains in the interior of the circle then the vertex *j* is obtuse. Formally, if the circle with a diameter of *ViVk* is denoted as *Cik*, then the angle of the Vertex *j* (*Vj*) is calculated using the following algorithm:

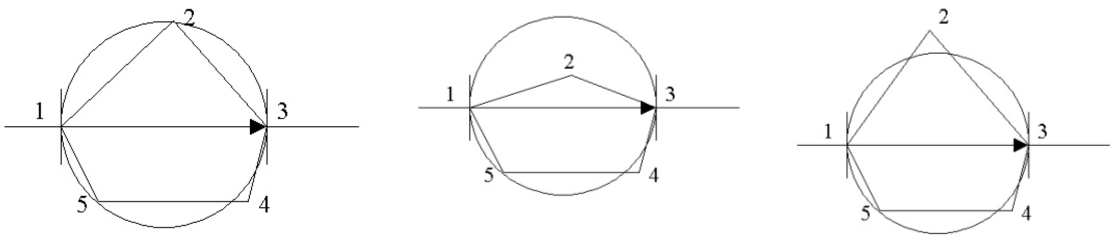
*If Vj Ç dCik ¹ Æ then Vj is right- angled,*

*Else if Vj Ç Cikº ¹ Æ then Vj is obtuse*

*Otherwise Vj is acute.*

The part of the “otherwise” of the above algorithm occurs when Vj ∩ Cik- ≠ ∅.

The following figure shows an example in the form of a graph for each of these cases (figure 5.2).



a) Right-angled angle b) Obtuse Angle c) Acute Angle

***Figure 5.2.*** *Examples of determination of the angle of vertex 2, using an oriented line from vertices 1 and 3and a circle of diameter vertex1vertex3; a) for a right – angled angle; b) for an obtuse angle and c) for an acute angle.*

### 5.1.3 Determining the Length

To determine the relative length of each edge of a rectilinear segment between three contiguous vertices (relative length of edges between the edge from vertex *i* to vertex *j* and the edge from vertex *j* to vertex *k*) a new length model has been developed which has been inspired in the model by [Hernández, Clementini, and Di Felice 95] and [Escrig and Toledo 98].

The length model developed compares lengths of two consecutive edges of the object. As we compare lengths, at least two lengths have to be available, and as a result we find that one length is bigger, smaller than or equal to the other.

Therefore the reference system named Length Reference System (LRS) is defined by a set of qualitative length labels:

LRS={smaller, equal, bigger}.

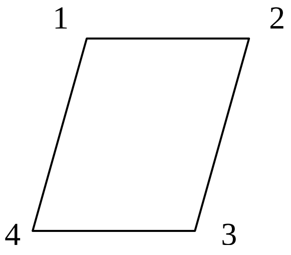


***Figure 5.3.*** *Example of the lengths to compare. If we want to calculate the value for the compared length in vertex one, then we compare the length from vertex 4 to 1 with the length from vertex 1 to 2, and it is determined as bigger.*

The length calculated in the reference point *j* is the length of the edge from the point *i* to the point *j* compared with the length of the edge from the point *j* to *k* (see figure 5.3 for an example) , using this LRS. Therefore it is inferred that:

- First the length of each edge is calculated by using the Euclidean distance $D(V_i, V_j)$ between two points:

  $D(V_i, V_j)=((X_{vj}-X_{vi})^2+(Y_{vj}-Y_{vi})^2)^{1/2}$, where $V_i = (X_{vi}, Y_{vi})$ and $Vj=(X_{vj}, Y_{vj})$.

- Then, both lengths are compared and the corresponding label of the LRS is assigned as the value of the relative length to the vertex *j*.

## 5.2. THE REPRESENTATION OF SHAPES OF OBJECTS WITH CURVES.

In the case of a shapes with curvilinear segments, the description of each reference point is given by a set of three elements (triple) which is:

- <Aj,Cj,Lj> for the case of its straight segments (described in section 5.1), or
- <Curve,Cj,TCj>, where the symbol Curve means that the node in the description string is describing a curve , Cj means the type of convexity of point j and TCj means the type of curvature of the curve associated to the point j, where:

  Cj $\in$ {convex, concave} and

  TCj $\in$ {plane, semicircle, acute}

If the shape contains only curvilinear segments all its triples for describing it will be of the form <Curve,Cj,TCj>.

Therefore, for describing an object with curves we follow the following steps:

1. First of all the symbol curve is fixed to indicate that the next node in the qualitative description of the object corresponds to the description of a curve.
2. To describe qualitatively a curve, 3 points are used: the initial and final points of the curve and the point of maximum curvature of the curve (as depicted in figure 5.4a), which are obviously consecutive points. The description, however, is associated only to the node of maximum curvature.

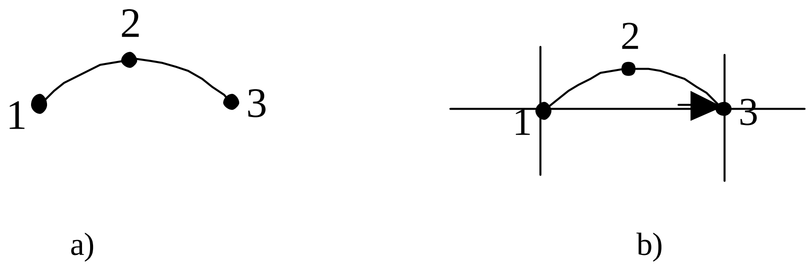


***Figure 5.4.*** *a) The 3 points considered for the description of a curve and b) the placement of the oriented line between them.*

The central idea of the qualitative shape representation for curves consists of, given the three reference points, *i, j, k*, of the curve which are consecutive, the qualitative description of the reference point *j* (the one of maximum curvature) is determined by positioning an oriented line from the points *i* (the previous point) to the point *k* (the following point) as figure 5.4b) shows. In figure 5.4b) the point 1 is point *i*, the point number 2 is the point *j* and the point number 3 is the point *k*.

### 5.2.1 Determining the Convexity

The convexity (Cj) of the point *j* is determined by the oriented line from *i* to *k* as follows: if the reference point *j* remains on the left of the oriented line from *i* to *k* then the point *j* is a convex vertex. Otherwise if the point *j* remains on the right of the oriented line from *i* to *k* then the point *j* is concave. As *j* is the point of maximum curvature in a curve segment from *i* to *k*, then it is not possible that the reference point *j* remains exactly over the oriented line from *i* to *k*. Formally, if Vj means vertex *j* (reference point which belongs to the one of the maximum curvature), and *wrt* means the relation of the vertex *j* with respect to the oriented line from vertex *i* to vertex *k*, we can formulate:

*If Vj wrt ViVk Î left then Vj is convex.*

*If Vj wrt ViVk Î right then Vj is concave*

### 5.2.2 Determining the Type of Curvature

The type of curvature (TCj) of the point *j* is determined by calculating two distances and comparing them (figure 5.5). For calculating both distances the centre point of the line between *i* and *k* is calculated, named point *ik* (Pik). The first distance (da) calculated is the distance between *i* and the new point, Pik, and the second distance (db) considered is the one between the point *j* and the new point (Pik). Then comparing both distances TCj is determined as follows:

If da<db → TCj = acute

If da=db → TCj = semicircle

If da>db → TCj = plane

Figure 5.6 shows examples of the 3 possible cases.

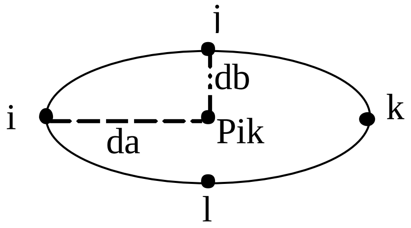


***Figure 5.5.*** *Distances calculated for determining TC$_2$.*

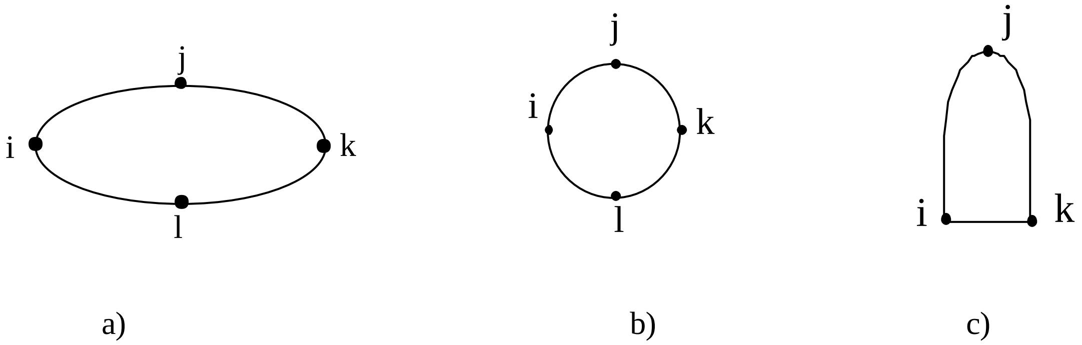


***Figure 5.6.*** *a) Point j has a TCj = plane, b) TCj = semicircle, and c) TCj = acute.*

## 5.3. THE REPRESENTATION OF SHAPES OF OBJECTS WITH HOLES.

To describe objects with holes we use the topological concept of Completely Inside Inverse (CIi) defined in section 3, due to the fact that the hole is always Completely Inside (CI) the boundary of the closed objects. Moreover, the cardinal reference system by Frank (Frank, 1991) is used also in order to relate the position of the hole inside the object.

Therefore, for describing an object with holes we follow the following steps:

1. The qualitative shape description of the exterior boundary of the object (container) is constructed following the steps described in previous sections.
2. Then the qualitative shape description of the boundary of each hole is constructed.
3. Each hole and the container are related by adding two types of information:
    a. The topological relation between the container and each hole is fixed. The holes in the case of closed objects are always Completely Inside Inverse (CIi defined in section 3) of the container.
    b. The orientation of each hole inside the container is determined (this is necessary because we can have objects with a hole in which the boundaries of containers are equal and boundaries of the holes too, but the hole is placed in another position of the container so that they are not the same object). The orientation is fixed using Frank's Cardinal Reference System (CRF) [Frank 92], which divides space into eight or more cones (which allows working with different levels of granularity) as figure 5.7 shows. The CRF is defined by placing its origin into the centroid calculated with the definition of the centroid of a close non regular polygon given in [Steger 96]. In the case of curvilinear shapes or shapes which contain curve segments, these shapes are approximated to polygonal shapes in which the vertices are the reference points considered for the qualitative description of the shape and these vertices are joined by rectilinear segments. [Steger 96] calculates the centroid ($\alpha_{1,0}$ is the x coordinate and $\alpha_{0,1}$ is the y coordinate) in basis of the area ($\alpha$) as:

$$a = \frac{1}{2}\sum_{i=1}^{n} x_{i-1}y_i - x_i y_{i-1}$$

$$a_{1,0} = \frac{1}{6a}\sum_{i=1}^{n}(x_{i-1}y_i - x_i y_{i-1})(x_{i-1} + x_j)$$

$$a_{0,1} = \frac{1}{6a}\sum_{i=1}^{n}(x_{i-1}y_i - x_i y_{i-1})(y_{i-1} + y_j)$$

We call centre (C ) to the orientation that occurs when the hole is placed around the centroid, and all orientations hold.

When several orientations hold for a given hole, then the orientation is fixed to a set of all the orientations (figure 5.7).

Then once the CRF is placed in the object, the orientation of the hole with respect to the object is calculated. For instance, figure 5.6 calculates the orientation of the hole with respect to the container, obtaining that the hole is [NE,E,SE] oriented inside the container.

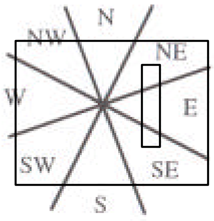


***Figure 5.7.*** *Example of the Orientation Calculation of a hole with respect to its container.*

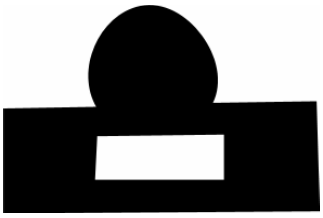
## 5.4. THE COMPLETE DESCRIPTION OF A SHAPE

As the colour is a relevant characteristic in the case of some of the applications that we have developed, the colour of the shape is stored as RGB colour and then in the matching process the colour can be considered qualitatively using the Delta E distance between colours.

Given a shape, its complete description is defined with the following tuple: [holes_type, curves_type, [Colour, [A1,C1,L1 | Curve, C1, TC1]…[An,Cn,Ln | Curve, Cn, TCn]],(CIi,Orientation,[curves_type,[AH1,CH1,LH1|Curve,CH1,TCH1]…[AHj,CHj,L Hj |Curve,CHj,TCHj]])$^{m}$], where *n* is the number of vertices (reference points) of the container and *j* is the number of vertices of the holes (reference points). The holes_type belongs to the set [without-holes, with-holes], the curves_type belongs to the set [without-curves, with-curves, only-curves], both symbols are introduced to speed up the matching process. Colour is the RGB colour of the piece described by a triple as the set [R,G,B] for the Red, Green and Blue coordinates. Each set , [A1,C1,L1 | Curve, C1, TC1] represents a node of the qualitative description which can be the description of a vertex of a rectilinear segment, and then A1,…,An, C1,…,Cn and L1,…,Ln are the qualitative angle, convexity type, and relative length of the vertices of rectilinear segments of the container respectively, or ( represented by the symbol | ) it can represent a node of the qualitative description of a curvilinear segment and then it is formed by the symbol Curve to indicate that it is a curvilinear segment. C1, …, Cn, and TC1, …, TCn are the qualitative description of the convexity type and curvature type respectively. The same happens with the container of the hole: AH1,..AHj, CH1,…,CHj and LH1,…LHj, are respectively the qualitative angle, convexity type, and relative length of the vertices of rectilinear segments of the hole and CH1,…, CHj, and TCH1,…, TCHj are respectively the convexity type and curvature type of the curvilinear segments of the hole. The string CIi, Orientation, [[AH1,CH1,LH1]…[AHj,CHj,LHj]] is repeated for each hole inside the container. CIi is the topological relation relating the hole with its container. Finally, Orientation is the set of orientation relations given by the Frank's Cardinal Reference System (CRF) in order to give the orientation of each hole in the container.

Therefore, in order to describe completely a shape, first we have to repeat the process of giving the qualitative description of each vertex to describe the boundary of the container and the boundary of each hole (if they appear). Colour is stored as RGB coordinates. The orientation relations between the container and each hole are calculated using CRF. And the final set (string) with the characteristics of the shape is constructed.

Figure 5.8 shows an example of a shape with a hole, rectilinear and curvilinear segments and its qualitative shape description, formally named QualShape(S), being S the reference to the object described.



QualShape(S)=[with-holes, with-curves, [[0,0,0], [right-angle,convex,bigger], [curve, convex, acute], [right-angle, convex,bigger], [right-angle, convex, smaller], [right-angle, convex,bigger]],CIi,C,[[right-angle, convex, smaller], [right-angle, convex, bigger],[right-angle, convex, smaller], [right-angle, convex, bigger]]].

***Figure 5.8.*** *Example of a black (RGB = 0R,0G,0B) shape with a hole, curvilinear and rectilinear segments.*

## 5.5. THE MATCHING PROCESS

The matching process determines, in the case that there are two objects, if both have the same shape and colour. To do that, one of the objects is taken as reference and the other is considered the shape to be compared and matched.

The matching process is made as follows, first the qualitative description of the object taken as reference is constructed, as defined in previous sections. Then the qualitative description of the other object is constructed but only up to the description of the container, it means that the holes are not yet described. With these two strings we compare if both strings are of the same type (with or without holes), same colour (in a qualitative way) and the containers are equal. For comparing the colour qualitatively, we considered that colours will be always solid colours, the Delta E distance between colours is used. The Delta E distance using RGB colour systems is calculated as:

Given two colours in RGB, named C1 determined by (R1,G1,B1) and C2 determined by (R2,G2,B2), then the Delta E distance between colours is calculated as the Euclidian distance between the RGB coordinates of each colour as:

$$\text{Delta\_E}(C1,C2)=((R1-R2)^2+(G1-G2)^2+(B1-B2)^2)^{1/2}$$

If the Delta_E is less than 0,2 it is because an experienced human eye in the recognising of colour fields cannot differentiate between the two colours.

Due to implementation reasons, the vertices of containers and holes of each shape are numbered in a counter clockwise way, being the first vertex (number 1) the uppermost (left) vertex of the shape.

To compare the containers, the algorithm ComparingVertices (algorithm 5.1) is applied, which is a cyclical ordering matching algorithm which given two set of vertices, returns if both strings are equal cyclically and the vertex in the second object which corresponds to the vertex number 1 in the first one. If both sets are not equal the vertex in the second set is not found, therefore a –1 value is assigned.

```
Algorithm    ComparingVertices    (INPUTS:    SetVertices1,
SetVertices2, OUTPUTS: vertex02, equal)
{
  N=Calculus size SetVertices1
  M=Calculus size SetVertices2
  If N == M then
{
   //Both sets have the same number of vertices
   For (I=0;I<N-1;I++)
         {
       For (J=0;J<N-1;J++)
             { //cyclic comparison
//Compare  Vertex1(0)  of  SetVertices1  with  Vertex2(j)  of
//setVertices2
           If Vertex1(0) == Vertex2(j)
               {
        Num=0 //Init a counter
        For (K=1;K<N-1;K++)
                {
                    If (Vertex1(K)==Vertex2(J+1%N)) then
                    {
                    NUM++;
                    }
                    If (NUM==N)
                   {
                     Return equal=true;
                              Return vertex02= j;
                     Break
                    }
            } //For K
         } // If Vertex1(0) ==Vertex2(j)
      } //For J.
   } //For I
  If (NUM<>N)
  {
      Return equal=false;
      Return vertex02= -1;}
  } //If N==M
 else
 {
      Return false;
  }
} //End Algorithm
```

***Algorithm 5.1.*** *Cyclical ordering matching algorithm.*

If the objects have no holes the process finishes here.

We decided to start the matching process in this way because the objects with holes that are found rotated with respect to the reference object will describe the holes in an orientation different from the one given by the reference object both being the same object. This occurs if we consider an absolute and fixed orientation for the Frank's Cardinal Reference System (CRF) as figure 5.9 shows. This happens because we do not know yet how we have to orientate the CRF with respect to the container. So, first of all we have to know how to orientate the CRF with respect to the container.

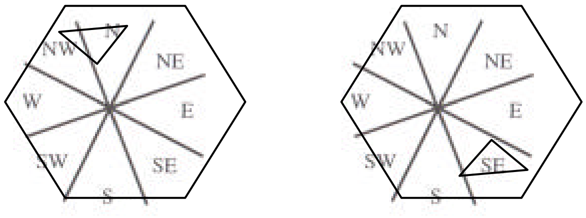


***Figure 5.9.*** *Two qualitative equal shapes, but with different orientations. Using an absolute and fixed CRF the holes will have different orientations and therefore both qualitative descriptions will be not equal.*

So once it can be demostrated that both objects are equal up to the container, and both contains holes, the string describing the holes of the second object (not the reference object already completely described) is constructed by using the following steps:

1. Each hole in the object is numbered as being the vertex number 1 the vertex closest to the vertex which corresponds to the vertex 1 in the reference object, calculated when the cyclic comparison has been made.

2. Include the string CIi in the qualitative description of the object for the first hole and calculate the orientation of the first hole with respect to the container placing the NW of the CRF oriented to the vertex which corresponds to the vertex 1 in the reference object, and include it in the qualitative description of the object.

3. Calculate the qualitative description of the boundary of the first hole ([[AH1,CH1,LH1]…[AHj,CHj,LHj]]) and include this description in the qualitative description of the object.

4. Repeat steps 2,3, and 4 for each hole inside the object.

Once the qualitative description of the second object is completed, first we compare the number of holes, if both objects have the same number of holes we continue comparing using the following steps:

1. Choose one hole of the figure to compare.

2. Compare the orientation of this hole with the orientation of each hole of the reference figure. If there is not a hole with the same orientation, the matching process finishes answering that the figures are not equal, otherwise the next step is performed.
3. Compare the qualitative description of both holes by doing a non cyclic comparison. A cyclic comparison for figures with holes like the ones in figure 5.10 would return that they are equal.
4. Repeat steps 1,2 and 3 for the rest of holes of the figure to compare.

If all the holes in the figure to be compared have a matching hole in the reference object both objects are equal.

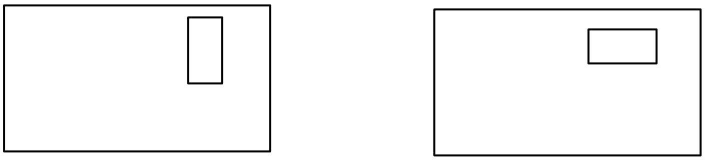


***Figure 5.10.*** *Two different objects with identical holes in different positions.*

Moreover, if in an application the size of the objects is an important feature (for instance two squares of very different size are not the same piece, and in a specific application this is an important consideration), then the area of the shapes is considered. The area is also needed for the calculation of the centroid of shapes, therefore we do not add more computational cost. The area once more is compared in a qualitative way. The limit to determine two shapes as the same is given by the application itself. For instance in chapter 6 we will see how this theory has been applied to the recognition of tiles which form a mosaic design, then in this application, size is important and the limit is given by the joint between tiles (space left between two tiles when they are assembled). As the joint differs from one type of design to another it is given by the user of the application. Then if the difference between the areas of the tiles is less than the joint size, the shapes represent the same object, otherwise they do not represent the same object.

### 5.5.1 Computational Cost of the Qualitative Theory of Shape Description and Matching.

To conclude this chapter we can say that we have defined a straightforward qualitative theory of shape description. It will allow us to reason about shape in a qualitative way as human beings do. The theory proposed here provides a simple example for representing shapes without and with holes and without and with curves. The interest of this qualitative shape description relies in the fact that it is less constrained than metrical information but more constrained than topological information, since our qualitative theory of shape description will allow us to determine the convexity or concavity of the shape, the length of edges, and the angle types. Moreover, another advantage of our theory is that it has a lineal computational cost:

- The temporal computational cost for the worst case need for the construction of a qualitative description of the figures is of the order O(N+M*K), where N is the number of vertices of the container, M is the number of vertices of the holes and K is the number of holes.
- The temporal computational cost for the worst case need for the matching algorithm between 2 shapes is of the order $O(n^2+k^2*(N-1)+2*M)$, being N the number of vertices of the containers, K the number of vertices of the holes and K the number of holes.

# CHAPTER 6

## APPLICATIONS

In this chapter we will show two different applications developed to implement the theories presented in previous chapters.

Although the topological calculus and motion theory have been implemented using PROLOG as a CLP problem, the algorithms developed have been used to test the correctness of the calculus with several examples. But the motion model presented in chapter 4 has been implemented too in an application for the navigation of a real Khepera[2] robot as we will show in section 6.1. The robot has the goal of navigating in its environment and describing two objects that it will find during its navigation using the qualitative shape description theory presented in section 5 and answering if both objects are of the same shape or not.

Section 6.2 explains another application (application number 2) of the qualitative shape representation and matching theory presented in chapter 5. It is an industrial application whose main objective is the automatic and intelligent recognition of mosaic tiles to be matched against a mosaic design, in order to be able to assembly them automatically for creating mosaics of different designs.

### 6.1. APPLICATION NUMBER 1: ROBOTICS APPLICATION

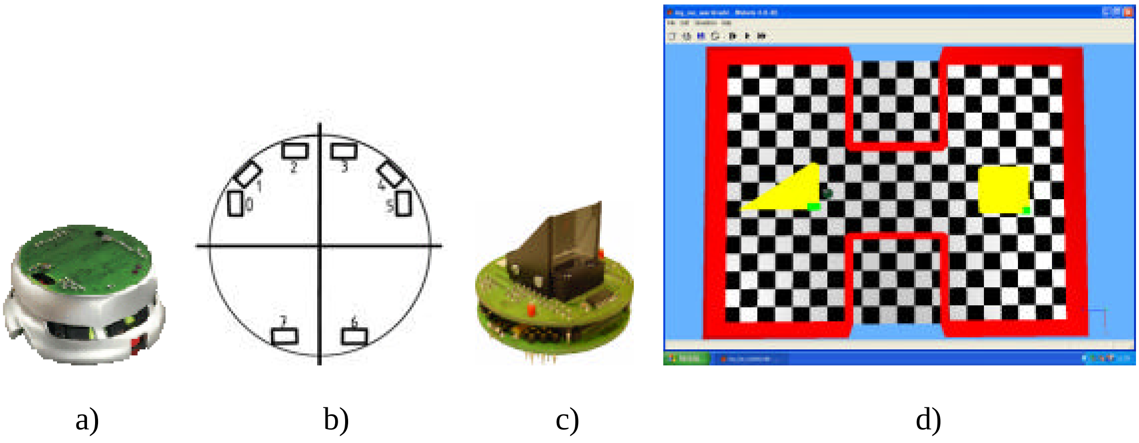
The application developed consists of the robot navigation of a Khepera[2] robot from K-Team (figure 6.1a) which:

1. First it constructs the topological map of its environment.
2. Then using the movement theory presented in chapter 4 it plans its path from the origin to the goal region in function of the different topological situations that the robot as a region has to hold with the other regions of its environment during its movement. After the robot uses the plan in order to avoid losing the correct

path. The plan will be based in the position of two objects. The two objects will be in two separated rooms (figure 6.1d), one of them will be the initial region and the other will be the goal region.

3. When the robot is in the initial region, it looks for the object in the room, it approaches the object and following the boundary of the object (using its infrared sensors) it constructs a qualitative description of it.
4. Next, it goes to the goal region comparing the plan defined in point 2 with each topological region holding between the robot and the region of the environment in which the robot is. If the result of the comparison is that both situations are equal the robot continues its navigation. Finally, when it is in the goal region, it looks for the second object, and approaches it. Then, as before, it follows the boundary of the object and it constructs its qualitative description.
5. Finally, it returns whether the objects are equal or not. If the objects are qualitatively equal the robot turns itself around, otherwise it stops.

The Khepera$^2$ robot (figure 6.1a) is a miniature desktop robot with eight infra-red proximity sensors which are placed around the robot and are positioned and numbered as shown in figure 6.1b). For the application the robot is provided with a K6300 Vision Turret, which is a colour matrix vision sensor with an image resolution of 160x120 pixels RG/GB 8-bit pixels, designed to provide the Khepera$^2$ robot autonomous embedded image processing (figure 6.1c). It is necessary to use this turret because the infra-red proximity sensors of the robot have a very short reach and using the turret the task of moving the robot close to the object is easier and faster. Moreover, in order to make the creation of the topological map easier, colours are used to distinguish different regions and the camera is used to recognise the colours. When the robot is close of the object, the infra-red sensors are used to follow its boundary.



a) b) c) d)

***Figure 6.1.*** *a) Snapshot of the Khepera2 robot; b) position of the infra-red sensors; c) K6300 Vision Turret and d) snapshot of the world of the robot in the simulated environment.*

All the concepts introduced here are going to be presented in more details in the following sections.

## 6.1.1 Learning the topological Map of its Environment

The robot constructs the topological map of its environment while it is moving around this environment. The topological map is made of several regions, each region contains:

- Its name which is a numerical identifier.
- Its colour.
- The approximate coordinates of the corners that compose the region.

The topological map is made up using a map of colours and a map of points. The following sections describe the process by which these maps are created.

### 6.1.1.1 Creation of the Map of Colours

Khepera$^2$ can distinguish all the colours of its environment using the k6300 vision turret. For the Map of Colours useful for the construction of the topological map, the relevant colours are: red and blue (because room walls of its environment are depicted in these colours) and black because it is the colour that will indicate the start and end point of the robot travel across the environment.

The camera takes pictures of its environment and then:

1. The robot is in a red room when the four pixels of the four corners of the picture are red. This situation occurs when the robot is in one of the four corners in one of the rooms (figure 6.2).

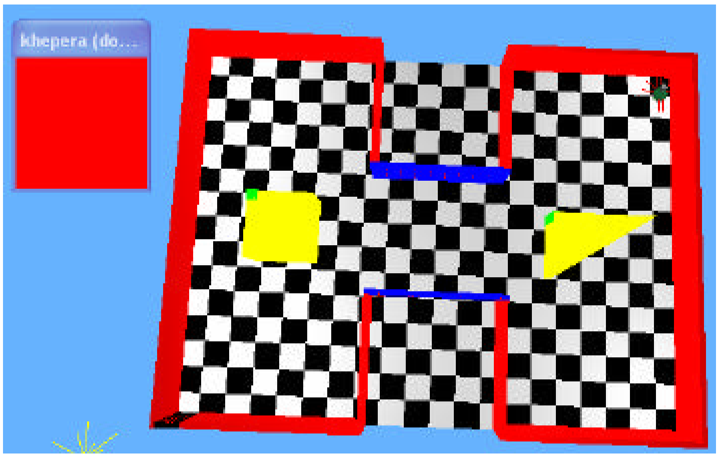


***Figure 6.2.*** *At the left there is the picture taken by the camera when the robot is in the situation of the right.*

2. The robot is in a corridor (a blue region) when the central pixel and the two pixels in the upper-right and upper-left corners are blue. This situation happens when the robot turns a corner of the corridor (figure 6.3).

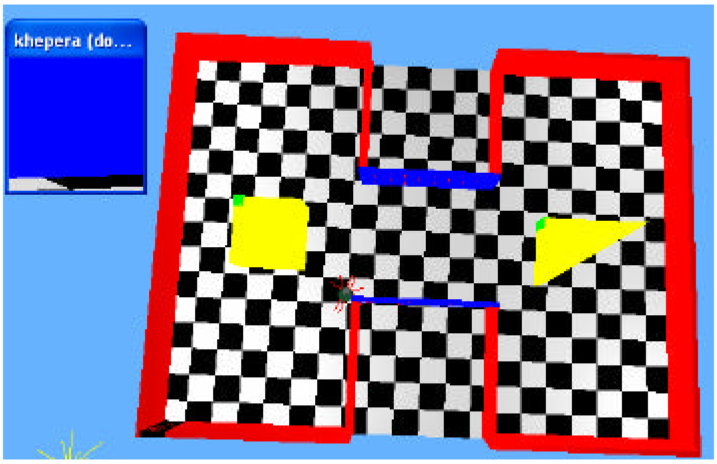


***Figure 6.3.*** *At the left there is the picture taken by the camera when the robot is in the situation of the right.*

When the robot detects the change from one colour to another (it goes from a room to the corridor or vice versa), the first colour is stored in the map of colours. The robot detects the change of colour because the robot remembers the last colour detected, then it will avoid detecting several times the same room or corridor, which would cause the robot to obtain an erroneous map of colours.

Finally, the robot detects the starting point of the environment (in order to start the construction of it) and later the ending point of the environment (then the map is finished) when it detects the black colour. This colour is determined if the colour of the pixel in the centre of the image is black (figure 6.4).

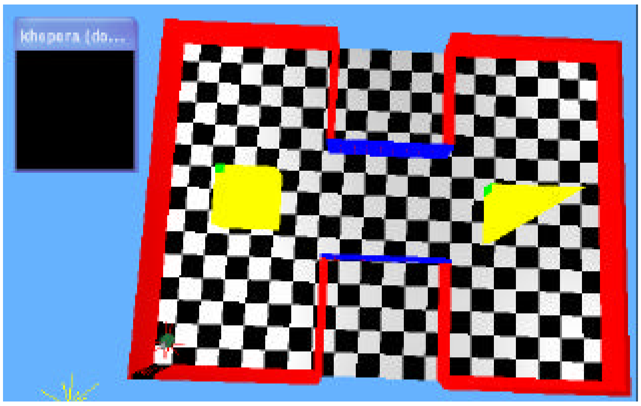


***Figure 6.4.*** *At the left there is the picture taken by the camera when the robot is in the situation of the right.*

Using the colours detected and exploiting the characteristics of semi-structured, close and symmetric respect the X axis environments (in this case made of rooms connected by corridors, figure 6.5) the process followed to obtain the final map is as follows:

- We store the room colour that the robot visits during its movement following the wall of the environment.
- Then, at the end of the movement we have a string of reversible number (due to the symmetry of the world) and the final map of colours will be just the first half of this string plus the central colour of it.

Figure 6.5 shows a virtual world made of four rooms connected by three corridors. After the movement of the robot through all the environment the string would be RBRBRBRBRBRBR (where R = red and B=blue), and the final map of colours would be the first half of this string plus the central element, that is RBRBRBR. This map of colours indicates to the robot the structure of the world, meaning that the world has four red rooms and three blue corridors.

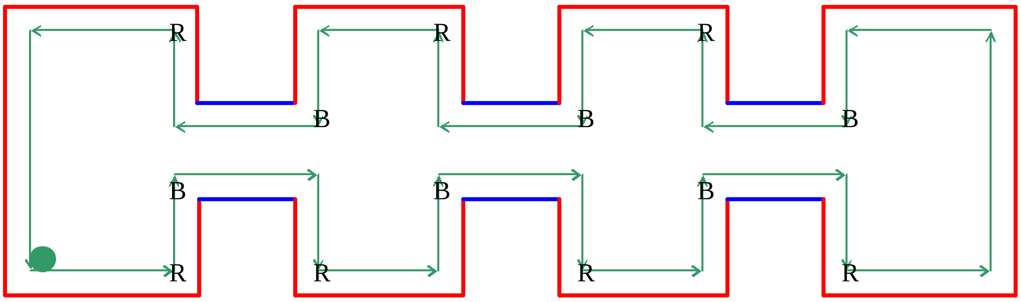


***Figure 6.5.*** *Virtual World and colours detected by the robot during its movement following the wall of the environment.*

Finally, applying this process to the simulated world used in Webots for the Khepera$^2$ (figure 6.3) the resulting string is RBRBR, and the final map of colours is RBR, which means two red rooms connected by a blue corridor.

#### 6.1.1.2 Creation of the Map of Points.

At the same time that the robot follows the wall and constructs the map of colours it also constructs the map of points. The robot determines if its movement is in a positive or negative direction respect to the X axis. To determine this direction the robot sees which sensors are close of the wall, if they are the right sensors then the robot follows a positive direction, otherwise (left sensors are the ones close to the wall) it follows a negative direction. Figure 6.6 shows the sketch of turns that the robot makes when it starts the process of following the wall in a positive direction. The direction of the turns will be the opposite one of the figure 6.6 if the robot follows the wall in a negative direction. From this sketch, we observe that each time that the robot is in a corner of a room its frontal sensors are close to a wall too and the robot must turn 90º in a positive direction. But when the robot goes following the wall from a room to a corridor, or vice versa, the robot turns 90º in a negative direction.

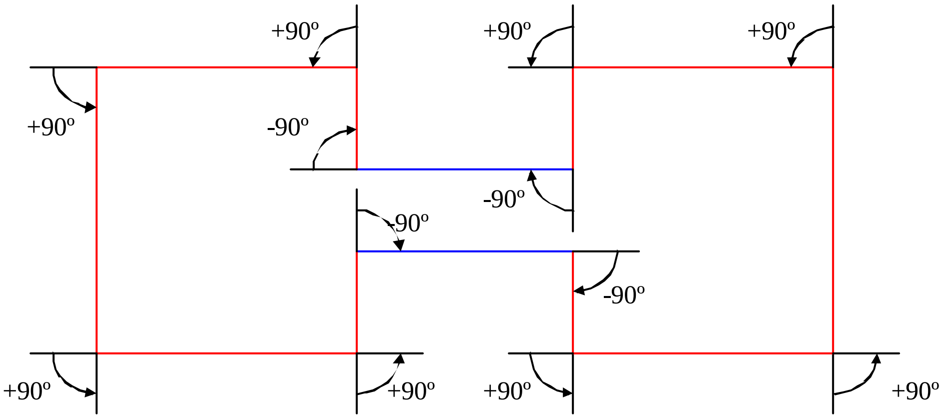
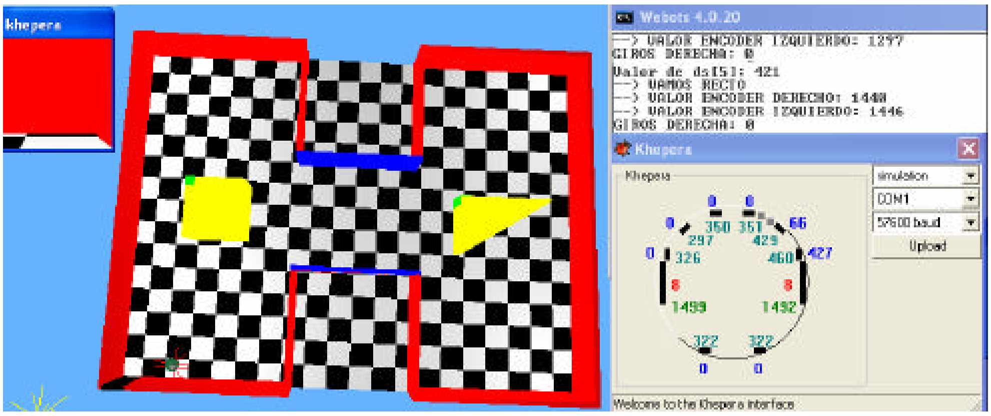


***Figure 6.6.*** *Sketch of robot turns through the environment when following the wall in a positive direction (right sensors close of the wall).*

On the other hand, robot encoders measures the movement of the robot during its route, then if the encoder value is reset after each turn made we obtain the distance, in the encoder unit, between one corner of the world and the next one. Figure 6.7 shows the values of the right encoders and left encoders during an execution.



```
--> VALOR ENCODER IZQUIERDO: 1297
GIROS DERECHA: 0
Valor de ds[5]: 421
--> VAMOS RECTO
--> VALOR ENCODER DERECHO: 1440
--> VALOR ENCODER IZQUIERDO: 1446
GIROS DERECHA: 0
```

Welcome to the Khepera interface

***Figure 6.7.*** *Right and Left Encoders Values. In the khepera sensors window this values are the one in the darker green.*

Finally, considering the angles of turning, the direction of turning, and the distance calculated by the encoder between two rooms the coordinates of corners of each region are calculated as follows:

$$(x + \text{incremento\_x},\ \ y + \text{incremento\_y})$$

where:

$$\text{incremento\_x} = \text{cos}(a) \cdot \text{desplazamiento\_encoders}$$

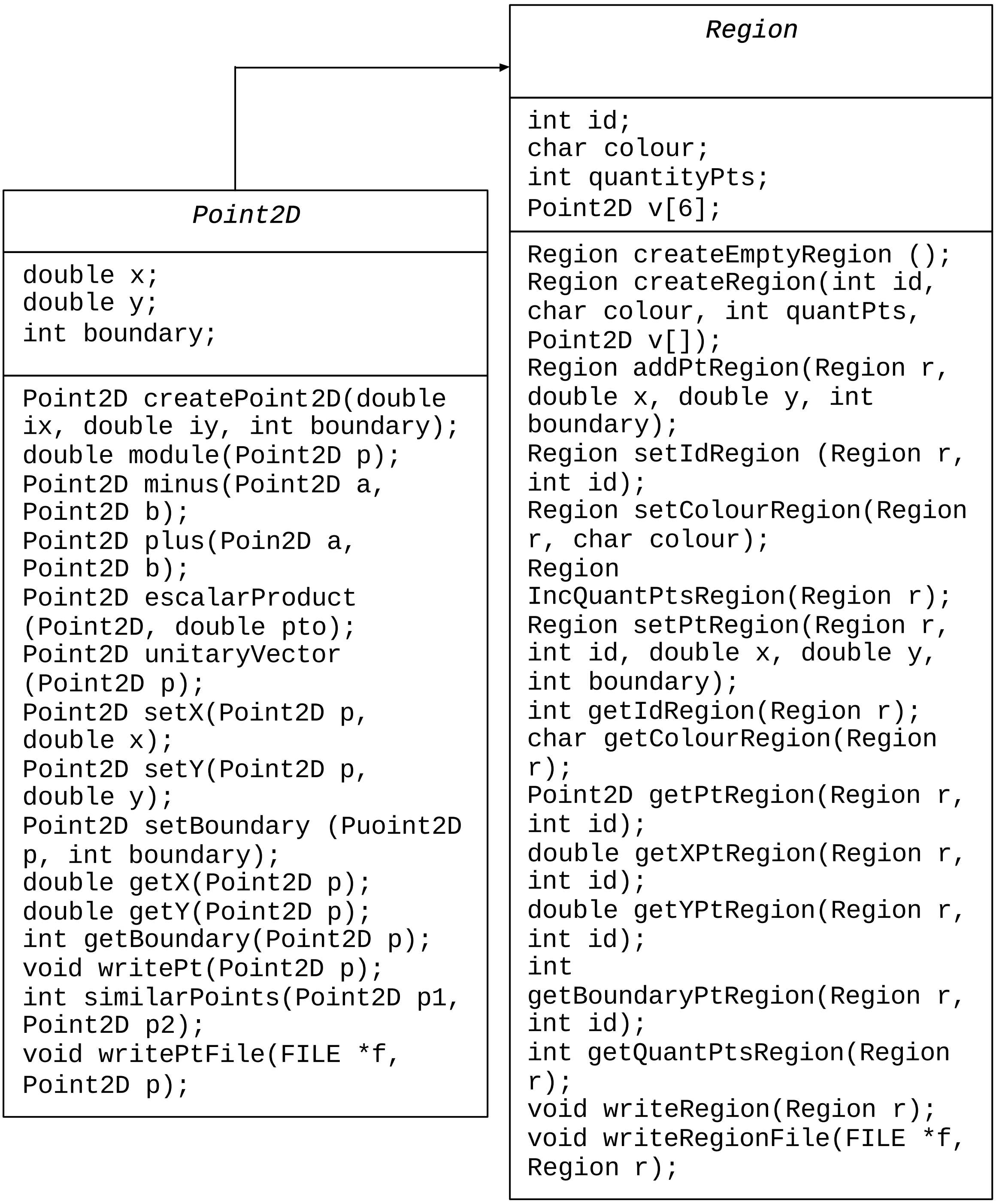
$$\text{incremento\_y} = \text{sen}(a) \cdot \text{desplazamiento\_encoders}$$

$$a = \text{ángulo de giro total}$$

To calculate these coordinates with the above formula, the robot must remember the coordinate of the last corner that it visited and the total angle that it has turned from the start of its movement.

Another important aspect is the detection of corners which are boundaries between two regions. This is the case of corners connecting rooms with corridors. In this case, the robot has to store the calculated coordinate and to indicate that it is a boundary coordinate. Later this information will be used to construct a final topological map coherent with the world.

The points are stored in a vector of registers, and each register stores a *Point2D* structure (figure 6.8), which is composed of the *x* and *y* coordinates calculated, plus one attribute to indicate if the point is a boundary point or not.



```
Point2D

double x;
double y;
int boundary;

Point2D createPoint2D(double
ix, double iy, int boundary);
double module(Point2D p);
Point2D minus(Point2D a,
Point2D b);
Point2D plus(Poin2D a,
Point2D b);
Point2D escalarProduct
(Point2D, double pto);
Point2D unitaryVector
(Point2D p);
Point2D setX(Point2D p,
double x);
Point2D setY(Point2D p,
double y);
Point2D setBoundary (Puoint2D
p, int boundary);
double getX(Point2D p);
double getY(Point2D p);
int getBoundary(Point2D p);
void writePt(Point2D p);
int similarPoints(Point2D p1,
Point2D p2);
void writePtFile(FILE *f,
Point2D p);
```

```
Region

int id;
char colour;
int quantityPts;
Point2D v[6];

Region createEmptyRegion ();
Region createRegion(int id,
char colour, int quantPts,
Point2D v[]);
Region addPtRegion(Region r,
double x, double y, int
boundary);
Region setIdRegion (Region r,
int id);
Region setColourRegion(Region
r, char colour);
Region
IncQuantPtsRegion(Region r);
Region setPtRegion(Region r,
int id, double x, double y,
int boundary);
int getIdRegion(Region r);
char getColourRegion(Region
r);
Point2D getPtRegion(Region r,
int id);
double getXPtRegion(Region r,
int id);
double getYPtRegion(Region r,
int id);
int
getBoundaryPtRegion(Region r,
int id);
int getQuantPtsRegion(Region
r);
void writeRegion(Region r);
void writeRegionFile(FILE *f,
Region r);
```

***Figure 6.8.*** *Attributes and methods of the structure of data defined for the topological map.*

#### 6.1.1.3 Creating the final Topological Map: Joining the Map of Colours and the Map of Points.

Once we have the map of colours and the map of points, joining both maps the robot will have the final topological map of its environment. The final topological map is created with a register vector where each register contains a structure called *Region* (figure 6.8). A Region is made of a region identifier, the colour of the region and a Point2D structure. The last structure stores the coordinates of the corners of the region.

In order to join the two maps that have been constructed, we know that the map of colours stores the correlative colours of the regions that make the final topological map, but the points in the map of points are not correlative because they have been stored while the robot was moving. An example of a virtual world and the stored coordinates in the map of points is shown in figure 6.9.

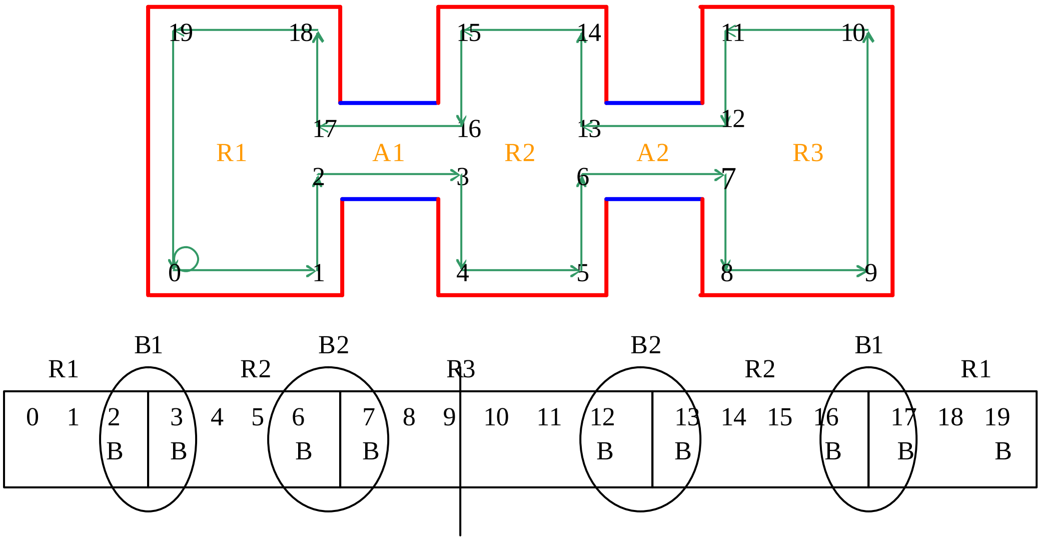

19
18
15
14
11
10
17
16
13
12
R1
A1
R2
A2
R3
2
3
6
7
0
1
4
5
8
9
B1
B2
B2
B1
R1
R2
R3
R2
R1
0 1 2 B 3 B 4 5 6 B 7 B 8 9 10 11 12 B 13 B 14 15 16 B 17 B 18 19 B

***Figure 6.9.*** *Virtual World covered by the robot and map of points obtained (B means that the point is a boundary point).*

Studying figure 6.9 we can classify the coordinates of the map of points in the corresponding regions such as:

1. Corridors (blue region B1 and B2) are made only by boundary points, and these points delimitate also rooms (red regions). Therefore, when we read the map of points and we find a boundary point, we know that a corridor starts, and the next boundary point will indicate that the corridor has finished and a new region starts, which is a room.

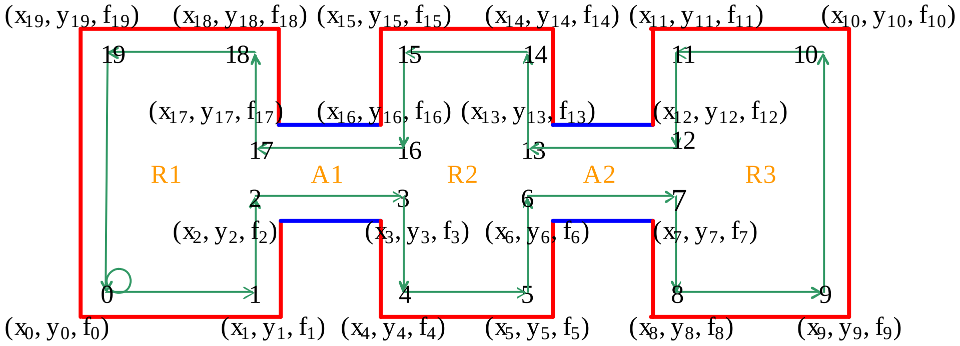
2. As the world is symmetric with respect to the X axis, the map of points obtained, is also symmetric and the points of which the regions are composed, are placed in symmetric positions with respect to the centre of the vector.

3. Then, using the information of points 1 and 2, the following algorithm stores the points in their corresponding regions of the final topological map (algorithm 6.1). The algorithm reads half of the map of points, and it stores the points of the initial region of the topological map and it increases the identifier of the region each time it finds a boundary point. Finally it reads the rest of points in the map of points (from the middle to the end). It stores the points that we find as final region and it decreases the region identifier each time it finds a boundary point.

```
Procedure construct_map(Region mapReg, int size_region_map,
Point2D mapPts, int size_points_map, string mapColours)
{
     int i; /*Index to visit the map of points */
     int indexRegions=0; /* Index to visit the topological Map*/
     mapReg[indexRegions] ← store region identifier (value
of indexRegions);
     mapReg[indexRegions] ← store colour of the region (from
map of colours);
     for (i←0; i<( size_points_map /2); i←i+1)
     {
         mapReg[indexRegions] ← store point (x,y, boundary
attribute);
         If (point.boundary = true) // It is a boundary point
         {
          indexRegions ← indexRegions +1;
          mapReg[indexRegions] ← store region identifier
(value indexRegions);
          mapReg[indexRegions] ← store colour of the region
(from mapColours);
          mapReg[indexRegions] ← store point (x,y,boundary
attribute) from mapPts;
         }
     }
     for (i←( size_points_map /2); i< size_points_map;
          i←i+1)
     {
             mapReg[indexRegions] ←store point
(x,y,boundary attribute) from mapPts;
         If (point.boundary = true) // It is a boundary point
         {
          indexRegions ← indiceRegiones-1;
          mapReg[indexRegions] ←  store point (x,y,boundary
attribute) from mapPts;
         }
     }
}
```

***Algortihm 6.1.*** *Algorithm for joining the map of colours and the map of points and obtaining the final topological map.*

To conclude, the topological map for the virtual world of the figure 6.9 would be the one in figure 6.10.

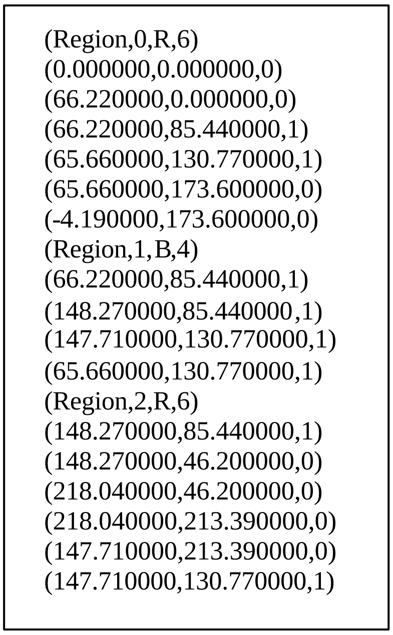


$(x_{19}, y_{19}, f_{19})$
$(x_{18}, y_{18}, f_{18})$
$(x_{15}, y_{15}, f_{15})$
$(x_{14}, y_{14}, f_{14})$
$(x_{11}, y_{11}, f_{11})$
$(x_{10}, y_{10}, f_{10})$
19
18
15
14
11
10
$(x_{17}, y_{17}, f_{17})$
$(x_{16}, y_{16}, f_{16})$
$(x_{13}, y_{13}, f_{13})$
$(x_{12}, y_{12}, f_{12})$
17
16
13
12
R1
A1
R2
A2
R3
2
3
6
7
$(x_2, y_2, f_2)$
$(x_3, y_3, f_3)$
$(x_6, y_6, f_6)$
$(x_7, y_7, f_7)$
0
1
4
5
8
9
$(x_0, y_0, f_0)$
$(x_1, y_1, f_1)$
$(x_4, y_4, f_4)$
$(x_5, y_5, f_5)$
$(x_8, y_8, f_8)$
$(x_9, y_9, f_9)$

| Id. Region: 1 | Id. Region: 2 | Id. Region: 3 | Id. Region: 4 | Id. Region: 5 |
|---|---|---|---|---|
| Colour: Red | Colour: Blue | Colour: Red | Colour: Blue | Colour: Red |
| Points: $(x_0,y_0,f_0)$<br>$(x_1,y_1,f_1)$<br>$(x_2,y_2,f_2)$<br>$(x_{17},y_{17},f_{17})$<br>$(x_{18},y_{18},f_{18})$<br>$(x_{19},y_{19},f_{19})$ | Points: $(x_2,y_2,f_2)$<br>$(x_3,y_3,f_3)$<br>$(x_{16},y_{16},f_{16})$<br>$(x_{17},y_{17},f_{17})$ | Points: $(x_3,y_3,f_3)$<br>$(x_4,y_4,f_4)$<br>$(x_5,y_5,f_5)$<br>$(x_6,y_6,f_6)$<br>$(x_{13},y_{13},f_{13})$<br>$(x_{14},y_{14},f_{14})$<br>$(x_{15},y_{15},f_{15})$<br>$(x_{16},y_{16},f_{16})$ | Points: $(x_6,y_6,f_6)$<br>$(x_7,y_7,f_7)$<br>$(x_{12},y_{12},f_{12})$<br>$(x_{13},y_{13},f_{13})$ | Points: $(x_7,y_7,f_7)$<br>$(x_8,y_8,f_8)$<br>$(x_9,y_9,f_9)$<br>$(x_{10},y_{10},f_{10})$<br>$(x_{11},y_{11},f_{11})$<br>$(x_{12},y_{12},f_{12})$ |

***Figure 6.10.*** *Sketch of the final topological map for the virtual World in figure 6.9, where ($x_i$, $y_i$) means the coordinates x and y of each point in the map, and $f_i$ represents the Boolean value indicating if the point is a boundary point between regions or not. In this example this coordinate will be true only for the points number 2, 3, 6, 7, 12, 13, 16, and 17.*

The topological map is written in a text file as the one in the example of the figure 6.11.



```
(Region,0,R,6)
(0.000000,0.000000,0)
(66.220000,0.000000,0)
(66.220000,85.440000,1)
(65.660000,130.770000,1)
(65.660000,173.600000,0)
(-4.190000,173.600000,0)
(Region,1,B,4)
(66.220000,85.440000,1)
(148.270000,85.440000,1)
(147.710000,130.770000,1)
(65.660000,130.770000,1)
(Region,2,R,6)
(148.270000,85.440000,1)
(148.270000,46.200000,0)
(218.040000,46.200000,0)
(218.040000,213.390000,0)
(147.710000,213.390000,0)
(147.710000,130.770000,1)
```

***Figure 6.11.*** *Example of the File with the topological map.*

The information of the topological map is stored in the file following the next format (figure 6.11):

- Each line of the file starts and finishes with a bracket and it contains several elements separated by commas.

- The start point of a region is when we find the symbol *Region.* After this symbol the identifier and the colour of the region appear (R means red and B means blue). The last element is the number of points which are part of the region.

- After this line, there are several lines, as much as the number of points that the region has. Each one of these lines contains the coordinate *x* and *y* and the Boolean value indicating if the point is a boundary point or not (1 means true).

### 6.1.2 Planning the movement

In this section we apply the qualitative movement theory presented in chapter 5.

First we infer the sequence of topological relations between the robot (seen as a region) and each region of its environment in order to navigate from one initial region to the goal one. In this application all the regions involved are considered areas. In order to plan the movement we will centre our attention to the time relation *next* because we consider that the time progresses when the topological situation between the robot and its environment changes.

The information we need to plan the movement is the relation between the robot, considered as an area region, and the initial region of the environment; and the relation between the robot and the next region of the environment that the robot will find up to the moment that the robot finishes its goal and is in the topological situation that is our goal.

We know also that the topological relation between the regions of the simulated world where the robot is sutiated, is always *Touch* or *Disjoint*.

With all this information we start the inference process. An example of this process is shown in figure 6.12. This example is the one that the Khepera[2] robot will perform to get the sequence of topological situations to go from the region 1 (the red room more at the left) to the region 3 (the red room more to the right) in the environment of figure 6.1d.

**Initial Situation:** Robot CI R1

Robot CI R1 & R1 T R2 $\xrightarrow{\text{AAA}}$ $\boxed{\text{Robot D R2}}$

**Inference 1:** Robot CI R1 $\xrightarrow{\text{Next}}$ $\boxed{\text{Robot TFI R1}}$

Robot TFI R1 & R1 T R2 $\xrightarrow{\text{AAA}}$ Robot {T, D} R2 → $\boxed{\text{Robot T R2}}$

**Inference 2:** Robot TFI R1 $\xrightarrow{\text{Next}}$ Robot {O,CI} R2 → $\boxed{\text{Robot O R1}}$

Robot O R1 & R1 T R2 $\xrightarrow{\text{AAA}}$ Robot {T, D, O, TFIi, CIi} R2 → $\boxed{\text{Robot O R2}}$

**Inference 3:** Robot O R1 $\xrightarrow{\text{Next}}$ Robot {T,TFI} R1 → $\boxed{\text{Robot T R1}}$

Robot T R1 & R1 T R2 $\xrightarrow{\text{AAA}}$ Robot {T, D,O,E,TFI,TFIi} R2 → $\boxed{\text{Robot TFI R2}}$

$\Rightarrow$ $\boxed{\text{R1 T R2} \rightarrow \text{R2 T R3}}$

**Inference 4:** Robot TFI R2 $\xrightarrow{\text{Next}}$ Robot {O,CI} R2 → $\boxed{\text{Robot CI R2}}$

Robot CI R2 & R2 T R3 $\xrightarrow{\text{AAA}}$ $\boxed{\text{Robot D R3}}$

**Inference 5:** Robot CI R2 $\xrightarrow{\text{Next}}$ $\boxed{\text{Robot TFI R2}}$

Robot TFI R2 & R2 T R3 $\xrightarrow{\text{AAA}}$ Robot {T, D} R3 → $\boxed{\text{Robot T R3}}$

**Inference 6:** Robot TFI R2 $\xrightarrow{\text{Next}}$ Robot {O,CI} R2 → $\boxed{\text{Robot O R2}}$

Robot O R2 & R2 T R3 $\xrightarrow{\text{AAA}}$ Robot {T, D,O,TFIi, CFIi} R2 → $\boxed{\text{Robot O R3}}$

**Inference 7:** Robot O R2 $\xrightarrow{\text{Next}}$ Robot {T,TFI} R2 → $\boxed{\text{Robot T R2}}$

Robot T R2 & R2 T R3 $\xrightarrow{\text{AAA}}$ Robot {T, D,O,E,TFI,TFIi} R2 → $\boxed{\text{Robot TFI R3}}$

**Inference 8:** Robot TFI R3 $\xrightarrow{\text{Next}}$ $\boxed{\text{Robot CI R3}}$

**Final Situation:** Robot CI R3

***Figure 6.12.*** *Example of an Inference process.*

In the figure 6.12, each arrow means the process of inference, that is, the composition of the relations before each arrow applying the composition table indicated above each arrow. From this figure, the sequence of topological situations between the robot and its environment results:

1. Initially we know that the robot is *Completely Inside* (CI) of the region 1, region 1 is *Touching* (T) the region 2, so using the composition table AAA-table we obtain the relation between the robot and region 2 initially, which is *Disjoint* (D).

2. Inference 1 uses the information obtained in the step number 1 and we deduce the situation of the robot with respect the region 1 in the *next* moment of time. This means to apply the composition movement table AAt-table and we obtain that the robot will be *Touching From Inside* (TFI) region 1. Using the new information and all the information we already know, we deduce the relation between the robot and region 2 in the *next* moment of time too. This is made once more using the composition table AAA-table, and we obtain that the robot will be *Disjoint* (D) or *Touching* (T) of the region 2. And as we also know that a new point of time appears when the topological relation between the robot and its environment changes, we can rule out the relation *Disjoint*, which is the same situation between the robot and region 2 in the step 1. Therefore the robot and the region 2 are *Touching* (T).

3. During the inference 2 and doing the process described in above steps we infer that the robot is *Completely Inside* (CI) or *Overlapping* (O) the region 1. As CI is the topological relation obtained in step 1 it will mean that the robot is coming back to its initial situation. Therefore we rule it out and the inferred relation between the robot and region 1 is O. With respect with the region 2, we initially infer that the robot can be T, D, O, TFIi, CIi the region 2. TFIi and CIi are ruled out because it is not possible that there are regions in which the robot can move that are inside the robot. T and D are also ruled out because they correspond to previous situations and then the robot would be coming back to the initial situation. Thus, the relation between the robot and region 2 is O.

4. In the inference 3, the relation between the robot and the region 1 could be TFI or T. TFI is ruled out because it is the relation if the inference 1. So, the robot is *Touching (*T) the region 1. The relation between the robot and the region 2 can be T, D, O, E, TFI, or TFIi. The relations T, D, O are ruled out because they belong to previous situations, TFIi is ruled out because it cannot be a region inside the robot. The relation *Equal* (E) is also ruled out because the robot cannot be equal to a region through which the robot has to move. Therefore, the topological relation between the robot and region 2 is TFI. At this moment, as the robot is already inside region 2,

to follow with the inference process we have to consider that the region 2 is *Touching* region 3.

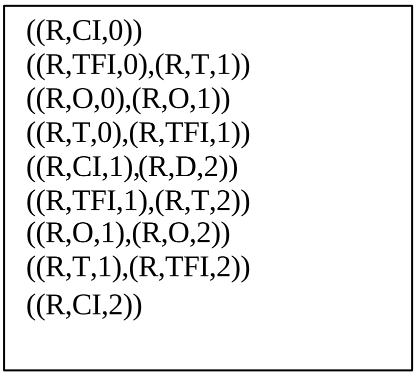
5. In the inference 4, we start the inference process with the region 2, and not with the region 1 because the robot has left it. Then we infer that the robot can be in CI or O region 2. O is ruled out. And the relation between the robot and region 3 is *Disjoint* (D).

From this point, the inferences are the same as the above one but with the region 2 and 3 and not the region 1 and 2. Therefore, the process is cyclic and we can define an automatic algorithm for the inference process (algorithm 6.2), useful for any region and for any initial and final region. This algorithm does not finish until the robot is in the final situation.

On the other hand, the resulting topological relations are written in a file as the one given in figure 6.13, which corresponds to the inference process of figure 6.13. This file will be useful to extract the sequence of topological relation for the planned movement and compare them with the real topological relations in each moment.

```
Procedure inference_process (num_initial_region,
initialSituation, num_final_region, finalSituation, exitNEXT,
exitAAA)
{
     inputNEXT ← initialSituation;
     while (num_initial_region ≠ num_final_region) or
(initialSituation ≠ finalSituation)
     {
     inference_table_Next(inputNEXT, setExitNEXT);
     if (quantity_components(setExitNEXT) > 1)
          select_relation_NEXT(setExitNEXT, lastInputNEXT,
exitNEXT);
     else
          obtain_component(setExitNEXT, 1, exitNEXT);

     inference_table_AAA(exitNEXT, setExitAAA);
     if (quantity_components(setExitAAA) > 1)
          select_relation_AAA(setExitAAA, lastExitAAA,
beforeLastExitAAA,
                              exitAAA);
     else
          obtenain_component(setExitAAA, 1, exitAAA);

     write_inferences_file(exitNext, exitAAA,
num_initial_region, num_initial_region+1);

        /*Store last input value of NEXT table*/
     lastInputNEXT ← inputNEXT;
     /* New input for NEXT table = exitNEXT always that we have
not find a new room (exitAAA != TFI*/
     if (exitAAA == "TFI")
          inputNEXT ← "TFI";
          num_initial_region ← num_initial_region +1;
     else
          inputNEXT ← exitNEXT;

     /* Store last results of  table AAA*/
     beforeLastExitAAA ← lastExitAAA;
     lastExitAAA ← exitAAA;
     }
}
```

**Algorithm 6.2.** *Algorithm for the Inference Process*



```
((R,CI,0))
((R,TFI,0),(R,T,1))
((R,O,0),(R,O,1))
((R,T,0),(R,TFI,1))
((R,CI,1),(R,D,2))
((R,TFI,1),(R,T,2))
((R,O,1),(R,O,2))
((R,T,1),(R,TFI,2))
((R,CI,2))
```

***Figure 6.13.*** *Example of the file created with the movement plan.*

### 6.1.3 Autonomous Robot Navigation using the Plan of Movements created.

In this section we explain the movement of the robot from one room to the other by using the planning constructed in previous sections. During its navigation the robot checks its movement by comparing its real topological relation in a moment of time *T* with the corresponding topological relation between the robot and its environment during the same moment of time *T* stored in the movement plan. We also will show examples of correct navigation with respect to the plan and of erroneous navigation.

Let’s start by explaining how the robot compares its topological situation with the one in the plan.

The topological relations of the robot with its environment are given by a qualitative symbol representing the topological situation and the numerical identifier of the region with which the robot shares this situation.

- To determine the numerical identifier of the room while the robot is moving we follow the next process:

  1. First the numerical identifier is initialised to 0.

  2. The identifier is increased after each turn from a room to a corridor or from a corridor to a room (figure 6.14), and this is done while the identifier is lower of the total number of regions minus 1 (because the identifier starts in 0).

  3. If the numerical identifier is the identifier of the last room of the environment, that means it is equal to the total number of regions minus 1, the robot starts the way back to the initial room again. Therefore the identifier is not incremented

but it is decremented when the robot turns from a room to a corridor or vice versa, up to the moment it is equal to 0 again, in which we start with the first step again (it is a cyclical process).

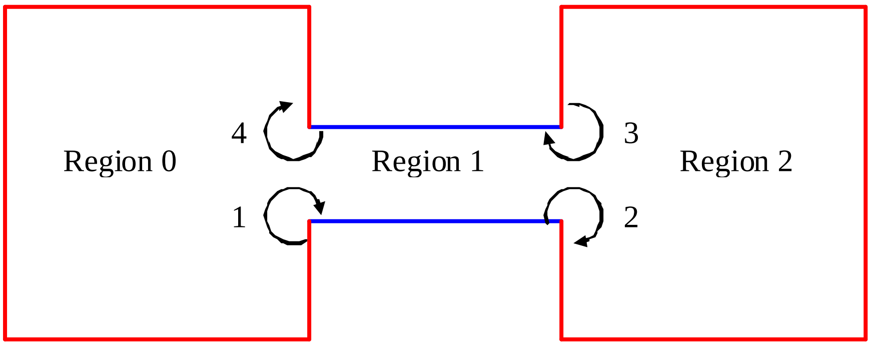


***Figure 6.14.*** *Turns of the robot when navigating from a room to a corridor or vice versa following the wall of the world.*

In order to avoid the robot turning around and going back to the same room due to an obstacle whilst it acts as if it were in a new room, we use the coordinates of each region stored in the topological map as follows:

In each situation in which the robot is moving from a room to a corridor (figure 6.14) we compare:

1. The coordinates obtained in the turns numbered 1 and 2 of figure 6.14 are compared with the coordinates of the initial point of regions 1 and 2 stored in the topological map respectively.

2. The coordinates obtained in the turns 3 and 4 of the figure 6.14 with the coordinates of the last point of the regions 2 and 1 stored in the topological map respectively.

In order to avoid the robot turning around and going back to the same room due to an obstacle, whilst it believes that it is in a new room, the robot compares the coordinates that it is calculating during its movement with the coordinates of each region stored in the topological map as follows:

- For the turns number 1 and 2 of figure 6.14, the robot compares the obtained coordinates with the stored ones of the initial point of the regions 1 and 2 respectively.

- For the turns number 3 and 4 of figure 6.14, the robot compares the obtained coordinates with the stored ones of the last point of the regions 2 and 1 respectively.

- To detect the different topological relations between the robot and the region where it has its current position and the adjacent region (figure 6.15) we follow the next process:

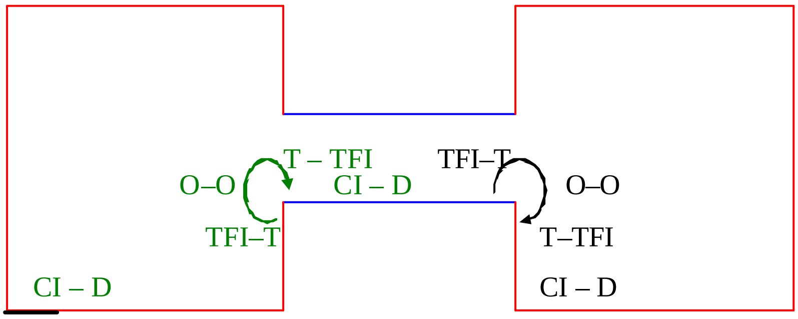


*Figure 6.15. Topological situations that the robot finds between it and the region in which it is and the next region.*

1. The topological relation CI between the robot and the region in which it is and the topological relation D between the robot and the next region that it will visit are determined once the robot has finished one of the turns described in figure 6.14 and in the first corner in which the robot detects red.

2. The topological relation TFI between the robot and the current region and T between the robot and next region are determined when the robot starts one of the turns of figure 6.14, which is determined when in these situations the robot has made 5 turns.

3. The topological relations O between the robot and the current and next regions are determined when the robot follows the turning movement in the situations of figure 6.14 (it has done more that seven turns).

4. Finally, the topological relation T between the robot and the current region and TFI between the robot and the next region is determined when the robot finishes the turning (one of them which are depicted in figure 6.14).

Then, each time that the robot reaches a new topological region, it compares the new situation with the corresponding one in the plan. For this comparison, the robot obtains the corresponding topological situation from the file created during the inference process. If both topological situations are equal, the robot follows its movement which is correct, otherwise it gets lost and it stops.

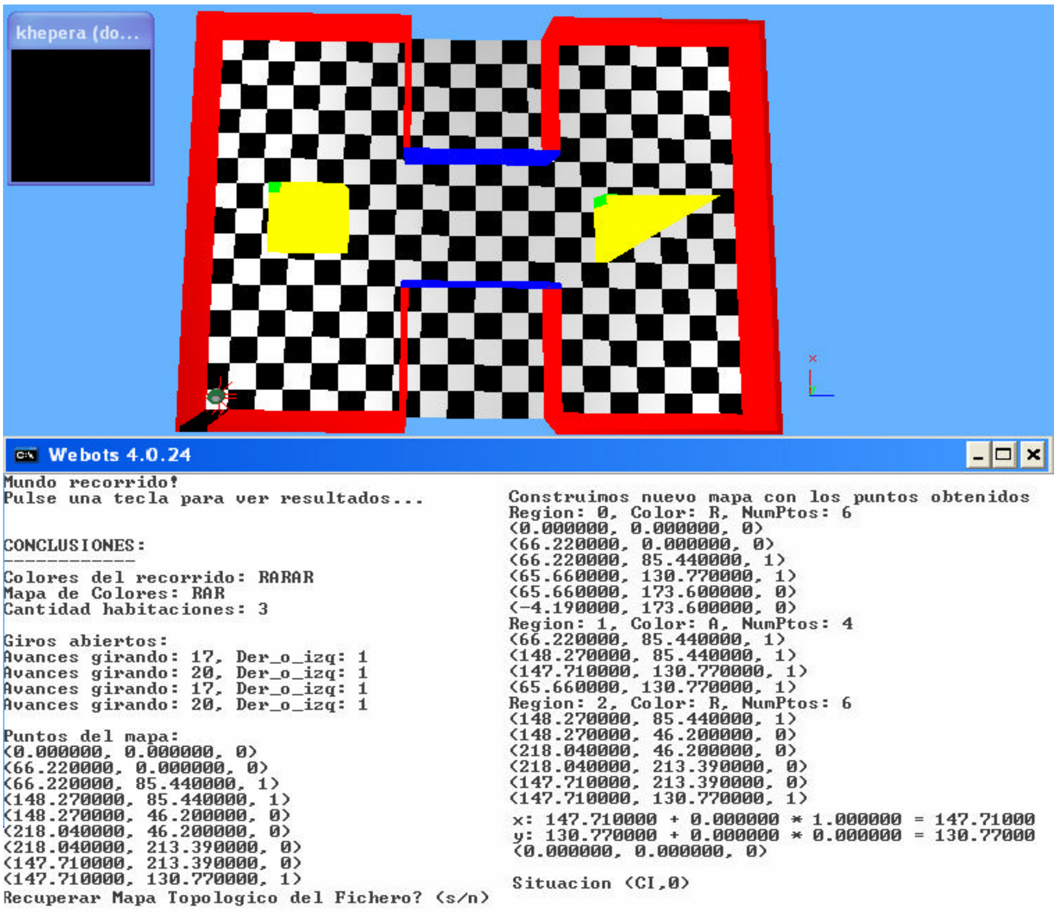
### 6.1.4 Examples of navigation using the Movement Plan

Once the robot has constructed its topological map, it knows that first of all it has to find the first object in the region 0, then the robot has to be always completely inside of the region 0, which means (R, CI, 0), and then it has to construct the qualitative description of the object. After the robot comes back to the initial point (black point) and knowing that CI is the region 0 and that CI has to be the region 2, because in this region it will find the second object for which the robot has to construct its qualitative description, it uses the topological map in order to deduce the movement plan as described in above section. Once it has the movement plan it should navigate from the region 0 to the region 2, detect its topological situation with respect to its environment and comparing the resulting situation with the corresponding one in the movement plan. In this section we are going to show two examples of this navigation from one region to the final one, the first one shows a correct navigation and the second one shows how the robot detects that it has moved in the wrong direction.

Finally, it should be remarked that when the robot has made both descriptions then it compares them and determines if the objects are equal or not, as we will see in section 6.1.5.

During the correct navigation of the robot from region 0 to region 2 the robot follows the following steps:

1. Figure 6.16 shows the initial situation of the robot, which is CI of the region 0, and shows also the topological map created.



```
Mundo recorrido!
Pulse una tecla para ver resultados...


CONCLUSIONES:
-------------
Colores del recorrido: RARAR
Mapa de Colores: RAR
Cantidad habitaciones: 3

Giros abiertos:
Avances girando: 17, Der_o_izq: 1
Avances girando: 20, Der_o_izq: 1
Avances girando: 17, Der_o_izq: 1
Avances girando: 20, Der_o_izq: 1

Puntos del mapa:
(0.000000, 0.000000, 0)
(66.220000, 0.000000, 0)
(66.220000, 85.440000, 1)
(148.270000, 85.440000, 1)
(148.270000, 46.200000, 0)
(218.040000, 46.200000, 0)
(218.040000, 213.390000, 0)
(147.710000, 213.390000, 0)
(147.710000, 130.770000, 1)
Recuperar Mapa Topologico del Fichero? (s/n)
```

```
Construimos nuevo mapa con los puntos obtenidos
Region: 0, Color: R, NumPtos: 6
(0.000000, 0.000000, 0)
(66.220000, 0.000000, 0)
(66.220000, 85.440000, 1)
(65.660000, 130.770000, 1)
(65.660000, 173.600000, 0)
(-4.190000, 173.600000, 0)
Region: 1, Color: A, NumPtos: 4
(66.220000, 85.440000, 1)
(148.270000, 85.440000, 1)
(147.710000, 130.770000, 1)
(65.660000, 130.770000, 1)
Region: 2, Color: R, NumPtos: 6
(148.270000, 85.440000, 1)
(148.270000, 46.200000, 0)
(218.040000, 46.200000, 0)
(218.040000, 213.390000, 0)
(147.710000, 213.390000, 0)
(147.710000, 130.770000, 1)
 x: 147.710000 + 0.000000 * 1.000000 = 147.71000
 y: 130.770000 + 0.000000 * 0.000000 = 130.77000
 (0.000000, 0.000000, 0)

 Situacion (CI,0)
```

***Figure 6.16.*** *Initial situation and creation of the topological map.*

Using the topological map and the initial relation between the robot and the region 0 (R,CI,0), and the final situation between the robot and region 2, the inference process described above determines the next movement plan:

- First the robot is CI region 0, which is represented by the string ((R,CI,0)).

- In the following change of time (when the topological situation between the robot and some region changes) we should have the relations ((R,TFI,0), (R,T,1)) between the robot and the region 0 and 1 respectively.

- In the next moment of time the relations should be ((R,O,0), (R,O,1)) meaning that the robot overlaps the region 0 and 1.

- Then, in the next moment the robot will share next relations with the region 0 and 1: ((R, T, 0), (R, CI, 1)).

- After the above situation the robot will be completely inside region 1 ((R,CI, 1)).

- At the next time change the topological situations will be ((R,TFI,1), (R,T,2)).

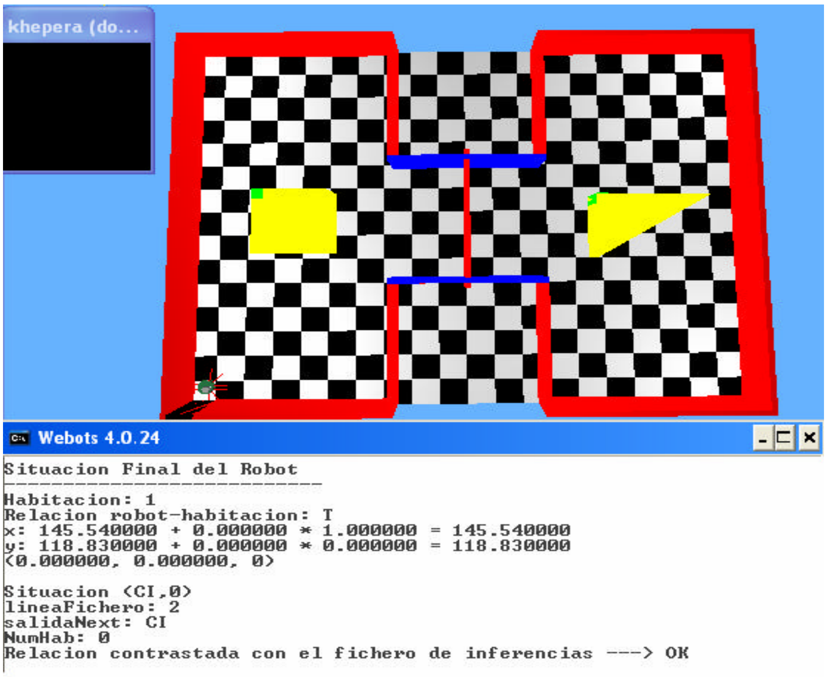
- Then, after the above situation the new situation during a correct navigation of the robot from region 0 to region 2 will be ((R,O,1), (R,O,2)).

- The robot is already in part of the region 2, therefore the next situation should be ((R,T,1), (R,TFI,2).

- Finally the robot will get the final topological situation, which was our goal, and it is (R,CI,2).

The above inference is stored in a file *(inference.txt)* and during the movement of the robot, when it detects a new topological situation, the file is used in order to compare the topological situation obtained with the corresponding one in this file. Therefore the rest of the steps are concerned with the detection of the new topological situations and the comparison described.

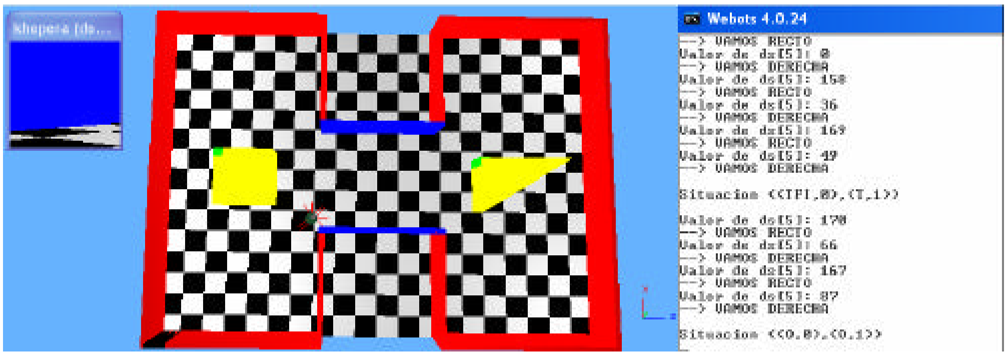
2. First the robot has to check if it is in the correct initial situation, which means (R,CI,0) (figure 6.17).



```
Situacion Final del Robot
---------------------------
Habitacion: 1
Relacion robot-habitacion: T
x: 145.540000 + 0.000000 * 1.000000 = 145.540000
y: 118.830000 + 0.000000 * 0.000000 = 118.830000
(0.000000, 0.000000, 0)

Situacion (CI,0)
lineaFichero: 2
salidaNext: CI
NumHab: 0
Relacion contrastada con el fichero de inferencias ---> OK
```

***Figure 6.17.*** *Correct checking, the robot is really (R,CI,0).*

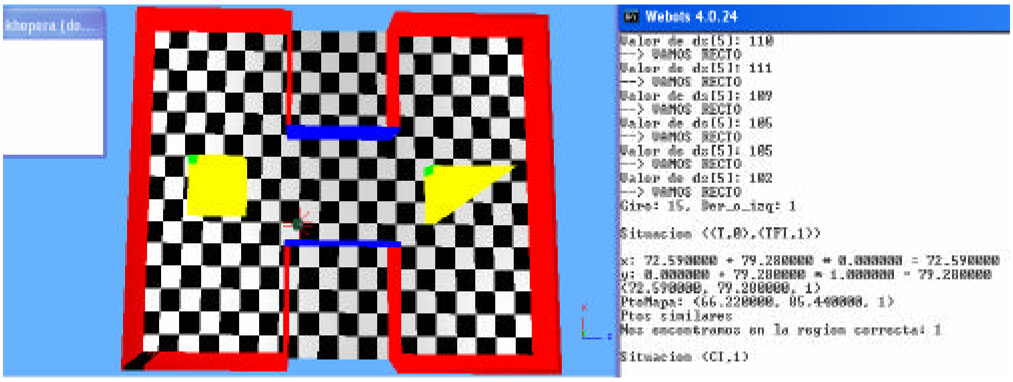
3. Then, when the robot is in a new topological situation, the robot detects which is its new situation and compares it with the situations in the file in the next moment of time. In this case it is correct and the robot continues its movement (figure 6.18) and after few steps the robot is (R,O,0) and (R,O,1).



```
--> VAMOS RECTO
Valor de ds[5]: 0
--> VAMOS DERECHA
Valor de ds[5]: 150
--> VAMOS RECTO
Valor de ds[5]: 36
--> VAMOS DERECHA
Valor de ds[5]: 169
--> VAMOS RECTO
Valor de ds[5]: 49
--> VAMOS DERECHA

Situacion ((TFI,0),(T,1))

Valor de ds[5]: 170
--> VAMOS RECTO
Valor de ds[5]: 66
--> VAMOS DERECHA
Valor de ds[5]: 167
--> VAMOS RECTO
Valor de ds[5]: 87
--> VAMOS DERECHA

Situacion ((O,0),(O,1))
```

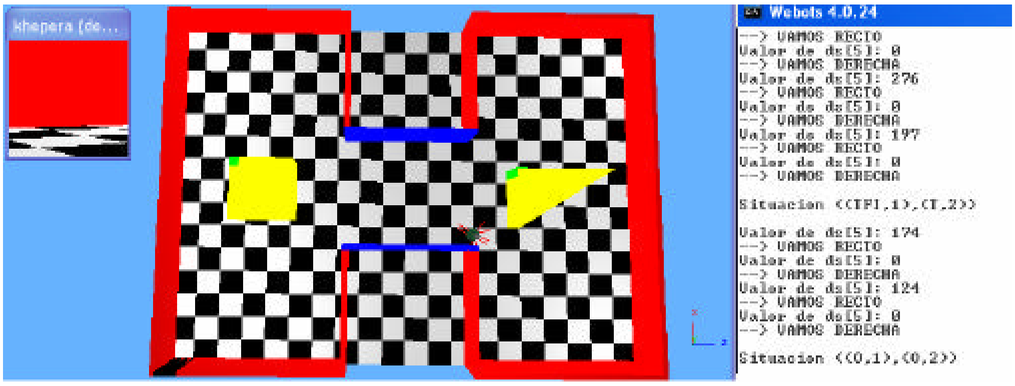
***Figure 6.18.*** *Correct checking, in this new moment of time the robot should be (R,TFI,0), (R,T,1) and after the next time, which it will be after few steps, it should be (R,O,0) and (R,O,1).*

4. The next figure (figure 6.19) shows the sequence of correct topological comparisons, therefore correct navigation up to the goal which in this case is that the robot is (R,CI,2).



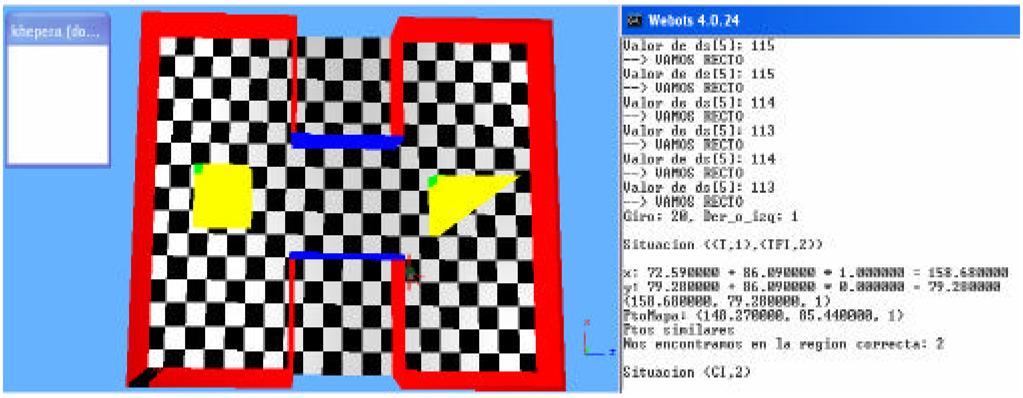
```
Valor de ds[5]: 110
--> VAMOS RECTO
Valor de ds[5]: 111
--> VAMOS RECTO
Valor de ds[5]: 109
--> VAMOS RECTO
Valor de ds[5]: 106
--> VAMOS RECTO
Valor de ds[5]: 105
--> VAMOS RECTO
Valor de ds[5]: 102
--> VAMOS RECTO
Giro: 15. Der_o_izq: 1

Situacion <<T,0>,<TFI,1>>

x: 72.590000 + 79.280000 * 0.000000 = 72.590000
y: 0.000000 + 79.280000 * 1.000000 = 79.280000
<72.590000, 79.280000, 1>
PtoMapa: <66.220000, 85.440000, 1>
Ptos similares
Nos encontramos en la region correcta: 1

Situacion <CI,1>
```

a) The robot is (R,T,0) and (R,TFI,1) and after few steps it is (R,CI,1).



```
--> VAMOS RECTO
Valor de ds[5]: 0
--> VAMOS DERECHA
Valor de ds[5]: 276
--> VAMOS RECTO
Valor de ds[5]: 0
--> VAMOS DERECHA
Valor de ds[5]: 197
--> VAMOS RECTO
Valor de ds[5]: 0
--> VAMOS DERECHA

Situacion <<TFI,1>,<T,2>>

Valor de ds[5]: 174
--> VAMOS RECTO
Valor de ds[5]: 0
--> VAMOS DERECHA
Valor de ds[5]: 124
--> VAMOS RECTO
Valor de ds[5]: 0
--> VAMOS DERECHA

Situacion <<O,1>,<O,2>>
```

b) The robot is (R,TFI,1) and (R,T,2) and after few steps the robot is (R,O,1) and (R,O,2).

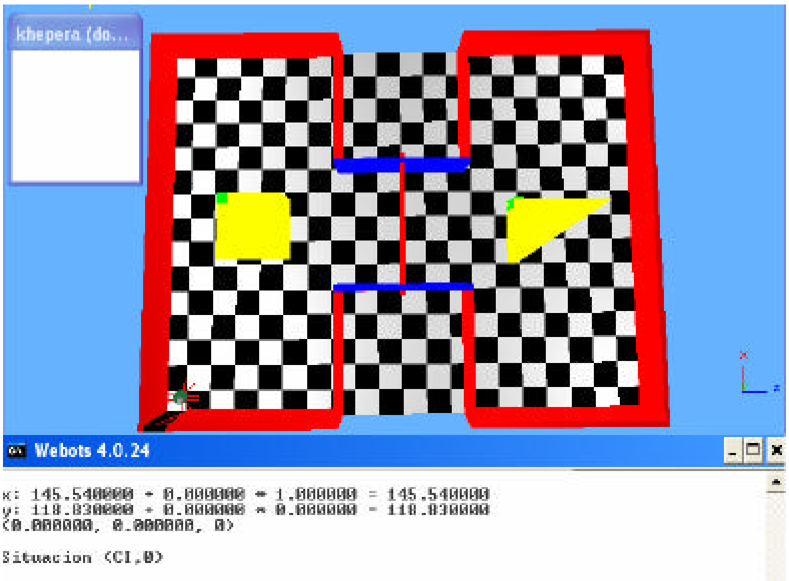


```
Valor de ds[5]: 115
--> VAMOS RECTO
Valor de ds[5]: 115
--> VAMOS RECTO
Valor de ds[5]: 114
--> VAMOS RECTO
Valor de ds[5]: 113
--> VAMOS RECTO
Valor de ds[5]: 114
--> VAMOS RECTO
Valor de ds[5]: 113
--> VAMOS RECTO
Giro: 20. Der_o_izq: 1

Situacion <<T,1>,<TFI,2>>

x: 72.590000 + 86.090000 * 1.000000 = 158.680000
y: 79.280000 + 86.090000 * 0.000000 = 79.280000
<158.680000, 79.280000, 1>
PtoMapa: <148.270000, 85.440000, 1>
Ptos similares
Nos encontramos en la region correcta: 2

Situacion <CI,2>
```

c) Finally, the robot is (R,T,1) and (R,TFI,2) and few steps later the robot detects (R,CI,2) which was our goal.

***Figure 6.19.*** *Sequence of correct topological situation up to our goal.*

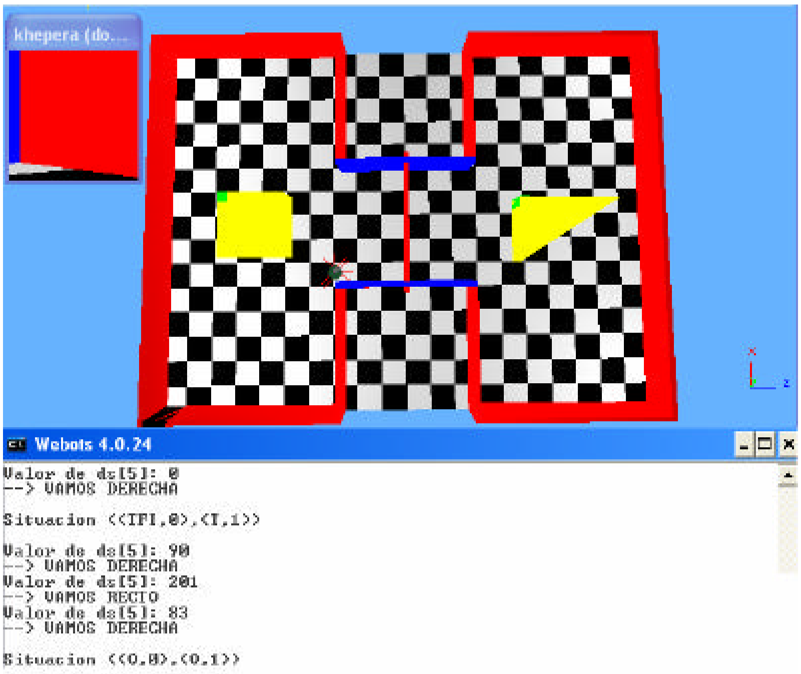
Let us to examine an example of the same na vigation, that uses the same initial and goal situations with the same inference file, but with a big obstacle that makes the robot gets lost. The robot will detect this situation because the comparison of a new topological situation between the robot and some region with the corresponding one in the inference file is not *true*. Figure 6.20 shows the world with the big obstacle in region 1, this figure also shows how the robot checks if it is CI region 0.



```
x: 145.540000 + 0.000000 * 1.000000 = 145.540000
y: 118.830000 + 0.000000 * 0.000000 = 118.830000
(0.000000, 0.000000, 0)

Situacion (CI,0)
```

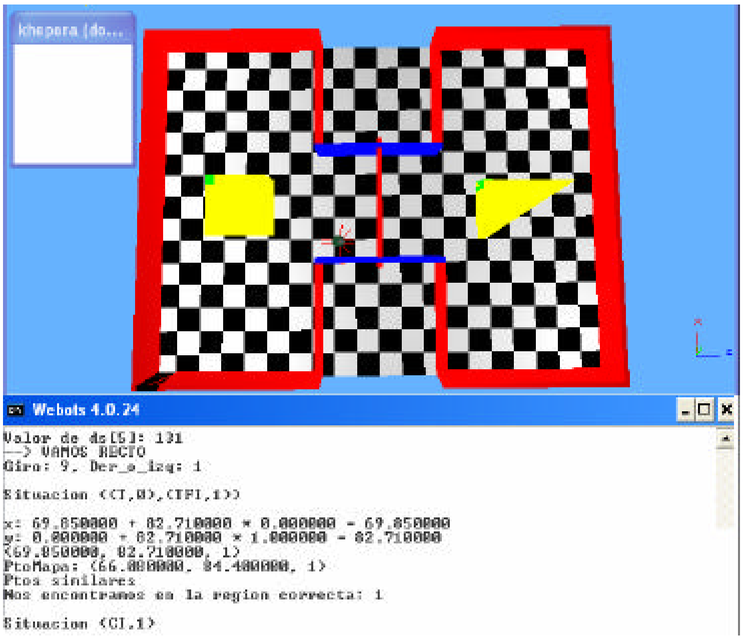
***Figure 6.20.*** *Environment of the robot with a big obstacle, and correct checking: the robot is CI region 0.*

As the first checking is correct the robot continues its movement and it makes the following comparisons:



```
Valor de ds[5]: 0
--> VAMOS DERECHA

Situacion ((IFI,0),(I,1))

Valor de ds[5]: 90
--> VAMOS DERECHA
Valor de ds[5]: 201
--> VAMOS RECTO
Valor de ds[5]: 83
--> VAMOS DERECHA

Situacion ((O,0),(O,1))
```

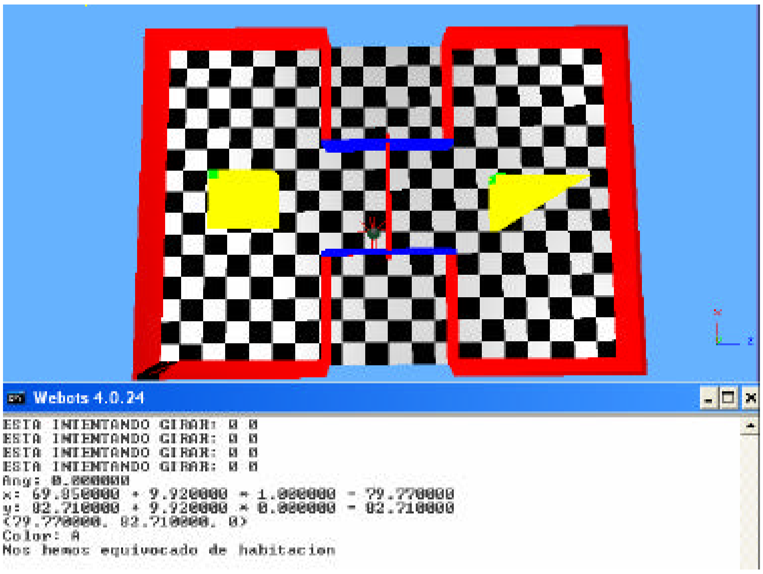
***Figure 6.21.*** *Correct situation: next time the robot is (R,TFI,0) and (R,T,1) and after few steps the robot is (R,O,0) and (R,O,1).*



```
Valor de ds[5]: 131
--> VAMOS RECTO
Giro: 9. Der_o_izq: 1

Situacion ((T,0),(TFI,1))

x: 69.850000 + 82.710000 * 0.000000 - 69.850000
y: 0.000000 + 82.710000 * 1.000000 - 82.710000
(69.850000, 82.710000, 1)
PtoMapa: (66.000000, 84.400000, 1)
Ptos similares
Nos encontramos en la region correcta: 1

Situacion (CI,1)
```

***Figure 6.22.*** *The robot detects a new correct situation, the robot is (R,T,0) and (R,TFI,1). After few steps the robot is (R,CI,1) which is also correct.*

Finally, when the robot finds the obstacle, it has to turn, this is the situation that makes the robot check if its topological relations with the environment have changed. As it is seeing blue, then it determines that it is still inside the region 1. Then it calculates the approximate coordinates of its current position and it compares them with the corresponding ones in the topological map. The comparison does not return *true* which means that the robot is not changing to the region 2, and it should be doing this movement. Therefore the situation is not the correct one and it determines that it has gone in the wrong direction.



```
ESTA INTENTANDO GIRAR: 0 0
ESTA INTENTANDO GIRAR: 0 0
ESTA INTENTANDO GIRAR: 0 0
ESTA INTENTANDO GIRAR: 0 0
Ang: 0.000000
x: 69.850000 + 9.920000 * 1.000000 = 79.770000
y: 82.710000 + 9.920000 * 0.000000 = 82.710000
(79.770000, 82.710000, 0)
Color: A
Nos hemos equivocado de habitacion
```

***Figure 6.23.*** *The robot has lost its direction of movement.*

### 6.1.5 Matching of Objects

This section explains how the robot describes the object qualitatively. In this application the objects will be always regular or non regular polygons, the object will not contain curves or holes. The case of holes has no sense in this application because the robot will be not able to detect the holes inside an object. The case of curves is an extension.

In order to facilitate the process of describing the objects, one of the corners of the object is marked with a green strip. When the robot, using the turret, detects a green strip then it knows that this is the first vertex of an object. The next step to take is to find all the vertices of the object, which has been made knowing that the robot has to turn at each corner, knowing this fact, the robots detects that it has found another vertex. The robot gives to the first vertex the (0,0) coordinates and using the relative distance of the displacement of the robot and the degrees of the rotation then the next (x,y) coordinates are calculated. With all the vertices of the object, the robot, by applying the qualitative theory described in section 5, constructs the qualitative description of the object. Next, the robot looks for the other object in the other room and repeats the process. When the robot has described both objects qualitatively, the matching algorithm is executed and the robot returns whether both objects are qualitatively equal or not. As the robot is running in a standalone mode to indicate that both objects are equal it revolves around itself.

## 6.2. APPLICATION NUMBER 2: INDUSTRIAL MOSAIC APPLICATION.

The goal of this application is the recognition of tiles in a mosaic design in order to allow the automatic and intelligent assembling of mosaics in the ceramic industry.

A mosaic is a form of decorative art in which small tiles (called tesseraes) are used to create a picture. Mosaics were used from the ancient times for domestic interior decoration. Mosaics are particularly associated with Roman dwellings, for example on floors, but the craft has continued through the ages, and many modern examples exist. There are two main methods of creating mosaics. They are commonly referred to as the direct method of mosaics construction and the indirect method of mosaic construction.

The *direct method of mosaic construction* involves directly placing (gluing) the individual tesserae onto the supporting surface. This method is well suited to surfaces which have a 3 dimensional quality such as vases.

But, the direct method suits small projects which are transportable. Another advantage of the direct method is that the resulting mosaic is progressively visible allowing for any adjustments to tile placing or colours to be done immediately. The disadvantages of the direct method are that the artist must work directly on the chosen surface which is often not practical for long periods of time. It is unsuitable for large-scale projects. Also, it is difficult to control the evenness of the finished surface. This is of particular importance when creating a functional surface such as a floor or a tabletop. If such qualities are important in the finished mosaic surface, then *indirect method of mosaic construction* may be more useful.

The *indirect method of mosaic* construction involves pre-assembling and arranging the tesseraes in several grids that later will be placed onto the supporting surface following a plan to create the final picture or pattern (figure 6.24). The individual tesseraes are placed on the grid according to a design for each grid, which will create the intended picture.

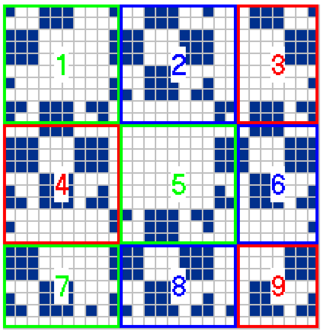
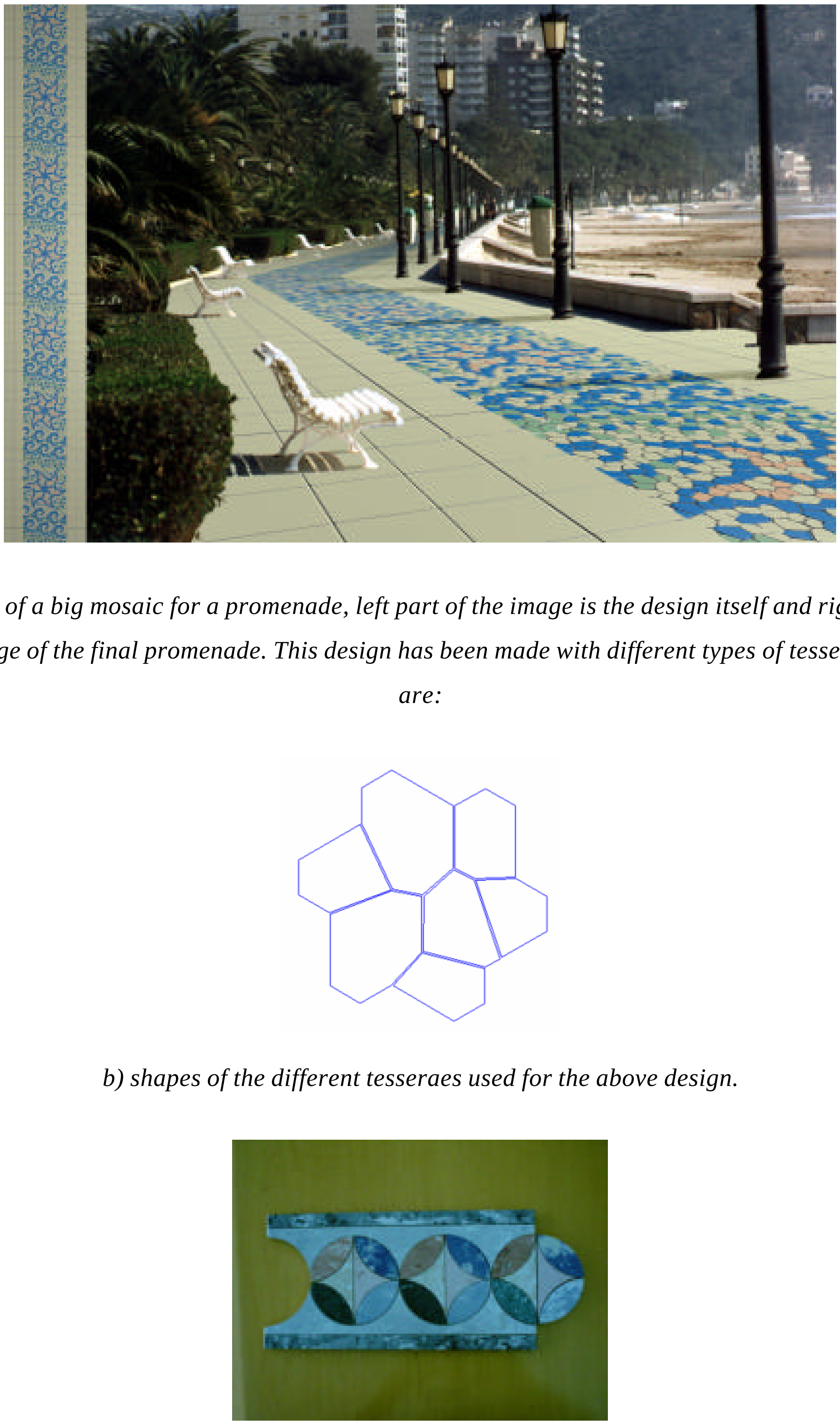


***Figure 6.24.*** *Plan to place the different grids to create a mosaic. Each square number represents a grid to be assembled.*

The application has been developed with the aim of being applied in a complete system to automate in an intelligent way the indirect method of mosaic construction, which means the process of pre-assembling the tesseraes and creating the different grids which have to be placed in the construction place. Therefore, the application consists of a software that recognise the tessereas and determines not only their position in a mosaic design but also the rotational angle that a robot arm has to make when picking up the tesserea from its centre (using suckers) to place the tesserea in the correct orientation. This software implements the qualitative theory for shape representation and matching presented in chapter 5. For this application a qualitative approach is the most suitable one because no two manufactured tiles or tesseraes are exactly identical, and working qualitatively we can manage the uncertainty.

Examples of different types of mosaics that we want to assemble automatically, can be seen in figure 6.25. The figure 6.25a) is a mosaic composed by a huge number of grids (b)) and c) represents a border mosaic made of one unique grid. These figures also show that a mosaic can be done with different types of tesseraes.



*a) Example of a big mosaic for a promenade, left part of the image is the design itself and right part is a virtual image of the final promenade. This design has been made with different types of tesseraes, which are:*

*b) shapes of the different tesseraes used for the above design.*

*c) Real border mosaic example, made of tesseraes of 4 different shapes.*

***Figure 6.25.*** *Examples of real mosaics.*

Given its final goal, the application has to interact with a vision system to obtain the images of the tesseraes to place, and a robot arm which will be the one in charge of placing that tesseraes in their correct position in each grid. Figure 6.26 shows a diagram of the final system in which the application will work.

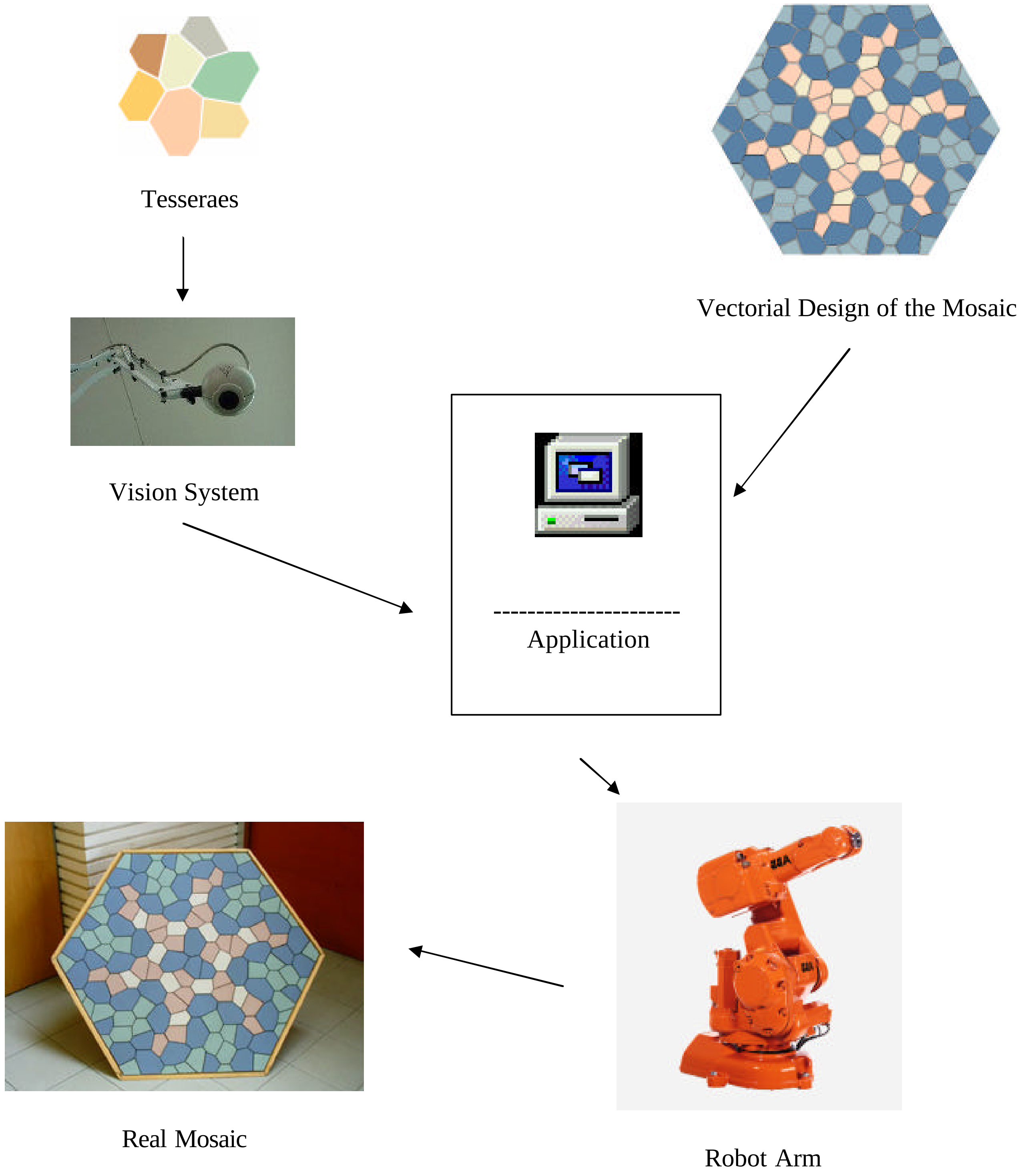


***Figure 6.26.*** *Diagram of the full system in which the application will be integrated.*

The application here presented has two entries, the vectorial design of a mosaic (figure 6.27 is an example) and a picture of one or more tesseraes which comes from the vision system (figure 6.28 is an example). The application requires the whole theory described in section 5, because it has to be able to recognise objects with holes, straight and curve segments in the same model, including also the information about the colour and the area of the objects to recognise. We are going to explain in these sections not only the matching software implemented but also how the pictures are processed to obtain the qualitative information that we need.

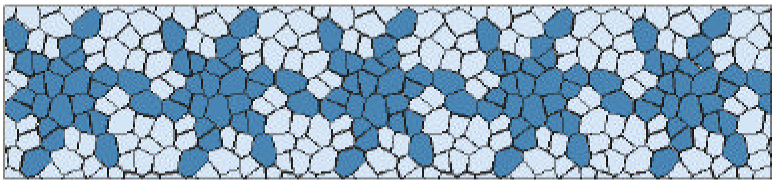


***Figure 6.27.*** *Example of a mosaic design.*

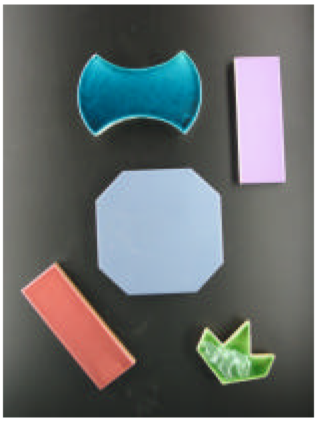


***Figure 6.28.*** *Example of a picture of several tessereas to be recognised.*

The application has been developed in four phases:

1. Development of a first application for the matching of polygonal regular and non-regular tesseraes, which are composed of straight lines and without holes.
2. Improvement of this first application adding the capability of recognising and matching of tesseraes including only one hole.
3. New improvement of the application including the matching of tesseraes with curves of completely curvilinear tesseraes in their boundaries and their holes's boundaries.
4. Improvement of the application for matching of tesseraes with several holes.

### 6.2.1 Constructing the Qualitative Description of a Mosaic Design.

In order to construct the qualitative description of each figure we need its relevant points. As we have stated in section 5, in the case of straight segments we need the vertices as relevant points, and in the case of curve segments we need the starting and final points of the curve and the point of maximum curvature. For the case of figures with holes we find the relevant points of the container and of the boundary of each hole in the figure.

Due to implementation reasons, we have numbered the relevant points always in the same way. The first point is the one in the upper-left hand corner of the object. The rest of points are numbered following a counter-clockwise order.

We have to extract the relevant points from two types of entries: the pictures of figures taken by a vision system and the design of the mosaic to be assembled.

We will explain the process to obtain the relevant points from pictures given by a vision system in subsection 6.2.2.

In the case of the design of the mosaic, it consists of a file with .AI extension (Adobe Illustrator file). These types of files are obtained when designing a mosaic with any vectorial design tool, as it could be Macromedia Freehand MX, the one used in the project. In a .AI file we have each object of the design defined by a series of points and indicating if the points belong to a curve or a straight line. Figure 6.29 shows an example of the part of a .AI file giving the vectorial description of a mosaic designed.

```
(Primer plano) Ln    //Starting point of the vectorial design
of the objects

300 Ar

1 XR //Starting point of the description for the first figure

0.8078 0.5765 0.3765 Xa    //Colour of the figure in RGB format

252.3002 489.3314 m        //First   point   of   the   figure,
indicated by the symbol “m” at the end of the line

363.8649 489.7667 L        //Lines finishing by the symbol “L”
//represent straight segments which the starting point is the
//one before this line and the end point is the one before the
//symbol “L”.

328.8179 319.5903 L
```

***Figure 6.29. Part I.*** *Example of the part of a .AI file with the vectorial description of the mosaic design.*

```
251.8649 358.7197 L
216.8179 423.155 L
252.3002 489.3314 L  //As they are closed polygons the first
//and the last points are always the same.
f               //Ending point of the first figure.
1 XR                //Starting point of the second figure
0.216 w
3.863678 M
[ ] 0 d   //This line indicates that the following is a
boundary.
259.5871 386.3039 m       //Boundary points, and as later we
have the same points
//for the figure itself we not consider these points, because
//they are the same as the ones describing the figure itself.
279.1211 379.5986 295.0553 386.1536 307.3886 405.9683 C
318.1631 425.8567 315.6703 445.9215 307.3399 465.7956 C
292.797 442.5224 L
294.4936 435.9872 293.6803 429.0534 290.3574 421.7218 C
283.535 410.8038 277.1242 407.163 271.1069 407.8114 C
259.5871 386.3039 L
f    //Ending point of the description of the boundary of the
//second figure, and the following are the description of the
//figure itself
1 XR
1 0 0.6 Xa      //Colour of the second figure in RGB format
259.5871 386.3039 m        //First point of the second figure
279.1211 379.5986 295.0553 386.1536 307.3886 405.9683 C    //If
//the line finishes with the symbol “C” then we are studying a
//curve segment, and the curve is given as a Bezier cubic curve,
//with its control points, and the end point, remember that the
//first point is always the one in the line above the actual one
//in the file.
318.1631 425.8567 315.6703 445.9215 307.3399 465.7956 C
292.797 442.5224 L
294.4936 435.9872 293.6803 429.0534 290.3574 421.7218 C
283.535 410.8038 277.1242 407.163 271.1069 407.8114 C
259.5871 386.3039 L
f               //Ending point of the second figure.
LB         //End of the vectorial description of figures.
```

***Figure 6.29. Part II.*** *Example of the part of a .AI file with the vectorial description of the mosaic design.*

To obtain the relevant points from the .AI file, first we have to consider only the part of the file describing the vectorial objects of the design, as the one shown in figure 6.29. In figure 6.29 we can observe the following:

- The starting point of the description of a figure is indicated by the symbol “XR”, and the ending point is indicated by the symbol “f”.
- The colour of each figure is given in RGB format and it is indicated in a line finishing by the symbol “Xa”.
- The first point (x, y coordinates) of each figure is always indicated by a line in the .AI file finishing by the symbol “m”.
- The rest of points (x,y coordinates) of the figure are written down in consecutive lines, indicated in separate lines of the file, each line finishing with the symbol “L” if the points belong to a straight segment or by the symbol “C” if the points belongs to a curved segment. In the case of curved segments the points written down are the control points and the end point of a cubic Bezier curve.
- In the case that the design of each figure is drawn with an additional boundary around it, the vectorial description of the boundary is written down before the vectorial description of the figure itself (the interior), and both descriptions have the same points, the only difference is that the boundary description starts with the line “[] 0 d”. So, in order to avoid obtaining two descriptions of the same figure, we ignore the description of the boundary and we consider only the description of the figure itself (the interior).

The process to obtain the relevant points in straight segments is simple because they are the points given by the .AI file itself, but this process is not so easy in the case of curved segments. As we have stated in the case of curved segments the .AI file represents the initial point of the curve, its control points and the end point, representing the curve as a cubic Bezier curve. But the relevant points needed to obtain the qualitative description of the figure are the initial and final points of the curve, which are given in the .AI file, and the point of maximum curvature of the curve. In order to obtain the point of maximum curvature we proceed as follows:

1. First we calculate the equation of the straight line passing through the first and last points of the curve, using the director vector joining both points.
2. Then, we calculate the distance in perpendicular between each point of the curve to the straight line calculated. The point whose distance to the straight line is bigger than the others is the point of maximum curvature. Each point of the curve is calculated with the equation of the cubic Bezier curve given by the four points in the file, as:

$$x = (1-u)^3\ x_i + 3u\ (1-u)^2\ x_{c1} + 3u^2\ (1-u)\ x_{c2} + u^3\ x_f$$

$$y = (1-u)^3\ y_i + 3u\ (1-u)^2\ y_{c1} + 3u^2\ (1-u)\ y_{c2} + u^3\ y_f$$

*where,*

$u$ = 0, 0.2, 0.4, ..1

*($x_{c1}$, $y_{c1}$) y ($x_{c1}$, $x_{c2}$) are the control points given in the .AI file*

*($x_i$, $y_i$) is the initial point given in the .AI file*

*($x_f$, $y_f$) is the final point given in the .AI file*

In graph form, figure 6.30 shows an example of three points of the curve and its distances to the straight line joining the starting and the ending point of the curve. In this figure the point of maximum curvature would be the one called $P_M$.

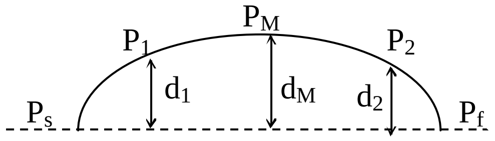


***Figure 6.30.*** *Graphical example of the calculus of the point of maximum curvature.*

Once we obtain the relevant points, we can construct the qualitative description of the figures in the mosaic, according to the theory described in section 8, proceeding with the following steps:

- The angle of each vertex is classified as acute, right-angled or obtuse. To determine the angle we calculate the distance between the vertex and the centre of the circle passing through the anterior and posterior vertices of the one we are calculating the angle for, as figure 6.31 indicates.

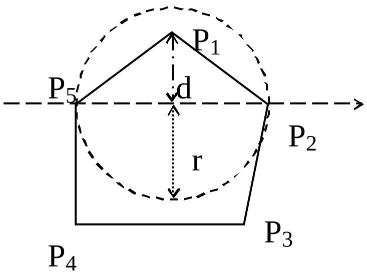


***Figure 6.31.*** *Calculus of the angle of the vertex P1, where d is the distance and r is the radius of the circle passing through P5 and P2 (anterior and posterior vertices of P1 respectively).*

According to the comparison between the distance (d) calculated and the radius of the circle we determine the angle as follows:

```
If d>radius then angle = acute
Else if d<radius then angle = obtuse
Otherwise angle= right-angled
```

- The edge between two consecutive vertices is classified as concave or convex. To determine the type of convexity associated with the vertex (x,y) we use the following equation, which calculates where the vertex is with respect to the straight line passing through the anterior ($x_0$,$y_0$) and posterior ($x_1$,$y_1$) vertices:

$$f(x,y) = (x-x_0)(y_1-y_0) - (y-y_0)(x_1-x0) \qquad (6.1)$$

```
Si f(x,y)> 0→ concave
Sino f(x,y)<0 → convex.
```

- The relative length of an edge would be the result of the comparison of its length and the length of the following edge. Both lengths are calculated using the Euclidean distance between two vertices. The result of the comparison will be one of the symbols in the set [less, equal, bigger].

The above steps are used when we are giving the qualitative description of the straight segments, but for the case of curve segments we need to calculate the type of convexity and the type of curvature of the curve. These features are calculated as follows:

- The type of convexity of the curve is determined as we have described for the case of straight segments but the reference points used in the equation (9.1) are the initial and final point of the curve. The qualitative representation is always associated only with the point of maximum curvature.

- The type of curvature ([acute, semicircle, plane]) is determined by calculating two distances: $d_a$ (distance between the initial point of the curve and the point $P_{ik}$, which is the one in the middle of the straight line joining the initial and the final point of the curve) and $d_b$ (distance between the point of maximum curvature and $P_{ik}$) as we have explained in section 5 (see figure 5.4 to remember). The point Pik is calculated simply by doing the next operation, where $(x_i,y_i)$ is the initial point of the curve and $(x_f,y_f)$ is the final point of the curve:

$$P_{ik}(x_{ik},y_{ik})=(\frac{x_f-x_i}{2},\frac{y_f-y_i}{2})$$

Distances $d_a$ and $d_b$ are the Euclidean distances between the initial point and $P_{ik}$ and between the point of maximum curvature and $P_{ik}$ respectively. Then we compare both distances and determine the type of curvature as follows:

```
If $d_a=d_b$ then type_curvature=semicircle
Else If $d_a<d_b$ then type_curvature=acute
Otherwise type_curvature=plane.
```

Each figure also stores its colour, obtained as RGB coordinates from the .AI file, and two symbols to indicate if the figure has holes and/or curves. The symbols are chosen from the sets [without-holes, with-holes], and [without-curves, with-curves, only-curves] respectively as soon as we detect that the figure has holes or curves. These symbols are introduced to speed up the matching process.

In the case that a figure contains holes, we repeat the above steps for each hole of the figure and we relate each hole with the container using the topological relation CIi and we calculate the relative orientation set of each hole with respect to its container as follows:

- First of all, as Frank's Reference System is placed at the centroid of the container with the NW orientated to the first vertex of the figure, we calculate the straight line through the centroid and the upper-left vertex of the container, which has been the one considered as the first vertex. This straight line will be the bisection line of the NW region. As each region covers 45º, for example, the bisection line plus 22,5º limits the starting line of the N region, and the bisection line minus 22,5º limits the starting line of the W region.

- Then, we calculate the straight lines between the centroid of the container an each vertex of the boundary of the hole.
- Finally we calculate the angle between the first straight line and each straight-line through the vertices of the hole, and in function of the value obtained for each angle ($\alpha_i$) we determine the orientation as follows:
  - If $\alpha_i \in$ [0º, 22,5º] or [337,5º,360º] → NW
  - Else if $\alpha_i \in$ (22,5º,67,5º] → N
  - Else if $\alpha_i \in$ (67,5º,112,5º] → NE
  - Else if $\alpha_i \in$ (112,5º,157,5º] → E
  - Else if $\alpha_i \in$ (157,5º,202,5º] → SE
  - Else if $\alpha_i \in$ (202,5º,247,5º)→S
  - Else if $\alpha_i \in$ (247,5º,292,5º] → SW
  - Else if $\alpha_i \in$ (292,5º,337,5º] →W

The set of regions obtained with each $\alpha_i$ angle (with i=1, …,k being k the number of vertices of the hole) represents the set of orientations that holds between the hole and the container.

Repeating the process described above we can obtain the qualitative description of all figures in the design, but as the .AI file does not have the points in a counter-clockwise order, the order of the points of each figure are the one followed by the designers when they make the mosaic design, what we do is to reorder the description of the vertices taking as the first vertex the one in the upper-left corner of each figure, using the (x,y) coordinate of each point that we still have and then we have the final qualitative description (a string) of all the figures in the mosaic design.

### 6.2.2 Constructing the Qualitative Description of the Figures in a Picture.

We have developed a hybrid (quantitative plus qualitative) method in order to extract the relevant features of the objects in an image, this is needed in order to use them to create the qualitative description of the objects following the qualitative description theory presented in section 5.

The development of a new method is obviously needed because we need to extract only the points defined by the theory as relevant points. This section shows how this task has been done. We classify the method developed as hybrid because initially we use an ordinary quantitative method to extract all the points in the boundary of each object in the image, and then these points are managed qualitatively in order to extract only the approximated relevant points. The quantitative algorithm chosen for the detection of the points of the boundary of a given image is the Canny algorithm [Canny 86].

#### 6.2.5.1 Quantitative Extraction of the Points of the Boundary of an Image.

Edges characterize boundaries and therefore their detection is a problem of fundamental importance in image processing. Edges in images are areas with strong intensity contrasts – a jump in intensity from one pixel to the next. Edge detecting significantly reduces the amount of data and filters out useless information, while preserving the important structural properties in an image.

The Canny edge detection algorithm [Canny 86] is known as the optimal edge detector.

Canny's algorithm receives a colour .jpg image file and it translates the colour image received to a grey-scale image, then it obtains the edges which are all the points belonging to the boundary of the objects in the image using a threshold.

First, the canny edge detector smoothes the image to eliminate noise. Then it finds the image gradient to highlight regions with high spatial derivatives. The algorithm then tracks along these regions and suppresses any pixel that is not at the maximum (method called non-maximum suppression). The gradient array is now further reduced by hysteresis which is used as a means of eliminating streaking. Streaking is the breaking up of an edge contour caused by the operator output fluctuating above and below the threshold. If a single threshold, T1 is applied to an image, and an edge has an average strength equal to T1, then due to noise, there will be instances where the edge dips below the threshold. Equally it will also extend the threshold making an edge look like a dashed line. To avoid this, hysteresis uses 2 thresholds, a high and a low. Any pixel in the image that has a value greater than T1 is presumed to be an edge pixel, and is marked as such immediately. Then, any pixels that are connected to this edge pixel and that have a value greater than T2 are also selected as edge pixels. If you think about following an edge, you need a gradient of T2 to start but you don't stop till you hit a gradient below T1.

But this algorithm obtains too many points and we need to create an additional filter for them. This filter consists of extracting the continuous line of points creating real boundaries and eliminating the ones that do not belong to this line by using the 8-neighbourhood pixel concept (figure 6.32). From the first pixel (point) resulting using the Canny algorithm we store only the points which are neighbours of the given pixel, until the same pixel is reached. Then the pixels which are not in the continuous line creating the boundary are eliminated and also all the pixels which do not define a closed polygon. If from the first point selected we do not reach itself again we start with the next point in the file to be analysed.

| | | |
|---|---|---|
| NW | N | NE |
| W | * | E |
| SW | S | SE |

***Figure 6.32.*** *The centre pixel marked with the symbol * and its 8 neighbours pixels.*

### 6.2.5.2 Qualitative Management of Boundary Points to Obtain Relevant Points.

Once we have obtained all the points of the boundaries of the objects in the image being analysed, the next step is to manage them in a qualitative way in order to obtain only the qualitative relevant points according to the qualitative shape matching theory explained in section 5. This means that we are going to determine:

- for each segment of the image if it is a straight segment or a curved segment,
- the vertices in the case of straight segments,
- the initial point, final point and point of maximum curvatura in the case of curves.

To determine the relevant points of a boundary we calculate the slope between two points $P_i$ and $P_j$, called $s_j$ and the slope between the first point $P_i$ and other consecutive point $P_k$, called $s_k$.

We have to notice that given a boundary of n points, the points chosen for the calculus of the slopes depend on the granularity (see section 1.1.4 for its definition) chosen. For instance, if the granularity is set to a value of 20, the first point ($P_i$) will be the point $P_0$ of the set of points of the boundary, the point $P_j$ will the point $P_{19}$, Pk will be the point $P_{39}$ and so on. The granularity is determined in function of the length of the edges of the object being analysed, the large the edge the bigger the value for their granularity. As we are managing tessereas, we have tested different granularities and we have set it at 20.

We obtain the slopes $s_j$ and $s_k$ by using next expression:

$s_j = \frac{(x_j - x_i)}{(y_j - y_i)}$; $s_k = \frac{(x_k - x_i)}{(y_k - y_i)}$, being $P_i=(x_i,y_i)$, $P_j=(x_j,y_j)$ and $P_k=(x_k,y_k$.

Once we obtain the slopes, we compare both and we can find the following situations:

1. If sj == sk then we have a straight line;
2. If sj != sk then we have a curved segment.

We repeat this process for a new point $P_l$, calculating the slope between $P_i$ and $P_l$ and comparing the resulting slope with the first slope sj.

Slopes comparison is not an exact equality, we left a margin because of the noise we find when obtaining the points. The margin set for our application is 0,5, which means that if the slopes differ in less than 0,5 then we consider both slopes as equal.

To determine the relevant points, the process followed consists of calculating slopes between the first point and the rest of points (chosen using the granularity), and comparing slopes we know if the points belong to a straight line or a curve, then:

1. If we are in a straight line, in which the slope is constant, the new relevant point will be the one that the slope is not constant anymore.
2. If we are in a curved segment, when we find a slope whose sign changes then we have found the next relevant point, which will be the point of maximum curvature.

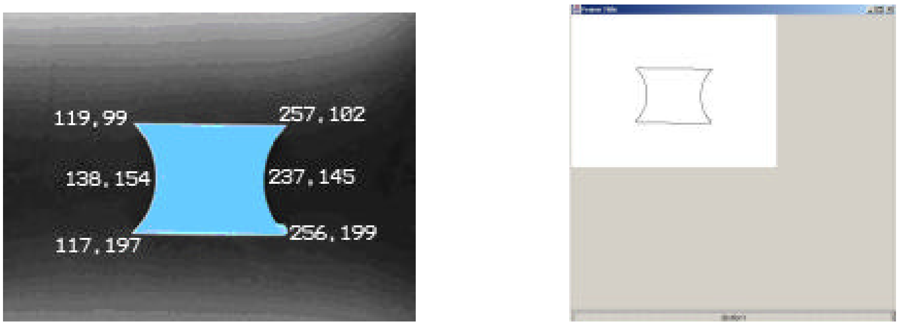
Then the process starts again with the new relevant point founded.

Although usually the first point of the set of points of the boundary will be a relevant point, we are not sure of that, therefore first of all we have to find the first relevant point. To detect the first relevant point we not only compare slopes from the initial point to the points after it, but we compare the slopes between the 10 points before the initial point of the set of points and the slopes between the 10 points after this initial point, refining the granularity to 5 only for this case. Then:

1. If the comparison between the slopes before the initial point of the set and after the initial point turns out to be equal (the slope is constant), the points before the initial point belong to the same straight segment as the initial point, therefore it is not the relevant first point and we follow the calculus of slopes until we find a point in which the slope calculated is not constant (the difference is bigger of 0,5), which will be the first relevant point.

2. If slopes are not constants with respect to the points before the first one, it could be that we are in a curved segment or it could be that it is the first relevant point of the file. Therefore we consider it as the first relevant point and we look for the second relevant point. If the second relevant point is determined because the sign of the slope changes we determine the segment as a curve, otherwise we determine the segment as a straight segment.

All the relevant points are stored in a vector with the next information: coordinates of each relevant point (determined as absolute coordinates respect to the image) followed by a string determining if the following segment is a straight segment or a curve segment. For instance the resulting vector of relevant points, v, for the figure 6.33 is v=[(257,102), curve, (237,145),curve, (256,199), straight-line, (117,197), curve, (138,154), curve, (119,99), straight-line], where the points are depicted over the image.



a) b)

***Figure 6.33.*** *a) Example of an image to extract its relevant points in which we have added the points obtained applying the method and b) graphical result obtained.*

At this point of the method, we have to check if we really have determined the type of segment correctly, because, as the points are chosen in function of granularity, sometimes we can determine that the relevant point belongs to a curve when it really belongs to a straight segment or vice versa. Therefore, if the final vector of relevant points has *m* elements, to check the type of segment we take the follwing steps:

1. The calculus of the straight line passing through two consecutive points in the vector, named $P_i = (x_i, y_i)$ and $P_{i+2} = (x_j, y_j)$, for i=0,…,m-2.
2. For the corresponding segment in the image we have to extract 3 points: the one at the middle and 2 more points close to $P_i$ and $P_{i+1}$ respectively. These points are extracted depending on the positions in the initial complete set of points of object $P_i$ and $P_{i+1}$. If the position (index of the point in the complete set of points of the boundary of the object) of $P_i$ is *t* and the position of $P_{i+1}$ is *g*, then the positions of the points that we are looking for are calculated as:

$$\mathtt{Central\_Pos} = \frac{(g - t)}{2} + t$$

$$\mathtt{ClosePi\_Pos} = \frac{(\mathtt{Central\_Pos} - t)}{4} + t$$

$$\mathtt{ClosePi} + 1\mathtt{\_Pos} = g - \frac{(g - \mathtt{Central\_Pos})}{4}$$

where *Central_Pos* means the position of the point in the middle of the real segment between $P_i$ and $P_{i+1}$ of the image (for this reason it is divided by 2), *Close_Pi_Pos* means the position of the point close (in order to be close we divide by 4) to $P_i$ in the real segment between $P_i$ and $P_{i+1}$ of the image and finally *ClosePi+1_Pos* means the position of the point close to $P_{i+1}$ in the real segment between $P_i$ and $P_{i+1}$ of the image.

Then, the points in the complete set of boundary points of the object obtained using Canny Algorithm at the positions calculated are extracted, named as $P_{cen}$, $P_{closei}$, $P_{closei+1}$.

3. With these three points we calculate 3 distances, the first one is the distance between $P_{cen}$ and the calculated straight line through $P_i$ and $P_{i+1}$ calculated in the step 1 (dc), the second one the distance between $P_{closei}$ and the same straight line (d1), and finally the third distance is the distance between $P_{closei+1}$ and the same straight line (figure 6.34). These distances are calculated with the formulas bellow, where for instance $\vec{p}_{i+1i}$ is the vector between $P_{i+1}$ and $P_i$, the symbol *x* means the vectorial product and the symbols | | indicates the absolute value of the result of the operation between them:

$$d_c = \frac{\left|\vec{p}_{i+1i} \times \vec{p}_{ceni}\right|}{\left|\vec{p}_{i+1i}\right|};$$

$$d_1 = \frac{\left|\vec{p}_{i+1i} \times \vec{p}_{closei\_i}\right|}{\left|\vec{p}_{i+1i}\right|};$$

$$d_2 = \frac{\left|\vec{p}_{i+1i} \times \vec{p}_{closei+1\_i}\right|}{\left|\vec{p}_{i+1i}\right|}$$

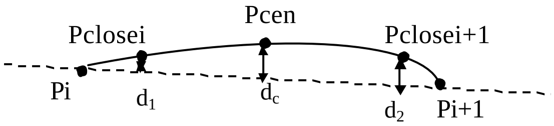


***Figure 6.34.*** *Example of the three distances calculated.*

4. Then we compare the three distances calculated, if $d_1$ or $d_2$ are bigger than $d_c$ then we are in a straight segment, otherwise, as we can find straight segments with steps, this means that the straight segment has not been perfectly obtained, so, we study the differences $dif_1=|d_1-d_c|$ and $dif_2=|d_2-d_c|$. If

dif$_1$ or dif$_2$ are less than a threshold (which in our application is set to 0,5), then we have a straight segment with steps, this means that we find aliasing effects. In all the other cases we have a curve segment.

If some type of segment changes then it is updated in the vector of relevant points and we obtain the final correct vector of relevant points together with the type of segment that follows each relevant point.

Finally, the colour of each object found in the image is stored as RGB coordinated. As the tessereas are coloured with one colour only it is calculated by getting the centroid of each object and storing the RGB colour.

Being an image of a number of pixels *high* * *width*, *k* the number of points (pixels) forming the boundaries of the figures in the image, and *d* the number of relevant points in the image, the computational cost of the hybrid method developed is of the order *O(high * width +k +d)*. Since, obviously *k* and *d* will be much smaller than *high * width*, the maximum cost is generated by this product which is the computational cost of the Canny Algorithm itself. Therefore we have not added a relevant computational cost to the most used quantitative method (Canny method), and at the end of the process we only have the relevant points of the figures in the image, together with information of the type of segments of each figure. This information will be the one needed to construct the qualitative string describing each object in the image as the qualitative shape matching theory describes, in the same way as described in section 6.2.1. Moreover, the boundaries obtained are more exact than the one obtained using only the Canny Algorithm, removing noise from the image. Figure 6.35a) shows in graphic form the boundaries obtained for the image in figure 6.28 using only the Canny method, and figure 6.35b) shows the result using the hybrid method here described for the same image.

The resulting vectors for each object in figure 6.35 are:

- For the figure at the north of the image, Figure1_v= [(164,87), curve, (249,113), curve, (317,81), curve, (351,167), curve, (322,229), curve, (241,200), curve, (167,233), curve, (131,152), curve];
- For the rectangle the east of the image, Figure2_v=[(487,94), straight-line, (497,315), straight-line, (418,315), straight-line, (410,94), straight-line];

- For the octagon of the image, Figure3_v=[(210,297), straight-line, (327,291), straight-line, (384,351), straight-line, (389,457), straight-line, (342, 510), straight-line, (220, 513), straight-line, (170,467), straight-line, (164,347), straight-line];
- For the rectangle at the south-west of the image, Figure4_v=[(95,457), straight-line, (235,632), straight-line, (172, 683), straight-line, (32,506), straight-line];
- And finally for the figure at the south-east of the image, like a leaf, Figure5_v=[(360,573), straight-line, (388,600), straight-line, (435,560), straight-line, (438,619), straight-line, (480,622), straight-line, (467,658), straight-line, (388,674), straight-line, (342,608), straight-line,].

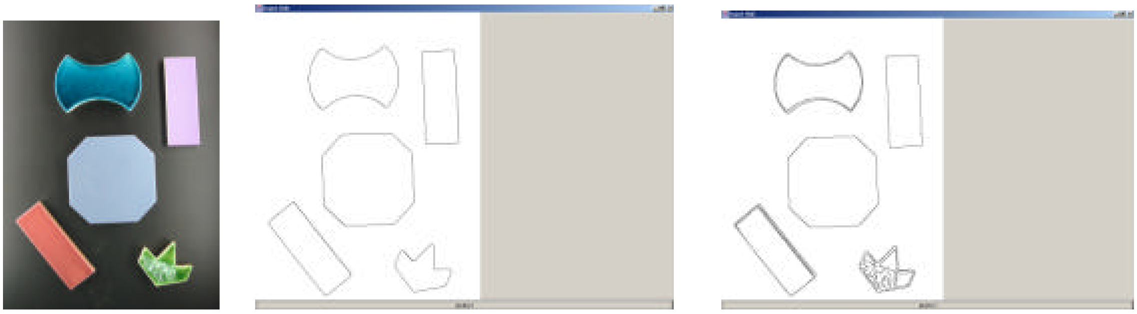

a) b) c)

***Figure 6.35.*** *Graphical representation of the boundaries obtained using only Canny Algorithm (a) and the hybrid method we have developed (b) for figure 6.30.*

Figure 6.36 shows an example when two pieces are so close, that they nearly share boundaries and we can see that the algorithm works correctly too. Figure 6.36 a) is the picture and figure 6.36b) shows the graphical result.

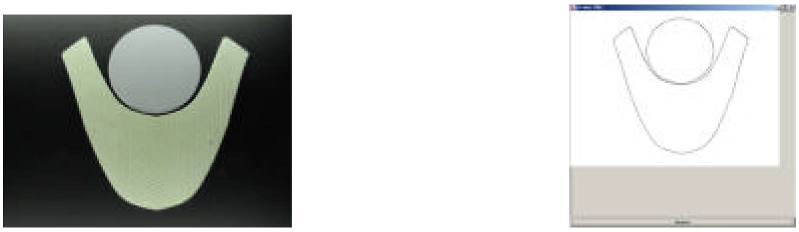

a) b)

***Figure 6.36.*** *Example of relevant points extraction with 2 close objects.*

We have to notice that the images are captured without any control of light conditions, distance between the camera and the objects captured and resolution. With these conditions we have captured up to 100 images and we have tested the extracting feature algorithm with them. The images can contain only one or several tessereas. The tests have been successful with 90 of the 100 images. Of the 10 images not successfully processed, from 8 of them a correct boundary was obtained but some relevant points are not well calculated, due to aliasing results, and then we obtain that the segment is a curve whilst it is a straight segment or vice versa. From the other 2 we cannot obtain the points of the boundary of part of the object or of one of the objects in the image, due to errors in the Canny method itself.

### 6.2.3 The matching process.

The matching process defined in section 5.5 is implemented in order to know if a tesserea belongs to a mosaic design.

The only relevant points for this implementation are described below.

In the case of objects with holes, section 5.5 establishes that the description of the object to match has to be made taking as first vertex the one which corresponds with the vertex taken as the first one when describing the reference object. Moreover, the description of the boundaries of the holes is also made in function of this vertex, because the first vertex considered for describing the boundary of the holes is the one closer to the first vertex of the description of the boundary of the container. The process to calculate the first vertex in the object to be matched with respect to the first vertex of the reference object (the one in the design) depends on the type of tesseraes that we are considering. If the tesserae is symmetric the process is more complex. Then we can differentiate between the following cases:

1. *No symmetric figures with holes.* As the figure is not symmetric, it is simple to find which vertex of the figure to be matched corresponds with the first vertex in the description of the reference figure. This vertex is calculated during the cyclic comparison of the boundaries of the figures.
2. *Symmetric figures with holes.* In this case, as there are several vertices that can be the first vertex of the figure to be matched that corresponds with the considered first vertex of the reference figure, the process is more complex. For instance, in a figure as a square the four vertices are equal; therefore we cannot determine a priori which vertex is the one taken as the first one in the reference figure. What we know is that in the case of the figures being equal

the orientation of their holes should be the same. Therefore, we choose two holes with cyclical identical boundary, one from the reference figure and the other from the figure to be matched. Then we calculate the orientation of the chosen hole in the object to match taking as first vertex all the candidate vertices to be the one considered the first one in the reference figure, this means that the NW of Frank's cardinal direction reference system will be oriented to each candidate vertex. Then we compare each obtained set of orientations with the set of orientation of the hole in the reference object. The one that is equal determines which vertex is the one that has to be considered as first vertex in the figure to be matched to carry on with the matching process. In the case that there is no set of orientations equal to the one in the reference figure, we look for another hole with the same cyclical boundary description as the one in the reference object, and we repeat the process with this new hole. If finally we do not find a cyclically identical hole to the one in the reference figure with a set of orientations equal, then we determine that the figures are not equal. Figure 6.37 shows in graphic form the calculus of the set of orientations for each candidate vertex (in the case of a rectangle there are two candidates):

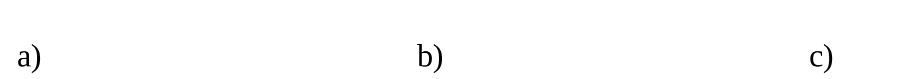


***Figure 6.37.*** *Example of finding the initial vertex of the object to be matched in symmetric figures; a) Reference figure b) Figure to match, orienting Frank Reference System to the first candidate vertex, and c) to the second candidate vertex. The correct one is the case b) in which the obtained set of orientations for the hole is the same as the set of figure a).*

Next we present the algorithm implemented for the matching process.

```
Function VerticesCyclicalComparison (SetVertices1,
SetVertices2){
     N=Calculus length SetVertices1
     M= Calculus length SetVertices2
     If N == M then {
     //Both sets have the same number of vertices
     For (i=0;i<N-1;i++) {
         For (j=0;j<N-1;j++){ //cyclic comparison of the
//containers
        //Compare Vertex1(0) from SetVertices1 with Vertex2(j)
     //from setVertices2
           If Vertice1(0) == Vertice2(j)
           {
          Num=0 //Init a counter
          For (k=1;k<N-1;k++)
                           {
            If (Vertice1(k)==Vertice2(j+i%N)) then
           {
                Num++;
                            }
           If (Num==N)
          {
             Return true
             Return j
             Break
                          }
              } //For K
           } // Si Vertice1(0) ==Vertice2(j)
        } //For j.
     } //For i
If (Num<>N)
{
        Return false
              }
      } //Si N==M
   Else
    {
       Return false;
    }
} //End function
```

***Algorithm 6.3. Part I. Description of the algorithm for the matching process.***

```
Function OrientationComparison(OrientS1, OrientS2)
{
     If (length.OrientS1 <> length.OrientS2)
     {
          Return false
     }
     Else
     {
          For(i=0; i++; i<lenght.OrientS1)
          {
               If (OrientS1[I] NO in OrientS2) then
               {
                    Return false
                    Break
               }
          }
          Return true
     }
}//End function

Function NonCyclicalComparison(holeS1, holeS2)
{
     If (length.holeS1<> length.holeS2) then
     {
          Return false
     }
     Else
     {
          equal = true
          i=0
          while (equal && i<length.holeS1)
          {
               If (holeS1[i]<>holeS2[i]
               {
                    equal = false
               }
               i++
          }
          Return equal
     }
}//End function
```

***Algorithm 6.3. Part II.*** *Description of the algorithm for the matching process.*

```
Function  Symmetry (Vector iv)
  {  //This  function  determines  if  the  vector  iv  describes  a
//symmetric figure
     length_iv = Calculus length iv;
     symmetry=false;
    If ((length_iv % 2) == 0) then
    {
      length_ss = length_iv / 2;
      a=0;
     while ((a< length_ss) && ! symmetry)
      {
         Vector  iv1, iv2;
          for( k=a; k<length_ss+a; k++)
          {
              iv1.add(iv[k]);
          }
          for ( h=long_ss+a; h<long_iv+a;h++)
          {
            If (h<length_iv) then
            {
              iv2.add(iv[h]);
            }
            Else
            {
              iv2.add(iv[h%long_iv]);
            }
          }
          If  (iv1 ==  iv2) then
            symmetry =true;
        a++;
      }
    }
   Else
    { //Number odd of vertices// odd vector length
      length_ss = length_iv / 2;
       a=0;
      while ((a<length_ss+1) && ! symmetry)
      {
         for(int k=a; k<length_ss+a; k++)
        {
            iv1.add(iv[k]);
        }
        for (int h=length_ss+a; h<length_iv+a-1;h++)
       {
            If(h<length_iv) then
            {
              iv2.add(iv[h]);
            }
            Else
            {
              iv2.add(iv[h%long_iv]);
            }
         }
```

***Algorithm 6.3. Part III.*** *Description of the algorithm for the matching process.*

```
        If (iv1 ==iv2) then
            symmetry =true;
       If (!symmetry) then
       {
           for (int h=length_ss+a+1; h<length_iv+a;h++)
          {
              If (h<length_iv) then
              {
                iv2.add(iv[h]);
              }
              Else
              {
                iv2.add(iv[h%length_iv]);
              }
          }
          If  (iv1 ==iv2) then
              symmetry =true;
      }
      a++;
     }
   }
   Return symmetry;
  }//End function
Algorithm FigureComparison (figure(S1), figure(S2))
If (Type_holes.S1 <> Type_holes.S2)  then
 {
      Return false
 }
Else { //Both same type_holes (or without-holes or with holes)
    If (Type_curve.S1 <> Type_curve.S2)  then
  {
      Return false
  }
  Else { //both same type_curve (or only_curves, with_curves or
without_curveS)
      Calculus Delta_E(C1,C2) //C1 and C2 colours of the figures
      If (Delta_E => 0,2) then
     {
     Return false
     }
   Else //Same colour
   {
        Extract containers characteristics (up to CIi).
        Value1=VerticesCyclicalComparison
(ContainerS1,ContainerS2)
        If (Value1==false) then
      {
     Return false;
      }
     Else {//Equal containers, if there are holes, compare them.
     If (Type_holes.S1==without-holes ) then
     {
           Return true
}
```

***Algorithm 6.3. Part IV.*** *Description of the algorithm for the matching process.*

```
Else
{//they have holes
    Calculus num_holes.S1 and num_holes.S2
    If num_holes.S1 <> num_holes.S2 then
     Return false
    Else
   {
    If (( num_holes.S1 == 1) && (!Symmetry (S1))) then
   {
     Value2=   OrientationComparison   (Orient_hole_S1[i],
Orient_hole_ S2[j])
     If (Value2==false) then
    {
     Return false
    }
    Else
   {
     Extract  qualitative  characteristics  hole_S1[i]  &
hole_S2[j]
     Valor2=      NonCyclicalComparison      (hole_S1[i],
hole_S2[j])
     If (Value==true) then
     {
         Return true
     }
     Else
     {
        Return false
     }
   }//Value1 = true
}//num_hole1 and not symmetry
Else
{ //One hole and symmetric figure
  If(( num_holes.S1 == 1) && (Symmetry(S1)))
  {
    boolean find_vertex =false;
    int i=0;
    while( i< Length_ContainerS1-1 && ! find_vertex)
   {
      vector_dir =ContainerS1[i];
      Value3=     OrientationComparison(Orient_hole_S1[i],
Orient_hole_ S2[j], vector_dir)
     If (Value3==false) then
     {
     i++;
     }
    Else
    {
     find_vertex = true;
   }
  }//while !find_vertex
}
```

**Algorithm 6.3. Part V.** *Description of the algorithm for the matching process.*

```
     If (Value3== false) then
         Return false
     Else
     {//found the vertex which returns the same orientation
          Corresponding_vertex = i;
          Extract qualitative characteristics hole_S1[i] &
     hole_S2[j]
          Value4= NonCyclicalComparison (hole_S1[i],
     hole_S2[j])
          If (Value4==true) then
          {
              Return true
          }
          Else
          {
             Return false
     }//equal orientation
       }//num_holes=1 and symmetry
    Else
    { //More than 1 hole and NOT symmetry.
     If ( (num_hole.S1 >1) && (!Symmetry(S1)) then
     {
       For (i=0; i++; i<num_holes)
        {
          counter = 0
          j = 0
          find = false
          while (!find &&  j<num_holes)
          {
              Value5= OrientationComparison(Orient_hole_S1[i],
Orient_hole_ S2[j])
              If (Value5==true) then
             {
               Extract qualitative description hole _S1[i] &
          hole_S2[j]
               Value6= NonCyclicalComparison (agujero_S1[i],
          agujero_S2[j])
               If (Value6==true) then
               {
                    find=true
                   counter++
               }
            }
             j++
          }
       }
      If (counter != num_holes) then
       {
          Return false
       }
      Else
      {
          Return true
       }
```

***Algorithm 6.3. Part VI.*** *Description of the algorithm for the matching process.*

```
}//num_holes >1 and not symmetry
Else
{ //num_holes >1 and Symmetry
    If ((num_holes.S1>1) && Symmetry(S1)) then
    {
     For (i=0; i++; i<num_holes)
     {
          counter = 0
          j = 0
         find = false
         while (!find &&  j<num_holes)
         {
              boolean find_vertex =false;
              int i=0;
             while(     i<     Length_ContainerS1-1     &&
!find_vertex)
               {
           vector_dir =ContainerS1[i];
           Value7=OrientationComparison(Orient_hole_S1[i],
     Orient_hole_ S2[j], vector_dir)
           If (Value7==false) then
           {
                 i++;
           }
           Else
           {
                find_vertex = true;
           }
         }//while !find_vertex
          If (Value7==false) then
         {
           Return false
         }
        Else
       { // Found the corresponding vertex
              Corresponding_vertex = i;
              Extract qualitative description hole_S1[i] &
     hole_S2[j]
              Value8=  NonCyclicalComparison  (hole_S1[i],
hole_S2[j])
              If (Value8==true) then
              {
            find=true
            counter++
                }
            }
            j++
        }
      }//end while
         }//end for
     }//end while
         }//end for
```

*Algorithm 6.3. Part VII. Description of the most important function for the matching process.*

```
              If (counter != num_holes) then
             {
           Return false
             }
             Else
            {
           Return true
            }
         }//num_holes >1 and  symmetry
            }
       }
}End Algorithm
```

*Algorithm 6.3. Part VIII.* Description of the most important function for the matching process.

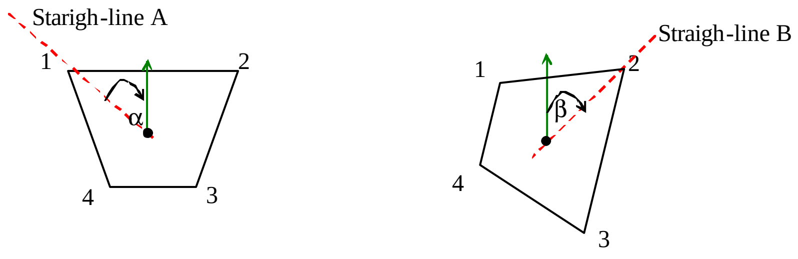
### 6.2.4 Calculus of the rotation angle.

The application not only returns which tesserae from an image belongs to which tesserae in a mosaic design but it also calculates the rotation angle that a robot arm has to make when picking up the tesserae by its centroid in order to place it in the correct orientation given by the design.

This rotation angle is calculated as follows (figure 6.38):

1. First we calculate the straight-line, called A, between the centroid and the first vertex of the reference figure (vertex number 1 in figure 6.38a)).
2. Secondly we calculate the straight-line, called B, between the centroid and the corresponding vertex (the one which is the first vertex in the reference figure) of the figure to match (vertex number 2 in figure 6.38b)).

The centroids of both figures in figure 6.38 are depicted with a black dot.



*a) Reference Figure*                    *b) Figure to match*

***Figure 6.38.*** *Straight-lines calculated for the calculus of the rotation angle.*

3. Then we calculate the director vector in the X axis direction (taking as centre of coordinates the centroids of the figures) in both figures (figure 6.38, depicted in green).

4. Then we calculate two angles: $\alpha$, which is the angle between straight-line A and the director vector in the reference figure, and angle $\beta$, which is the angle between the straight-line B and the director vector in the figure to match. Both angles are calculated using the following formulae:

$$\cos(ß) = \left|\cos(u, v)\right| = \left|\frac{u \times v}{|u| * |v|}\right|$$

, where *u* and *v* are director vectors and × means the scalar product.

This formulae returns the smaller angle between the 4 angles that appear when two straight-lines cross (figure 6.39). As one of the straight-lines is determined always by the director vector (1,0), to select the angle which we are looking for, we have to consider the position of the other straight-line crossing the director vector. If this straight-line is in the third or fourth quadrant of a coordinate system centred in the centroid of the figure, then the angle that we are looking for is not the angle that the above formula returns but it is its supplementary angle, that is 360º - angle returned by the formula.

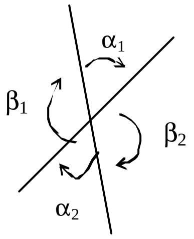


***Figure 6.39.*** *The 4 angles between 2 crossing straight-lines.*

5. Finally, we calculate the difference between both angles, and this (as a positive value) is the rotation angle to give to the figure to match in order to get the same orientation as the reference figure.

In the case that the figure contains holes, we do the same calculus described above but we do it not only with the container of the figure but also with the boundary of one of the holes of the figure. We select only one of the holes, because we calculate the angle when we already know that the figures are equal, then it is enough to consider only one hole which should be the same one in both figures. This is made in this way because rotating the figure with respect to one hole is sufficient since the other holes will rotate equally. Then we have two angles for each figure, one referent to the container and the other referent to the selected hole (figure 6.40), named $\alpha 1$ and $\alpha 2$ respectively. Then we compare the 4 angles as follows:

1. The angles calculated ($\alpha 1$ and $\beta 1$) from the containers of both figures are compared between them. The comparison consists of the difference between the angles as before.
2. The angles calculated ($\alpha 2$ and $\beta 2$) from the selected hole of both figures are also compared (calculus of the difference).
3. Therefore we have two possible angles of rotation, the one determined by the holes and the other determined by the containers. If the angle determined by the holes is bigger than the angle determined by the containers, the first one is the final angle of rotation. Otherwise, the angle will be the one determined by the containers.

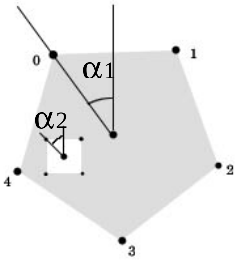
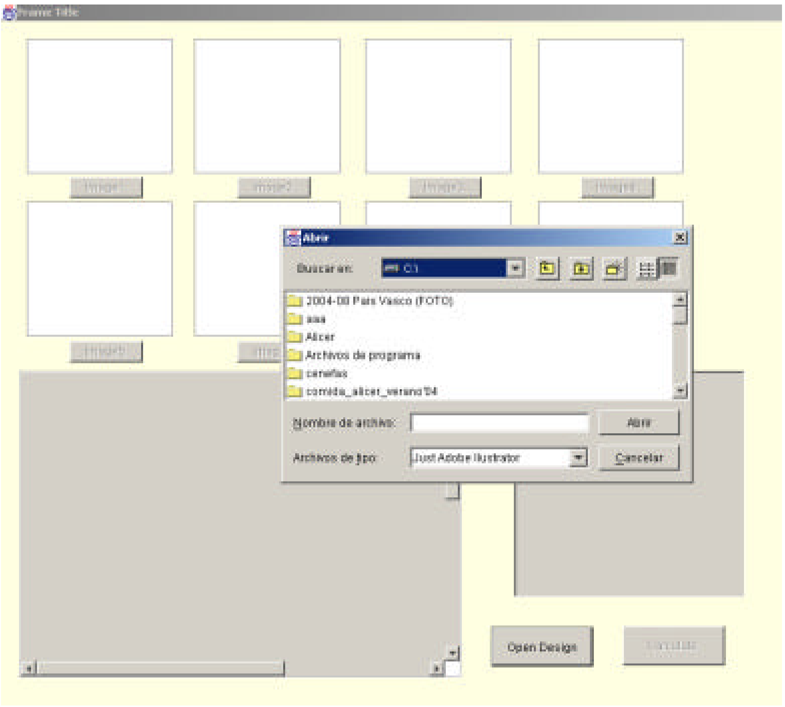


*Figure 6.40. Angles calculated in figure with holes.*

## 6.2.5 Tests

In this section we present the results of the application with mosaics design defined first with polygonal figures, secondly with holes and finally with curvilinear figures and holes.

First of all, when executing the application we should select the mosaic design (.AI file) that we want, clicking over the “Open Design” button (figure 6.41). When clicking the button a pop-up menu appears in which we look for the desired design. The user can only open files with Adobe Illustrator format.



Buscar en: C:\

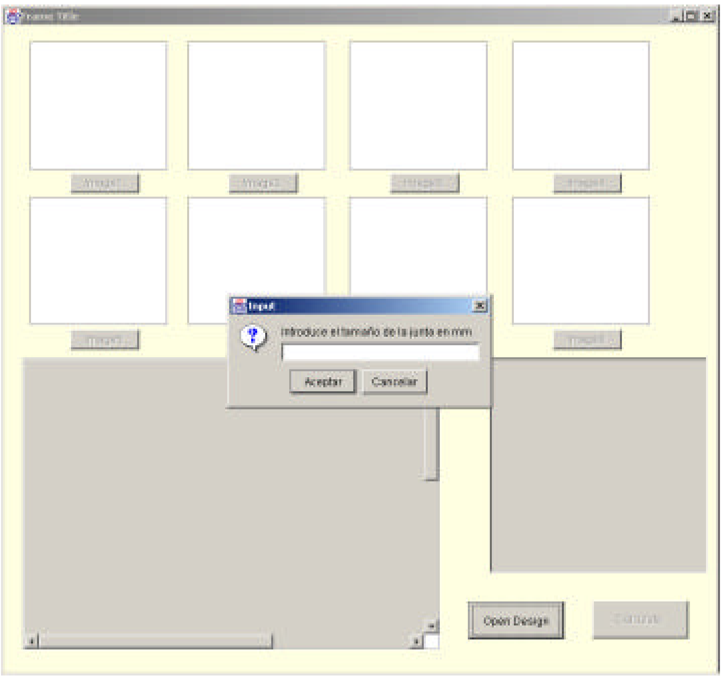
2004-08 Pais Vasco (FOTO)
aaa
Alicer
Archivos de programa
cenefas
comida_alicer_verano'04

Nombre de archivo:

Archivos de tipo: Just Adobe Ilustrator

*Figure 6.41.* *Picture of the application when selecting the mosaic design.*

Once the design is selected, the application asks the joint size (figure 6.42), because this data is used for comparing the size of the tesseraes in a qualitative way. This means that if two tesseraes differ more than the joint size they are not equal. The joint is the space left when placing the tesseraes side by side. There is default value which is established at 10 mm.



Introduce el tamaño de la junta en mm

*Figure 6.42.* *Pop-up menu asking for the joint size.*

The next section shows the results with different mosaic designs.

#### 6.2.5.1 Tests with Polygonal Tesseraes without Holes.

The individual tesseraes coming from the vision system are showed in the 8 small squares in the superior part of the application. As we can see in figure 6.43, all these tesseraes are depicted initially with its real colour but the tesseraes of the mosaic design are depicted using a lower intensity in order to indicate visually that the tesseraes in the mosaic are not yet placed. When one of the tesseraes coming from the vision system is matched against a tesserae of the mosaic design, the original image of the tesserae is depicted in a lower intensity and the corresponding one in the mosaic is depicted with its real colour. This shows visually that the tesserae in the mosaic has been matched against the one with lower intensity from the images (figure 6.44).

The design opened in figure 6.43 is a big mosaic with more than 300 tesseraes and the matching process is made very quickly. Most time in the application is consumed when drawing the figures again. But this time is not important because the real goal of the application is to interact with the robot arm, therefore it will be not necessary to redraw the figures. Now it is done in order to see if the application works correctly.

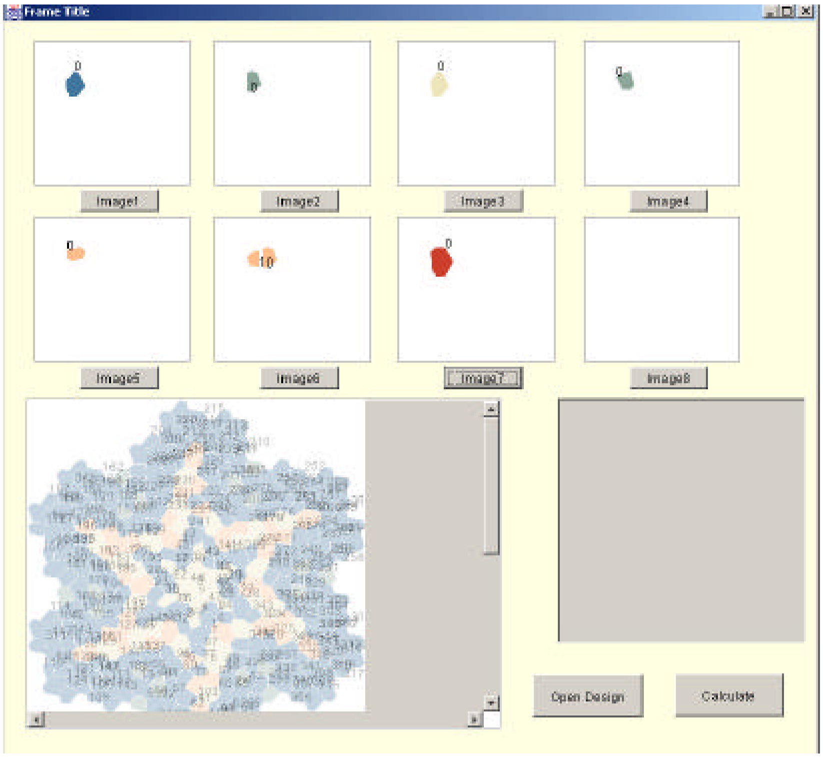
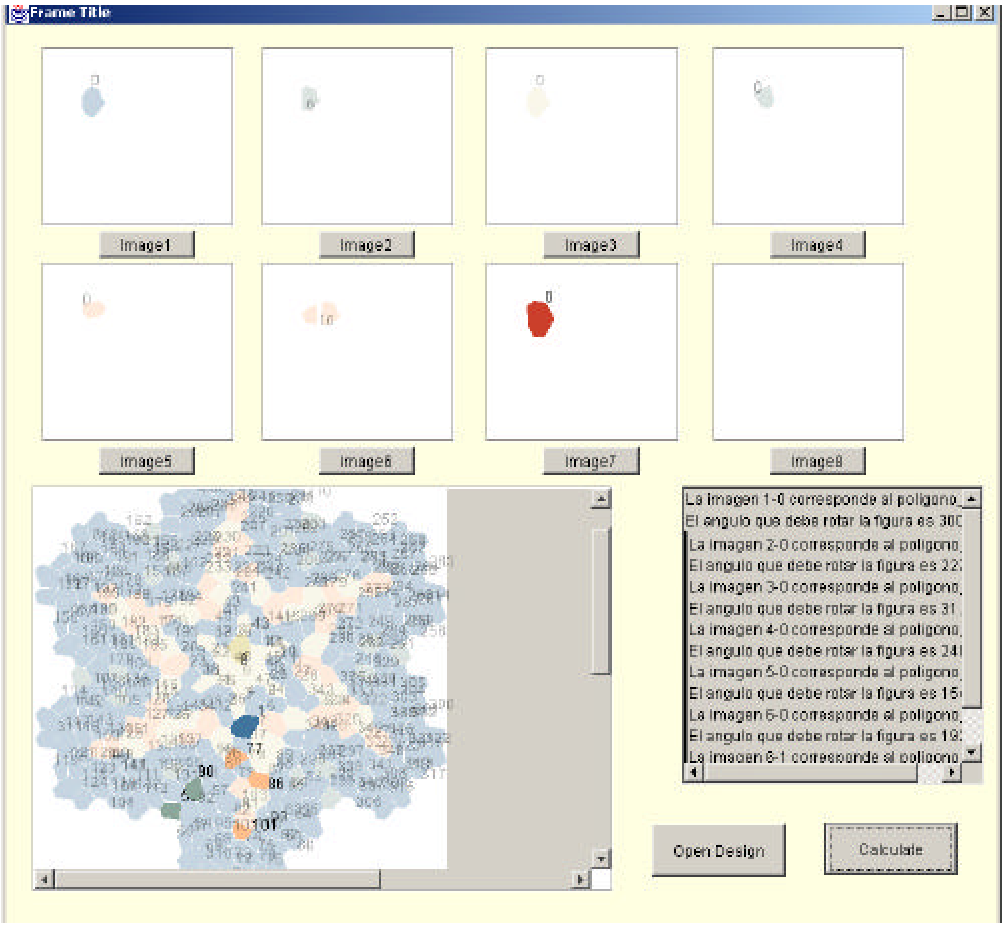


***Figure 6.43.*** *Mosaic with only polygons without holes.*

The matching process is launched clicking the button Calculate. When the matching process finishes, at the right part of the interface of the application shows which tesseraes of the images corresponds to tesseraes of the mosaic. In this section each tesserae of an image is identified by the number of the upper image to which it belongs and the number of polygon inside this image, and each tesserae of the mosaic design is identified by its polygon number. The angle of rotation for the tesserae in the image in order to be in the same orientation as the one in the mosaic is shown below the matching information of each pair of tesseraes.



```
La imagen 1-0 corresponde al poligono
El angulo que debe rotar la figura es 300
La imagen 2-0 corresponde al poligono
El angulo que debe rotar la figura es 22
La imagen 3-0 corresponde al poligono
El angulo que debe rotar la figura es 31
La imagen 4-0 corresponde al poligono
El angulo que debe rotar la figura es 24
La imagen 5-0 corresponde al poligono
El angulo que debe rotar la figura es 15
La imagen 6-0 corresponde al poligono
El angulo que debe rotar la figura es 19
La imagen 6-1 corresponde al poligono
```

*Figure 6.44.* *Result of the matching process.*

As we can see in figure 6.44, the Image 7, polygon 0 does not belong to the mosaic due to its colour.

#### 6.2.5.2 Tests with Polygonal Tesseraes with Holes.

Figure 6.45 shows an example of a mosaic design which contains two pieces with holes, one of them contains only one hole but the other one contains two holes.

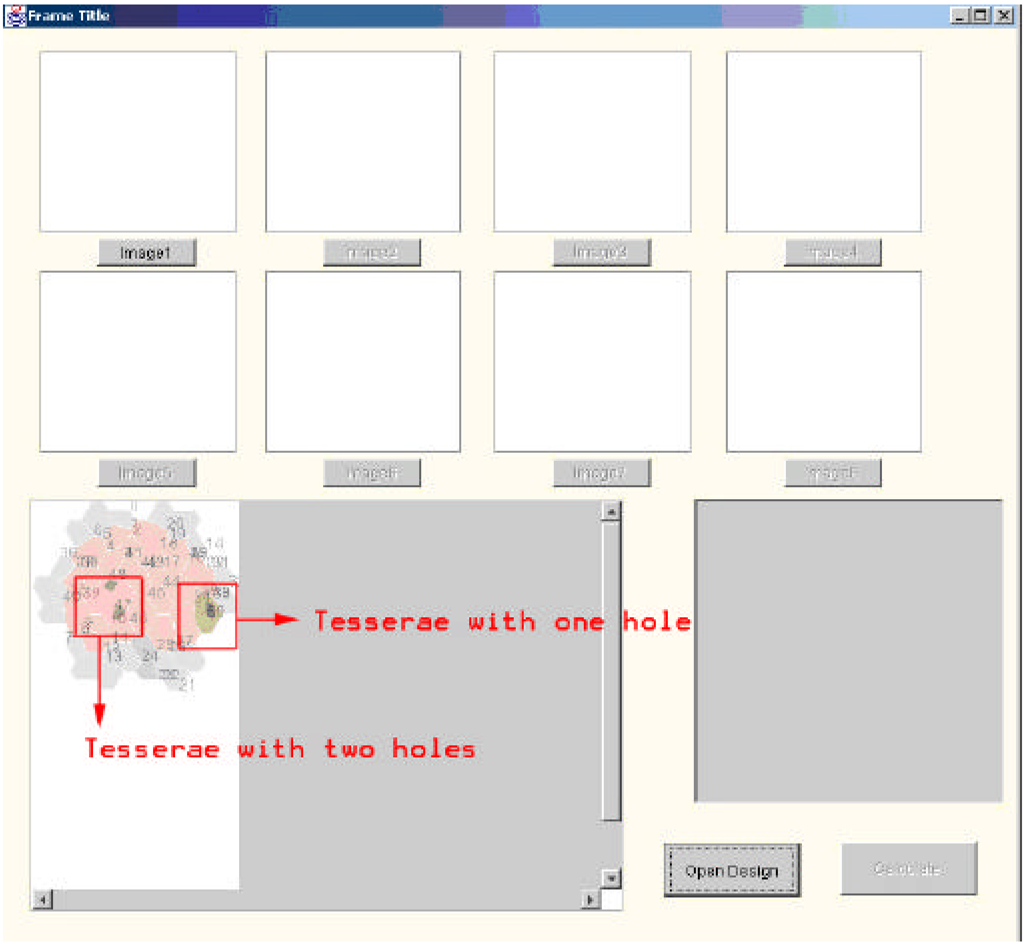
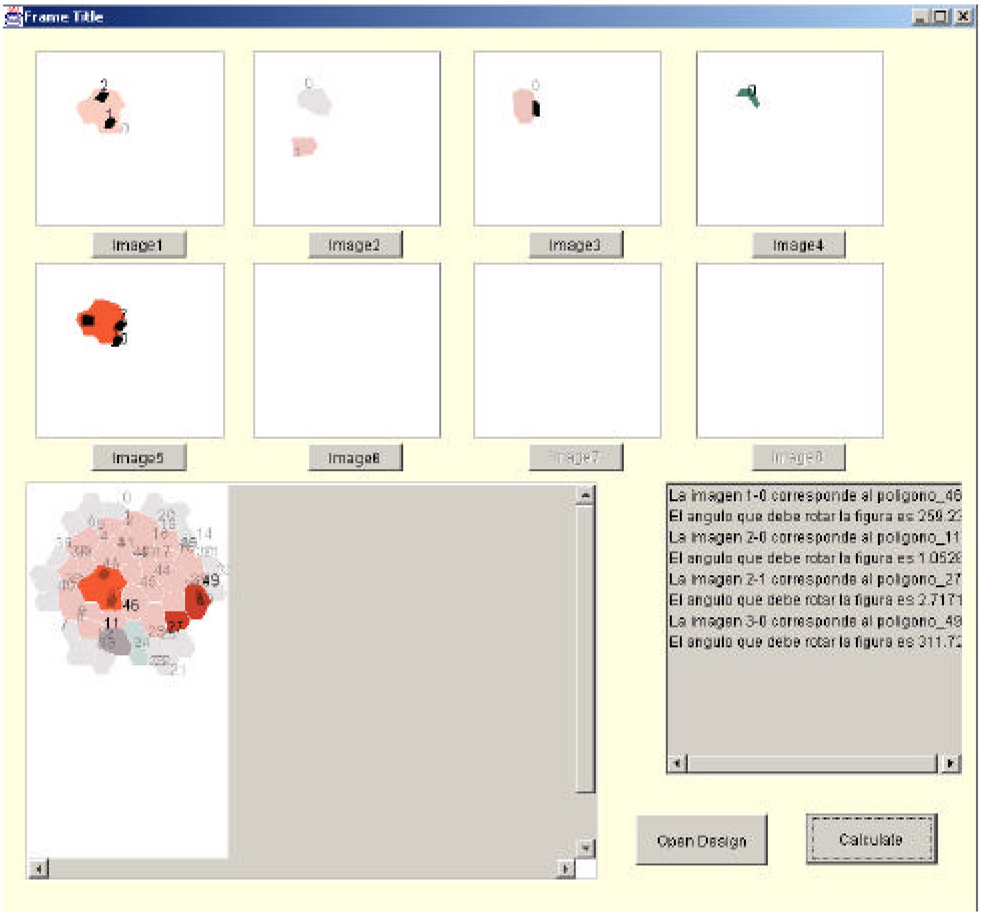


*__Figure 6.45.__ Example of a mosaic design with pieces containing holes.*

Figure 6.46 shows the result of the matching process for this example. The result indicates that the piece in the image 5 does not belong to the mosaic design because it has three holes. And the piece in the image 4 does not belong to the mosaic because there is no designed piece like the one in this image.

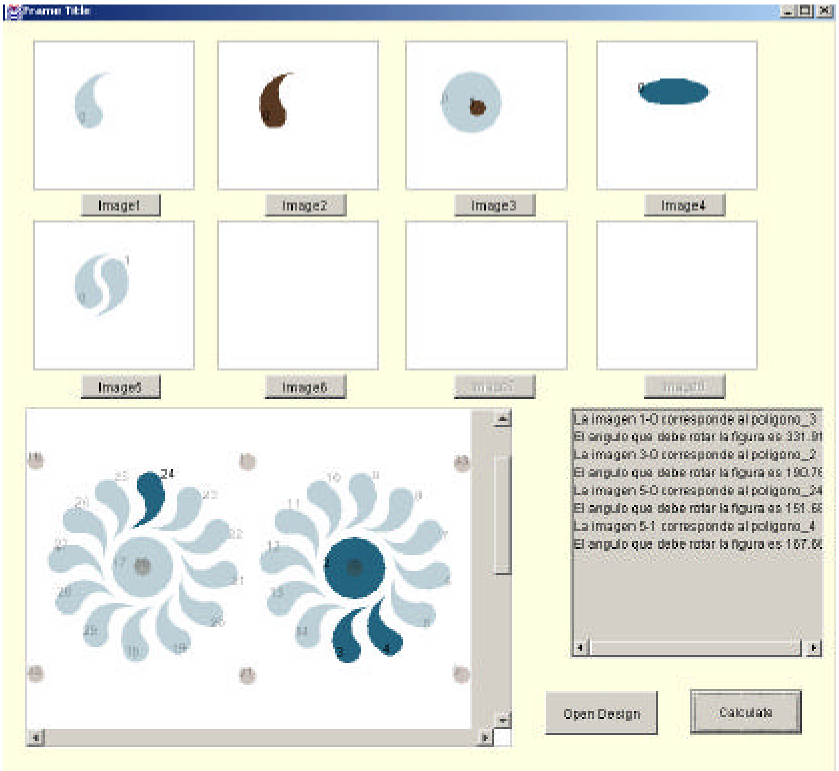


```
La imagen 1-0 corresponde al poligono_46
El angulo que debe rotar la figura es 259.22
La imagen 2-0 corresponde al poligono_11
El angulo que debe rotar la figura es 1.0528
La imagen 2-1 corresponde al poligono_27
El angulo que debe rotar la figura es 2.7171
La imagen 3-0 corresponde al poligono_49
El angulo que debe rotar la figura es 311.72
```

*__Figure 6.46.__ Result of the matching process in the case of pieces with holes.*

### 6.2.5.3 Tests with Curvilinear Tessereas with Holes.

In order to see if the theory has been completely and correctly implemented we have still to check if it works in the case of figures with curves or completely curved. Figure 6.47 is an example of this case, which shows also the result. The result indicates that the Image2, polygon 0 does not belong to the design due to its colour, and that the polygon in Image 4 does not belong to the design due to its shape. With this example we also show that the curvilinear figures with holes are also correctly matched.



```
La imagen 1-0 corresponde al poligono_3
El angulo que debe rotar la figura es 331.91
La imagen 3-0 corresponde al poligono_2
El angulo que debe rotar la figura es 190.76
La imagen 5-0 corresponde al poligono_24
El angulo que debe rotar la figura es 151.68
La imagen 5-1 corresponde al poligono_4
El angulo que debe rotar la figura es 167.06
```

***Figure 6.47.*** *Example of a design mosaic with curvilinear pieces with holes, some of them are symmetric too.*

This example shows how the theory is precise enough to distinguish between quite similar curved figures such as a circle and an ellipse.

Finally, it should be pointed out that the time for the matching process is negligible with respect to the time of drawing the design or the images in the application.

# CHAPTER 7

# CONCLUSIONS AND FUTURE WORK

## 7.1. CONCLUSIONS

To reason about space in a human-like way is he most important aim of QSR. In fact, from the end of 80's there has been a great interest in the study of qualitative models to represent and reason about features of space due to the fact that qualitative models are suitable to model the imprecision provided by human perception in the reasoning process. But, although different qualitative models have been developed for different aspects of space in recent years, there is still the need for having solutions for recognising objects and for describing and reasoning about the movement in situations with high imprecision and uncertainty. An example of these situations is the case of robotic applications where the robots have only sensorial information which is limited and imprecise. However, there are other applications where uncertainty is also present, for instance in order to know if two objects are equal or of the same quality within an industrial application.

Taking into account these open problems in QSR, the main contribution of this dissertation is that it develops and applies a model to reason about shape and about movement in a qualitative way. In depth, the contributions of this dissertation are the following:

- We have defined an approach for integrating topological aspects with other spatial aspects (orientation, cardinal directions and named distance) in the same model. To make the integration possible it is necessary to define:

  1. Its representation;
  2. The basic step of the inference process;
  3. The full inference process.

A uniformity of implementation of these three parts for each spatial aspect allows the integration. As shown in [Escrig and Toledo 98] this uniformity is achieved by using constraint logic programming extended with constraint handling rules (CLP + CHR) as a tool. The paradigm CLP+CHR is used to implement a constraint solver which solves in a straightforward way the complete inference process for each aspect of space to be integrated. Therefore CLP+CHR provide a suitable tool for the integration.

For the representation of topology, we have defined a constraint-based approach to the CBM calculus by [Clementini et al. 93]. What we have obtained is a calculus consisting of atomic relations and of the algebraic operations of converse and composition. As such, the calculus is an algebra in the same style as the one provided by Allen (1983) for temporal intervals. The objects manipulated by the calculus are point-like, linear and area features, contrary to most constraint-based frameworks in the qualitative spatial and temporal reasoning literature, which deal with only one type of feature (for instance, intervals in [Allen 83]). One problem raised by this was that the calculus had 27 composition tables, fortunately of moderate sizes. We have shown in this work that the use of 18 of these tables is sufficient, as from these 18 we can derive the other nine. Therefore, reasoning about knowledge expressed in this topological calculus can be done using a constraint propagation algorithm alike to the one in [Allen 83], guided by the 18 composition tables and the converse table. Such an algorithm has the advantage of being incremental: knowledge may be added without having to revise the processing steps achieved so far.

- The second main contribution of this dissertation is the definition of a new approach to reasoning about movement. In this PhD thesis, the movement of an object is seen as a shape whose topological relation with its environment (considered as other shape) changes through time.

  We have proposed an approach for movement as the integration of topological aspects together with time. The approach allows the integration of movement with other spatial aspects too (as orientation, distances, cardinal directions, topology, and so on) by following the three steps defined in [Escrig and Toledo 98], as we have done with the topological approach.

The movement theory, seen as a spatio-temporal representation model, has been implemented in a way that helps to reason about the sequence of topological situations that an autonomous robot should find during its way from a starting region to a target objective. It also helps to detect situations in which the robot is loosing its way. Then the robot could rectify its direction of movement to avoid getting lost.

On the other hand, the movement theory is also a constraint-based approach for modelling motion. What we have obtained is a calculus consisting of atomic topological relations changing with time and the algebraic operations of converse and composition. As such, the calculus is an algebra in the same style as the one provided by Allen [Allen 83] for temporal intervals. The objects manipulated by the calculus are point-like, linear and area entities, contrary to most of the constraint-based frameworks in the qualitative spatial and temporal reasoning literature, which deal only with one type of entities. Reasoning about knowledge expressed in the presented calculus can be achieved by using a constraint propagation algorithm like the one in [Allen 83], guided by the composition and converse tables presented in this dissertation. As before with topological information, such an algorithm has the advantage of being incremental: knowledge may be added without having to revise the processing step achieved so far.

- A third main contribution of this dissertation is the definition of a straightforward Qualitative Theory of Shape description. It will allow us to reason about shape in a qualitative way as human beings do. Moreover, most of the qualitative approaches developed nowadays are used for reasoning about object position, and the theory presented here allows us to use the same method to reason about position and shape. The theory proposed here is similar to the ordering information approaches but it is more cognitive in the sense that it does not need an external reference system to describe shapes, so it would not be suitable to apply it to the reasoning process of a robot. Nevertheless, the interest of ordering information for shape description relies in the fact that it is less constrained than metrical information but more constrained than topological information, which will not allow us to determine the convexity or concavity of the shape, or the length of edges, or the angle types.

The shape theory developed is a powerful theory which is able to recognise regular o non-regular polygons, shapes including curvilinear segments or being completely curvilinear, and all the types of shapes can contain one or more holes.

Moreover, the shape of an object, which is a spatial feature, has been also studied using topological concepts. The task of recognising objects during the movement of the robot is a very important task because for the development of very different tasks, the robot has to be able to recognise the objects that it finds during its movement. This is due to the fact that the objects can be landmarks which give spatial information to the robot about its environment.

- From the study of the next three spatial aspects: shape of the objects, topology and movement, we have shown that there are several works about these aspects, but most of them are theorical works and they have not been applied to robotics. Therefore, another main contribution of this dissertation is the development of theories for qualitative shape and movement descriptions applied to robotics in two ways:

    1. The qualitative movement and shape theories have been applied to a simulated robot navigation of a real robot, the Khepera2 robot. In this application the robot has to distinguish if two objects that it finds during its navigation are equal or not.

    2. The qualitative shape description and matching theory has been applied to an industrial application which is the automatic and intelligent recognition of tesseraes (tiles) against a mosaic design. This application is developed in order to allow the automatic and intelligent assembly of mosaics in the ceramic industry. A qualitative theory for this application presents several advantages against the use of a quantitative theory, such as the managing of uncertainty associated with the fact that no two tiles or tesseraes manufactured are identical.

- Finally, we have also developed a solution to extract from an image the relevant points of shapes needed for creating the qualitative description of shapes. This solution is called hybrid because, first we manage the data quantitatively and finally in a qualitative way by comparing the changes of slopes. This hybrid method has also been applied to the recognition of tiles creating a mosaic design in order to allow us an automatic and intelligent assembly of mosaic borders in the ceramic industry. We have also shown that this method works better than using only the Canny algorithm and it does not add any relevant computational cost.

## 7.2. FUTURE WORK

For future work we can identify several tasks that we summarize in this section.

- To develop a hierarchical study of different levels of granularity and integrate this treatment to the topological and movement concepts into our approach. The bigger the real physical space considered in the reasoning process is, the fewer details humans use to manage with it. For instance is we want to travel from Castellón to Paris by car, first of all we will look for the most important cities which will be close to the shortest path from the start to the goal city. Then we start to look for the best main roads which go from one place to another. Then, if we are already in Paris and we want to visit its interesting places nearby we will be interested in the city map of Paris, and so on. In the same way if we want to recognise landmarks during our visit to Paris in order to go to a specific place probably we will need to recognise buildings, but if we visit a museum the landmarks to be recognised will be probably rooms, columns or other architectural elements. Therefore, it is possible to distinguish between different levels of knowledge. The top level contains less information. It is an abstraction of the knowledge which appears in the lower levels. The lower levels will contain more details of a given part of the working space. Furthermore, no lower level can contain contradictory information with respect to the upper levels.
- The integration of the reasoning process with a learning process, more closely simulating human behaviour. The applications shown in this dissertation do not include any learning process. However, the use of learning techniques in both applications can improve the results.
- The extension of the autonomous robot navigation application to more complex environments and tasks and its transference to real robots and real environments and tasks.

- The application of multiagent technology in order to create a multiagent system to learn the structure of any structured environment which will accelerate the task of learning an environment.

- Extension of the Qualitative Shape Representation Theory by defining the operations needed to construct new shapes from a set of given shapes, that is, we should define operations such as union, intersection and difference of shapes. The idea is to define these operations in a topological way using the concepts of boundary, interior, exterior and dimension of the shapes. We can also develop a way to describe some relevant properties of the shapes which could help us during the matching task. These properties could be properties such as symmetry, alternation or iteration of some parts of the shape inside the shape itself.

- The software for the recognition of tiles against a mosaic design will be applied in the future to a robot arm which physically places the tiles in the correct position to create the final ceramic mosaic strip designed. Moreover, this application could be extended to the recognition and matching of objects of any other vectorial design, such as architectural sketches. This application has a high added industrial value, because its strategic goal is to reduce the costs associated with mosaics, which are products of high added value, that actually have associated high manufacturing costs because they are hand manufactured. Therefore, the application is a first step to getting a fully automatic and intelligent assembling process of a mosaic made of different tesseraes of different shapes and colours, a process which up to now is a quite slow and laborious. Thus, the application will reduce manufacturing costs and delivery time of the mosaics by using qualitative reasoning techniques.

- The application of the qualitative recognition theory to solve the problem of classification of objects, and then it could be used for the recognition and matching of objects partially occluded in the scene using the matching process for the visible part and the colour and texture of the objects.

- Improvement of the hybrid method to extract the relevant points from an image. It can be improved by:

    1. Establishing the best physical conditions (light, resolution, position of the camera, and so on) to capture the images which will improve the quality of the images captured.

2. As some problems in the image processing come because we find aliasing in the boundaries describing the shapes, and then we classify erroneously the type of edges, we can apply several techniques, as anti-aliasing techniques or filter techniques in order to solve this type of problem.
3. The thresholds used in the method have been established by testing several and fixing the one which obtains the best results. But as future work we could study the use of several methods, such as learning methods in order to adjust the thresholds better.

- Development of a more cognitive segmentation process for images, which would be more qualitative from its initial steps.

# REFERENCES

[Abbot et alt. 87] Abbott, K., Schutte, P., Palmer, M., and Ricks, W. “Fault.nder: a diagnostic expert system with graceful degradation for onboard aircraft application”. In *14th Int. Symp. Aircraft Integrated Monitoring Syst.*, 1987.

[Allen 83] Allen, J., “Maintaining Knowledge about Temporal Intervals”, *Communications of the ACM,* Vol. 26, No. 11, pg. 832-843, 1983

[Allen 84] Allen, J., “A General Model of Action and Time”. *Artificial Intelligence 23, 2*, July 1984.

[Allen 91] Allen, J.F. “Planning as Temporal Reasoning”. In *Proc. 2nd Principles of Knowledge Representation and Reasoning*, Morgan Kaufmann, 1991

[Alvarez-Bravo et alt. 04] J.V. Alvarez-Bravo, J.J. Alvarez-Sánchez, F.J. González-Cabrera, E. Valverde-Escorial, C. Alonso-González. "A conceptual approach to overcome learning difficulties in science using a qualitative ontology". IWPAAMS 04. Burgos (España). Octubre 2004.

[Anger, Guesgen and van Benthem 93] Anger, F., Guesgen, H.W., and van Benthem, J. editors. Proceedings of the Workshop on Spatial and Temporal Reasoning at the 13th International Joint Conference on Artificial Intelligence, 1993.

[Asher and Sablayrolles 95] N. Asher et P. Sablayrolles. A typology and discourse semantics for motion verbs and spatial PPs in French. *Journal of Semantics, 12(1)*:163–209, June 1995.

[Asher and Vieu 95] Asher N, Vieu L, "Towards a Geometry of Common Sense" in *Proceedings of IJCAI95*, 1995.

[Beaumont, Sattar et alt. 01] M. Beaumont, A. Sattar, M. Maher, and J. Thornton. “Solving over-constrained temporal reasoning problems”. In *Proceedings of the 14th Australian Joint Conference on Artificial Intelligence (AI 01)*, pp. 37–49, 2001.

[Bennett 94] B. Bennett. “Spatial reasoning with propositional logics”. In J. Doyle, E. Sandewall, and O. Torasso, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the 4$^{th}$ International Conference (KR94),* San Francisco, CA., Morgan Kaufmann.1994.

[Bennett, Cohn, Torrini and Hazarika 00a] Bennett, B; Cohn, A G; Torrini, P; Hazarika, S M. Describing rigid body motions in a qualitative theory of spatial region in: *Proceedings AAAI-2000 - the 17th National Conference on Artificial Intelligence,* pp. 503-509 AAAI Press. 2000.

[Bessa Machado and Bredeweg 03] V. Bessa Machado and B. Bredeweg, “Building Qualitative Models with HOMER: A Study in Usability and Support”. *Proceedings of the 17th International workshop on Qualitative Reasoning, QR'03*, P. Salles and B. Bredeweg (eds), pages 39-46, Brasilia, Brazil, 2003.

[Biederman 87] Biederman, I. Recognition-by-Components: A theory of Human Image Understanding. *Psychol. Rev., 94, 2*: 115-147, 1987.

[Bochman, 03] A. Bochman. “A logic for causal reasoning”. *In Proceedings IJCAI’03*, . Morgan Kaufmann, 2003.

[Borgo et al. 96] Borgo, S.; Guarino, N., and Masolo, C. A point-less theory of space base don strong congruente and connection. In Aiello, L. C., and Doyle, J., eds., *Principles of knowledge Representation and Reasoning, Proceedings of the 5th International Conference, KR96*. Morgan Kaufmann, 1996.

[Brady 83] Brady, M. Criteria for Representations of Shape. *Human and Machine Vision*, 1983.

[Brisson 89] Brisson, E. Representing Geometric Structures in d-dimensions: Topology and Order. *Proceedings 5th ACM Symposium on Computational Geometry, Saarbruchen,* 218-227, 1989.

[Canny 86] Canny, J. F. A computational approach to edge detection. *IEEE Transactions. on Pattern Analysis and Machina Intelligence*, 8, pp. 679-714, 1986.

[Chase 96] Chase S. C. "Design Modelling With Shape Algebras and Formal Logic" in *Design Computation: Collaboration, Reasoning, Pedagogy, proceedings of ACADIA '96*, Tucson, Arizona, October 31-November 3, 1996 Ed F Ozel, P McIntosh 99-113, 1996.

[Chase 97] Chase S. C. "Modeling spatial reasoning systems with shape algebras and formal logic", *AI EDAM: Artificial Intelligence for Engineering Design, Analysis and Manufacturing 11*:4 273-285, 1997.

[Chen and Freeman 90] S. Chen and H. Freeman, "Computing characteristic views of quadric-surfaced solids," *Proc.10th Int'l. Conf. on Pattern Recognition*, pp. 77-82, June 1990.

[Clarke 81] B. L. Clarke. “A calculus of individuals based on “connection””. *Notre Dame Journal of Formal Logic,* 23(3), pp. 204-218, 1981.

[Clarke 85] Clarke, B. L., "Individuals and points," *Notre Dame Journal of Formal Logic*, *vol. 26*, pp. 61—67, 1985.

[Claramunt and Thériault 96] Claramunt, C. and M. Thériault. Toward Semantics For Modeling Spatio-Temporal Processes Within GIS. *Advances in GIS II(SDH96).* M. J. Kraak and M. Molenaar. Delft, the Netherlands, Taylor and Francis: 47-64, 1996.

[Claramunt et al. 97] Claramunt, C., Parent, C., and Theriault, M., Design Patterns for Spatiotemporal Processes. *IFIP, 1997* (Chapman and Hall), 1997.

[Clement and Durfee 99] Clement, B., and Durfee, E. “Theory for Coordinating Concurrent Hierarchical Planning Agents using Summary Information”. *Proceedings of the 16thNational Conference on AI*, 1999.

[Clementini and Di Felice 94] Clementini, E., & Di Felice, P. Topology in an O-O-GIS. *In Proceedings SEBD 1994*, pp. 101-119, 1994

[Clementini and Di Felice 95] Clementini, E., & Di Felice, P. A comparison of methods for representing topological relationships. *Information Sciences*, *3,* 149-178, 1995.

[Clementini and Di Felice 97] Clementini, E. and Di Felice, P. A global framework for qualitative shape description. *GeoInformatica 1(1):*1-17, 1997.

[Clementini et al. 93] Clementini, E., Di Felice, P., & van Oosterom, P. A small set of formal topological relationships suitable for end-user interaction. *In D. Abel, & B. C. Ooi (Eds.), Advances in spatial databases - Third International Symposium, SSD'93*, Singapore (pp. 277-295). Berlin: Springer, 1993.

[Clementini et al. 97] Eliseo Clementini, Paolino Di Felice, Daniel Hernández. Qualitative Representation of Positional Information. . *Artif. Intell. 95(2):* 317-356, 1997.

[Cohn 94] Cohn, A G. "Reasoning about space". In: *1994 AISB Workshop on Automated Reasoning: Bridging the Gap between Theory and Practice*, pp. 9-10 University of Leeds/AISB. 1994.

[Cohn 95] Cohn, A.G. A Hierarchical Representation of Qualitative Shape based on Connection and Convexity. *Proceedings COSIT'95*, Springer-Verlag, 311-326, 1995.

[Cohn et al. 93] Cohn, A. G., Randell, D. A., Cu, Z., and Bennett, B. Qualitative spatial reasoning and representation. *In Proceedings of the III IMACS International Workshop on Qualitative Reasoning and Decision Technologies – QUARTET'93-* , Piera Carreté, N. and Singh, M.G. editors, pages 513-522, 1993.

[Cohn et al. 97] Cohn, A., Bennett, B., Gooday, J. and Gotts, N. "Qualitative Spatial Representation and Reasoning with the Region Connection Calculus", *Geoinformatica, 1*, 1-44, 1997.

[Cohn and Hazarika 01] Cohn, A G; Hazarika, S M. "Qualitative spatial representation and reasoning: an overview". *Fundamenta Informaticae, vol. 45*, pp. 1-29. 2001.

[Cohn and Hazarika 01b] Cohn, A.G. and Hazarika, S.M. "Spatio-Temporal continuity in Geographic Space*", Meeting on Fundamental Questions in Geographical Information Science*, Manchester, England, 2001

[Cui, Cohn and Randell 92] Z. Cui, A. G. Cohn, and D.A. Randell. "Qualitative simulation based on a logical formalism of space and time". In *Proceedings AAAI-92*, pp. 679-684, Menlo Park, California, AAAI Press, 1992.

[Cui et al. 93] Cui, Z; Cohn, A G; Randell, D A. Qualititative and topological relationships in spatial databases in: *Abel, D & Ooi, B C (editors) Advances in Spatial Databases*, pp. 296-315 Springer-Verlag. 1993.

[Davis 88] E. Davis. A logical framework for commonsense predictions of solid object behaviour. *Artificial intelligence in engineering, 3(3):*125–140, 1988.

[Damski and Gero 96] Damski, J. C. and Gero, J. S. A logic-based framework for shape representation. *Computer-Aided Design 28(3):*169-181, 1996.

[de Kleer 77] de Kleer, J., "Multiple Representations of Knowledge in a Mechanics Problem Solver", *Proceedings of IJCAI-77*, pp. 299-304, 1977.

[Del Pobil and Serna 95] Del Pobil, A.P., Serna, M.A. Spatial Representation and Motion Planning. *Lecture Notes in Computer Science no. 1014*, Springer-Verlag, 1995.

[Dressler and Struss 03] Dressler, O., Struss, P. "A Toolbox Integrating Model-based Diagnosability Analysis and Automated Generation of Diagnostics". In *Proceedings of the 14th International Workshop on Principles of Diagnosis* (DX03), pp. 99-104, Washington, D.C., USA, June 2003.

[Dugat, Gambarotto and Larvor 02] Vincent Dugat, Pierre Gambarotto, Yannick Larvor. Qualitative Geometry for Shape Recognition. In: Applied Intelligence, Kluwer Academic Publishers, V. 17 N. 3, p. 253-263, 2002.

[Egenhofer 91] Egenhofer, M. J. Reasoning about binary topological relations. *In Gunther, O. and Schek, H. J., editors, Advances in Spatial Databases, Second Symposium on Large Spatial Databases, Volume 525 of Lecture Notes in Computer Science*, pages 143-160, Springer, Berlin, 1991.

[Egenhofer, Clementini and Di Felice 94] M. Egenhofer, E. Clementini, and P. di Felice, Topological Relations between Regions with Holes, *International Journal of Geographical Information Systems 8 (2)*: 129-144, 1994.

[Egenhofer and Franzosa 91] Egenhofer, M.J. and Franzosa, R. "Point-Set Topological Spatial Relations." *International Journal of Geographical Information Systems 5 (2)*, pp. 161-174, 1991.

[Egenhofer and Franzosa 95] Egenhofer, M.J. and Franzosa, R. On the Equivalence of Topological Relations *International Journal of Geographical Information Systems 9 (2)*: 133-152, 1995.

[Egenhofer and Herring 95] Max J. Egenhofer, John R. Herring: *Advances in Spatial Databases, 4th International Symposium*, SSD'95, Portland, Maine, USA, August 6-9, 1995, Proceedings Springer 1995

[Escrig and Toledo 98] Escrig, M.T., Toledo, F., *Qualitative Spatial Reasoning: theory and practice. Application to Robot Navigation*. IOS Press, Frontiers in Artificial Intelligence and Applications, ISBN 90 5199 4125. 1998.

[Escrig & Toledo 00] Escrig, M.T and Toledo, F. Autonomous Robot Navigation using human spatial concepts. *International Journal of Intelligent Systems*, Vol.15, No.3, pp.165-196, 2000, March 2000.

[Escrig and Toledo 01] M.T. Escrig, F. Toledo, "Applying Compared Distances to Robot Navigation". In *the International Joint Conference on Artificial Intelligence, Workshop on Spatial and Temporal Reasoning with "agents" focus*. August 2001.

[Escrig & Toledo 02] Escrig, M. T. And Toledo, F. "Qualitative Velocity"; *Lecture Notes in Artificial Intelligence num. 2504 Topics in Artificial Intelligence*, 2002.

[Faltings 90] B. Faltings. Qualitative kinematics in mechanisms. *Artificial Intelligence, 44(1-2):*89–119, 1990.

[Ferguson et al. 00] Ronald W. Ferguson, Robert A. Rasch Jr., William Turmel, Kenneth D. Forbus, "Qualitative Spatial Interpretation of Course-of-Action Diagrams". *AAAI/IAAI 2000*, pp. 1119-1120.

[Ferruci and Paoluzzi 91] Ferrucci, V. and Paoluzzi, A. Extrusion and Boundary Evaluation for Multidimensional Polyhedra. *Computer Aided Design, 23, 1*: 40-50, 1991.

[Flynn and Jain 91] Flynn , P.J. and Jain, A.K. CAD-based Computer Vision: From CAD Models to Relational Graphs. *IEEE Transaction on P.A.M.I.,* 13:114-132, 1991.

[Forbus 83] Forbus ,Kenneth D.Modeling Motion With Qualitative Process Theory. *AAAI 1982*: 205-208

[Forbus 84] Forbus, K., "Qualitative process theory". *Artificial Intelligence 24*, pp. 85–168, 1984

[Forbus 96] K.D. Forbus. "Qualitative Reasoning". In A.B. Tucker, editor, The Computer Science and Engineering Handbook, pp. 715--733. CRC Press, 1996.

[Forbus, Ferguson and Usher 01] Kenneth D. Forbus, Ronald W. Ferguson, Jeffery M. Usher., "Towards a Computational Model of Sketching", *Intelligent User Interfaces,* 2001.

[Forrest 95] P. Forrest. Is space-time discrete or continuous?- an empirical question. *Synthese, 103*:327-35, 1995.

[Frank 92] A. Frank. Qualitative spatial reasoning with cardinal directions. *Journal of Visual Languages and Computing,* 3, pp. 343-371, 1992

[Frank et al. 93] Frank, A. U. and Campari, I. Editors. *Spatial Information Theory A Theoretical Basis for GIS. European Conference,COSIT'93,* Volume 639 of Lecture Notes in Computer Science, Springer,Berlin, 1993

[Freeman and Chakravarty 80] Freeman, H. And Chakravarty, I. The use of characteristic views in the recognition of three-dimensional objects. *Pattern Recognition in Practice*, 1980.

[Freksa 91] Freksa, C. "Conceptual Neighborhood and its role in temporal and spatial reasoning". In Singh, M., Travé-Massuyés, L. (eds.), Proc. Of the IMACS Worwshop on Decision Support Systems and Qualitative Reasoning, pp. 181-187, Noth-Holland, Amsterdam 1991.

[Freksa 92] Freksa, C. "Temporal reasoning based on semi-intervals". In *Artificial Intelligence, vol. 54,* pp. 199-227, 1992.

[Freksa 92b] Freksa, C., "Using Orientation Information for Qualitative Reasoning", in A. U. Frank, I. Campari, and U. Formentini (editors). Theories and Methods of Spatio-Temporal Reasoning in Geographic Space. *Proceedings of the International Conference on GIS- From Space to Territory, Pisa, volume 639 of Lecture Notes in Computer Science*, pp. 162-178, Springer, Berlin, 1992.

[Freksa & Zimmermann 93] Freksa, C. and Zimmermann, K. On the Utilization of Spatial Structures for Cognitively Plausible and Efficient Reasoning. *In Proc. Of the IEEE International Conference on Systems, Man and Cybernetics,* pp. 18-21. 1992

[Freksa and Zimmermann 96] Freksa, C. and Zimmermann, K., "Qualitative Spatial Reasoning Using Orientation, Distance, and Path Knowledge", in *Applied Intelligence, Vol. 6*, pp. 49-58, 1996.

[Fruehwirth 94] Fruehwirth, T., "Constraint Handling Rules", in A. Podelski (ed.), Constraint Programming: Basic and Trends, volume 910 of Lecture Notes in Computer Science, Springer, pp. 90-107, 1994.

[Galton 93] A. Galton. Towards an integrated logics of space, time and motion. IJCAI, 1993.

[Galton 97] A. Galton. Space, time and movement. *In O. Stock, editor, Spatial and Temporal Reasoning.* Kluwer, 1997.

[Galton and Meathrel 99] Antony Galton and Richard Meathrel. Qualitative Outline Theory, in Thomas Dean (ed.), *Proceedings of the 16th International Joint Conference on Artificial Intelligence (*IJCAI'99*)* , pages 1061-66, Stockholm, Sweden, 31 July-6 August 1999

[Gotts 96] Gotts, N. M. Topology from a single primitive relation: defining topological properties and relations in terms of connection (Technical Report No. 96_23). University of Leeds: School of Computer Studies, 1996.

[Gotts et al. 96] N M Gotts, J. M. Gooday and A. G. Cohn. "A connection based approach to commonsense topological description and reasoning". *The Monist, 79(1),* pp. 51-75, 1996.

[Guesgen 89] Guesgen, H.W. "Spatial reasoning based on Allen's temporal logic", Technical Report TR-89-049, International Computer Science Institute, Berkeley, 1989.

[Hayes 85] P.J. Hayes. An ontology for liquids. *In J.R. Hobbs et R.C. Moore, editor, Formal Theories of the Commonsense World. Ablex Publishing Corporation*, Norwood, 1985.

[Hernández 94] D. Hernández. *Qualitative Representation of Spatial Knowledege,* volume 804 of *Lecture Notes in Artificial Intelligence.* Springer-Verlag,1994.

[Hernández, Clementini and Di Felice 95] D. Hernández, E. Clementini, and P. Di Felice. Qualitative distances. In *Proceedings of COSIT'95, LNCS 988*, Semmering, Austria, Springer, 1995.

[Hoffman and Richards 85] D. D. Hoffman and W. A. Richards. Parts of Recognition, *Visual Cognition*, S. Pinker Editor, MIT Press, pp: 65-96, 1985.

[Hornsby and Egenhofer 02] Kathleen Hornsby and Max Egenhofer. Modeling Moving Objects over Multiple Granularities, *Annals of Mathematics and Artificial Intelligence 36 (1-2)*:177-194, 2002.

[Isli, Museros et al. 00] Isli A., Museros L., Barkosky T., Reinhard M., "A Topological Calculus for Cartographic Entities. Conceptualizang New Regions", *Lecture Notes in Artificial Intelligence 1849. Spatial Cognition II. Integrating Abstract Theories, Empirical Studies, Formal Methods and Practical Applications.* Springer-Verlag , ISBN : 3-540-67584-1, pgs. 225-238, 2000

[Jong 94] Jong, J.H., "Qualitative Reasoning About Distances and Directions in Geographic Space". PhD thesis. Department of Surveying Engineering. University of Maine, 1994.

[Jungert 92] Jungert, E., "The Observer's Point of View: An Extension of Symbolic Projections", in *International Conference GIS -From Space to Territory: Theories and Methods of Spatio-Temporal Reasoning in Geographic Space. Volume 639 of Lectures Notes in Computer Science.* Ed. Springer-Verlag, 1992.

[Jungert 94] Jungert, E. Symbolic spatial reasoning on object shapes for qualitative matching. *Spatial Information Theory*. Springer Verlag, Berlin, 444-462, 1994.

[Kuipers 84] B. J. Kuipers. "Commonsense reasoning about causality: deriving behavior from structure". *Artificial Intelligence 24,* pp. 169-203, 1984.

[Kumar 92] Kumar, V., "Algorithms for constraint satisfaction problems: A survey". AI Magazine, 13(1), pp. 32.44, 1992.

[Leyton 88] Leyton, M. A process-grammar for shape. *Artificial Intelligence, 34*:213-247, 1988.

[Ligozat 93] Ligozat, G. F. Models for qualitative spatial reasoning *In Anger et al.* pages 35-45, 1993.

[Mackworth and Freuder 85] Mackworth, A.K., and Freuder, E.C. "The complexity of some Polynomial networks consistency algorithms for constraint satisfaction problems", Journal of Artificial Intelligence 25, pp. 65-74, 1985.

[Matuszec 88] D. Matuszec, T. Finin, R. Fritzson, and C. Overton. "Endpoint relations on temporal intervals" Technical Report PRC-LBS-8810, Paoli Research Center, Unisys Corp. 1988.

[McDermott 82] D. McDermott, "A Temporal Logic for Reasoning about Processes and Plans", *Cognitive Science,* vol. 6 (1982), 101-155.

[Meathrel and Galton 00] Richard Meathrel and Antony Galton. Qualitative Representation of Planar Outlines, In *Werner Horn (ed.), Proceedings of the 14th European Conference on Artificial Intelligence (ECAI 2000),* pp.224-8, Berlin, Germany, August 20-25, 2000. IOS Press, Amsterdam, 2000

[Meathrel and Galton 01] Richard Meathrel and Antony Galton. A Hierarchy of Boundary-based Shape Descriptors. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence* (IJCAI'O1),, pages 1359-1364, Seattle, Washington, U.S.A, August 4-10, 2001.

[Meiri 91] Meiri, I., "Combining qualitative and quantitative constraints in temporal reasoning", in *Proc. of the 7th National Conference on Artificial Intelligence*, pp. 260-267, 1991.

[Mukerjee and Joe 90] Mukerjee, A., Joe, G., "A Qualitative Model for Space". In *8th-AAAI*, pp. 721.727, 1990.

[Mukerjee and Mittal 95] Mukerjee, A., Mittal, N. "A qualitative representation of frame-transformation motions in 3-dimensional space", in Proceedings ICRA, 1995.

[Muller 98a] Muller, P. "A qualitative theory of motion based on spatio-temporal primitives", *in Proceedings of KR-98*, ed., A. G. Cohn et al., pp. 131–141. Morgan Kaufman, 1998.

[Muller 98b] Muller, P. "Space-time as a primitive for space and motion", in *Formal ontology in Information Systems*, ed., N. Guarino, pp. 63–76. IOS Press, 1998.

[Museros 98] Museros, L. "Conceptualizing Regions on the basis of Cartographic Entities", *Wotkshop Working Notes of the WokShop on Spatial Inference. Designing the interface between mental and computational approaches to spatial problem solving*, 1998

[Museros and Escrig 01a] Museros L., Escrig M.T., "Enhancing Topological Information with other Qualitative Spatial Aspects ", *Diagnosis, Razonamiento Cualitativo y Sistemas Socioeconómicos. Carlos Alonso y Juan Antonio Ortega Eds.* ISBN : 84- 95499- 35- 5, pgs. 113-123, Julio 2001.

[Museros and Escrig 01b] Museros L., Escrig M.T. "The integration of topological information, qualitative orientation and positional information using constraint logic programming for robot navigation", *Proccedings of the IJCAI-2001 Workshop on Spatial and Temporal Reasoning with « Agents » Focus . Rita Rodriguez (Eds.),* pgs. 21-27, Agosto 2001.

[Museros and Escrig 01c] Museros L., Escrig M.T., " Qualitative Spatial Reasoning using orientation distances and topology applied to mobile robot navigation", *Boletín del ACIA núm. 25*, pgs. 24-32, Octuber 2001.

[Museros and Escrig 02a] Museros L., Escrig M.T., "Modeling Motion Qualitatively: The Integration of Topology and Time", Technical Report ICC 2002-01-1, January 2002

[Museros and Escrig 02b] Museros L., Escrig M.T., "Integrating Qualitatively Time and Topology for Spatial Reasoning", *Proccedings of the QR2002. 16th International WorkShop on Qualitative Reasoning.* Editors: Nuria Agell and Juan A. Ortega. ISBN: 84-95499-60-6, pgs. 105-112, Junio 2002

[Museros and Escrig 02c] Museros L., Escrig M.T., "Applying qualitative spatial reasoning combining space and time to mobile robot navigation", *Sistemas Cualitativos y Diagnosis. Automatización del Razonamiento Cualitativo y Aplicaciones.* Eds. Juan Antonio Ortega, Xavier Parra y Belarmino Pulido. ISBN : 84-95499-62-2.,pgs. 63-72, Junio 2002

[Museros and Escrig 02d] Museros L., Escrig M.T. "Combining Qualitative Spatial Information : The Integration of Topology, Time, Orientation and Distances for Spatial Reasoning", *Proccedings of Spatial and Temporal Reasoning ECAI-2002 WorkShop.* Editor: Rita V. Rodríguez., Julio 2002.

[Museros and Escrig 02e] Museros L., Escrig M.T., "Modeling Motion Qualitatively : Integrating Space and Time", *Lecture Notes in Artificial Intligence num. 2504 Topics in Artificial Intelligence. 5th Catalonian Conference on AI, CCIA 2002.* Eds. M. Teresa Escrig, Francisco Toledo and Elisabet Golobardes. Springer. ISBN: 3-540-00011-9, pgs. 64-74, Octubre 2002

[Museros and Escrig 02f] Museros L., Escrig M.T., "Modeling Motion Qualitatively: Integrating Space and Time", *Boletín del ACIA núm.28.* ISSN : 1577-1989, pgs. 115-120, Octubre 2002.

[Museros and Escrig 02g] M. Teresa Escrig, Lledó Museros, Julio Pacheco y Francisco Toledo, "Several Models on Qualitative Motion as instances of the CSP", *Inteligencia Artificial. Revista IberoAmericana de Inteligencia Artificial Número 17/Otoño 2002*. ISSN: 1137-3601, pgs. 55 –71, Noviembre 2002

[Museros and Escrig 03] Museros L., Escrig M.T., "Modeling Motion by the Integration of Topology and Time", *Journal of Universal Computer Science. September Issue . Vol. 9, No. 9, Volumen :9*, pgs.1096-1122, 2003

[Museros and Escrig 04a] Museros L. , Escrig M.T., "A Qualitative Theory for Shape Description", *Inteligencia Artificial. Revista IberoAmericana de Inteligencia Artificial num. 23, Volumen 8,* 2004. ISSN: 1137-3601, 2004

[Museros and Escrig 04b] Museros L., Escrig M. T., "A Qualitative Theory for Shape Representation and Matching for Design", *Proceedings ECAI 2004, IOS Press ISSN 0922-6389*, pgs. 858-862, 2004.

[Museros and Escrig 04c] Museros L., Escrig M. T., "A Qualitative Theory for Shape Representation and Matching", *Proceedings QR 2004, Edited by Johan de Kleer and Keneth D. Forbus.*, pgs. 3-10, 2004.

[Museros and Escrig 04d] Museros L., Escrig M. T., "Qualitative Recognition of objects without holes, with holes and curves", *WorkShop Notes del en Spatial and Temporal Reasoning WorkShop*, en el ECAI 2004, 16th European Conference on Artific ial Intelligence 2004, 2004.

[Museros and Escrig 04e] Museros L., Escrig M. T., "A Qualitative Theory for Shape Matching applied to Autonomous Robot Navigation.", *Frontiers in Artificial Intelligence and Applications. Recent Advances in Artificial Intelligence Research and Development.* ISBN 1 58603 466 9, 2004.

[Museros and Escrig 05] Museros L., Escrig M. T., "Extracting Relevant Features of an Image for Qualitative Shape Matching.", *Proceedings VII Jornadas de Trabajo ARCA, JARCA'05,* 2005

[Musto et al. 99] A.Musto, K. Stein, A. Eisenkolb, and T. R¨ofer, 'Qualitative and quantitative representations of locomotion and their application in robot navigation', in *Proceedings IJCAI-99,* pp. 1067 – 1072, (1999).

[Musto, Stein et al. 00] Musto, A., Stein, K., Eisenkolb, A., Röfer, T., Brauer W. and Schill K. "From Motion Observation to Qualitative Motion Representation."; In *Lecture notes in Artificial Intelligence 1849. Spatial Cognition II. Integrating Abstract Theories, Empirical Studies, Formal Methods and Practical Applications. Spatial Cognition.* Springer-Verlag, pp. 225 – 239, 2000.

[Paritosh 03] Paritosh, P.K., ”A Sketch of a Theory of Quantity”. In *Proceedings of the 17th International Workshop on Qualitative Reasoning,* Brasilia, Brazil, 2003.

[Paritosh 04] Paritosh, P.K., “Symbolizing Quantity”. In *Proceedings of the 26th Cognitive Science Conference*, Chicago, 2004.

[Park and Gero 99] Park, S-H. and Gero, J. S. (1999) Qualitative representation and reasoning about shapes, in *Gero, J. S. and Tversky, B. (eds), Visual and Spatial Reasoning in Design* , Key Centre of Design Computing and Cognition, University of Sydney, Sydney, Australia, pp. 55-68.

[Park and Gero 00] Park, S-H. and Gero, J. S. (2000) Categorisation of shapes using shape features, in Gero, J. S (ed.), *Artificial Intelligence in Design'00,* Kluwer, Dordrecht, pp.203-223.

[Pullar and Egenhofer 88] Pullar, D., & Egenhofer, M. Toward formal definitions of topological relations among spatial objects. *Third International Symposium on Spatial Data Handling,* Sydney, Australia, August 1988.

[Rajagopalan and Kuipers 94] Rajagopalan and B. Kuipers. Qualitative spatial reasoning about objects in motion: application to physics problem solving. San Antonio, TX, March 1994. *IEEE Conference on Artificial Intelligence for Applications* (CAIA-94).

[Randell and Cohn 89] D.A. Randell, and A. G. Cohn. “Modelling topological and metrical properties of physical processes”. In R. Brachman, H. Levesque, and R. Reiter, editors, *Proceedings 1st International Conference on th e Principles of Knowledge Representation and Reasoning,* pp. 55-66, Morgan Kaufmann, 1989.

[Randell and Cohn 92] D.A. Randell, and A. G. Cohn. “Exploiting lattices in a theory of space and time”. *Computers and Mathematics with Applications,* 23(6-9), pp.459-476, 1992.

[Randell, Cui and Cohn 92] D. A. Randell, Z. Cui, and A.G. Cohn. “A spatial logic based on regions and connection”. In *Proc.3rd Int. Conf. On Knowledge Representation and Reasoning*, Morgan Kaufmann, 1992.

[Renz and Nebel 98] Renz, J., & Nebel, B. Spatial reasoning with topological information. In *C. Freksa, C. Habel, & K. F. Wender (Eds.), Spatial cognition - An interdisciplinary approach to representation and processing of spatial knowledge.* Berlin: Springer, 1998.

[Requicha 80] Aristides A. G. Requicha. Representations for Rigid Solids: Theory, Methods, and Systems. *ACM Comput. Surv. 12 (4),* pp: 437-464, 1980.

[Röhrig 97] Ralf Röhrig. ”Representation and Processing of Qualitative Orientation Knowledge”. *KI 1997*, pp. 219-230, 1997.

[Russell 95] Russell, S. and Norvig, P., Artificial Intelligence: A Modern Approach”, Englewood Cliffs, NJ., Prentice Hall, 1995.

[Schlieder 93] Schlieder, C., “Representing Visible Locations for Qualitative Navigation”. In *Proceedings of the III IMACS International Workshop on Qualitative Reasoning and Decision Technologies*, pp. 523-532, 1993.

[Schlieder 95 ] Schlieder, C. “Reasoning About Ordering”. *COSIT 1995*, pp. 341-349, 1995.

[Schlieder 96] Schlieder, Ch. Qualitative Shape Representation. *Spatial conceptual models for geographic objects with undertermined boundaries*, Taylor and Francis, London, 123-140, 1996.

[Shokoufandeh, Marsic and Dickinson 99] Ali Shokoufandeh, Ivan Marsic, Sven J. Dickinson: View-based object recognition using saliency maps. *Image Vision Comput. 17(5-6)*: 445-460, 1999.

[Shokoufandeh, Dickinson et al. 02] A. Shokoufandeh, S. Dickinson, C. Jonsson, L. Bretzner, and T. Lindeberg 2002, " On the Representation and Matching of Qualitative Shape at Multiple Scales" *Proceedings, European Conference on Computer Vision,* Copenhagen, May, 2002.

[Song and Cohen 91] Fei Song, Robin Cohen. "Tense Interpretation in the Context of Narrative". *AAAI,* pp. 131-136, 1991.

[Song and Cohen 96] Fei Song, Robin Cohen: "A Strengthened Algorithm for Temporal Reasoning about Plans ". *Computational Intelligence 12*, pp. 331-356, 1996.

[Steger 96] Steger, C. On the Calculation of Arbitrary Moments of Polygons. Technical Report FGBV-96-05, Technische Universität München, October 1996.

[Stein and Musto 00] Stein, K., and Musto, A. A Computational View on Frames of Reference in Motion. In *Proceedings of the ECAI-2000-Workshop "Current Issues in Spatio-Temporal Reasoning"*. Berlin, 2000.

[Trave-Massuyes et alt. 03] Trave-Massuyes, L., Ironi, L., and Dague, P. "Mathematical foundations of qualitative reasoning", Artificial Intelligence Magazine, Winter 2003.

[Van de Weghe et al. 04] N. Van de Weghe, A. G. Cohn, P. Bogaert and P. De Maeyer, Representation of moving objects along a road network. In *Proc. 12th Int. Conf. on Geoinformatics - Geospatial Information Research: Bridging the Pacific and Atlantic,* pp 187-194, Sweden, June 2004.

[Vilain and Kautz 89] Vilain, M., H. Kautz, and P. van Beek. "Constraint Propagation Algorithms for Temporal Reasoning: A Revised Report". In Readings in *Qualitative Reasoning about Physical Systems*, D. S . Weld and J. de Kleer (eds.), Morgan-Kaufman, 373-381, 1989.

[Weida and Litman 92] Robert Weida and Diane Litman. "Terminological reasoning with constraint networks and an application to plan recognition". In *Proc.of the 3rd Int.Conf. on the Principles of Knowledge Representation and Reasoning (KR-92)*, pp. 282-293, Morgan Kaufmann, Los Altos, 1992.

[Williams 91] Williams, B. "A theory of interactions: unifying qualitative and quantitative algebraic reasoning". *Artificial Intelligence* 51(1–3), pp. 39–94, 1991.

[Wolter and Zakharyaschev 00] Frank Wolter, Michael Zakharyaschev , Spatio-temporal representation and reasoning based on RCC-8, Journal Title: Principles of Knowledge Representation and Reasoning, 2000

[Zimmermann 93] K. Zimmermann, "Enhancing qualitative spatial reasoning - combining orientation and distance," in *Proc. International Conference on Spatial Information Theory. A Theoretical Basis for GIS,* Elba, Italy, pp. 69-76, 1993.

[Zimmermann and Freksa 93] Zimmermann, K., Freksa, C., "Enhancing Spatial Reasoning by the concept of motion", in the *Proceedings of the AISB'93 Conference*, 1993.

# Articles Derived from this PhD Dissertation

**From 1998 to 2001**

During this period of time, we have done the following tasks of this PhD thesis:

1. Development of the State of the Art on Topological Models.
2. Development of the Topological Algebra, suitable for the integration with other features of space thanks to follow the three steps defined in the thesis:
    a. The representation of topology;
    b. The definition of the BSIP;
    c. The definition of the FIP.

From this work 5 articles have appeared:

[Museros 98] Museros, L. "Conceptualizing Regions on the basis of Cartographic Entities", *Wotkshop Working Notes of the WokShop on Spatial Inference. Designing the interface between mental and computational approaches to spatial problem solving*, 1998

This work presents a preliminary state of the art on topological models.

[Isli, Museros et al. 00] Isli A., Museros L., Barkosky T., Reinhard M., "A Topological Calculus for Cartographic Entities. Conceptualizang New Regions", *Lecture Notes in Artificial Intelligence 1849. Spatial Cognition II. Integrating Abstract Theories, Empirical Studies, Formal Methods and Practical Applications.* Springer-Verlag , ISBN : 3-540-67584-1, pgs. 225-238, 2000

This paper presents the topological algebra developed to accomplish the first step of the integration with other spatial aspect, which is the representation of the spatial aspect to be integrated.

[Museros and Escrig 01a] Museros L., Escrig M.T., "Enhancing Topological Information with other Qualitative Spatial Aspects ", *Diagnosis, Razonamiento Cualitativo y Sistemas Socioeconómicos. Carlos Alonso y Juan Antonio Ortega Eds.* ISBN : 84- 95499- 35- 5, pgs. 113-123, Julio 2001.

This paper presents the first steps towards the implementation of the BSIP and the FIP for the topological algebra we have developed.

[Museros and Escrig 01b] Museros L., Escrig M.T. "The integration of topological information, qualitative orientation and positional information using constraint logic programming for robot navigation", *Proccedings of the IJCAI -2001 Workshop on Spatial and Temporal Reasoning with « Agents » Focus . Rita Rodriguez (Eds.),* pgs. 21-27, Agosto 2001.

Taking into account the comments received from the presentation of the article [Museros and Escrig 01a] from the ARCA collective, this paper presents the final solution of the implementation of the BSIP and the FIP for the topological algebra we have developed.

[Museros and Escrig 01c] Museros L., Escrig M.T., " Qualitative Spatial Reasoning using orientation, distances and topology applied to mobile robot navigation", *Boletín del ACIA núm. 25*, pgs. 24-32, Octuber 2001.

Finally, this paper presents an application of the topological algebra developed and integrated with orientation and distances to mobile robot navigation.

**From 2002 to 2003**

During this period of time, we have done the following tasks of this PhD thesis:

1. Development of the State of the Art on Qualitative Movement Models.
2. Development of the Qualitative Movement Theory as the Integration of Topology and Time.

From these tasks the following papers have appeared:

[Museros and Escrig 02a] Museros L., Escrig M.T., "Modeling Motion Qualitatively: The Integration of Topology and Time", Technical Report ICC 2002-01-1, January 2002

Technical Report which presents the preliminary work of the development of the qualitative theory of movement.

[Museros and Escrig 02b] Museros L., Escrig M.T., "Integrating Qualitatively Time and Topology for Spatial Reasoning", *Proccedings of the QR2002. 16th International WorkShop on Qualitative Reasoning.* Editors: Nuria Agell and Juan A. Ortega. ISBN: 84-95499-60-6, pgs. 105-112, Junio 2002

This paper presents the final qualitative theory of movement integrating topology and time. Specifically, it represents the first steps towards integrating the movement theory with other spatial aspects, which is the representation of the feature of space, which in this case it is the representation of movement.

[Museros and Escrig 02c] Museros L., Escrig M.T., "Applying qualitative spatial reasoning combining space and time to mobile robot navigation"*, Sistemas Cualitativos y Diagnosis. Automatización del Razonamiento Cualitativo y Aplicaciones.* Eds. Juan Antonio Ortega, Xavier Parra y Belarmino Pulido. ISBN : 84-95499-62-2.,pgs. 63-72, Junio 2002

This paper explains the movement theory applied to a robotic problem.

[Museros and Escrig 02d] Museros L., Escrig M.T. "Combining Qualitative Spatial Information : The Integration of Topology, Time, Orientation and Distances for Spatial Reasoning", *Proccedings of Spatial and Temporal Reasoning ECAI-2002 WorkShop.* Editor: Rita V. Rodríguez., Julio 2002.

This work presents the integration of the movement theory with other features of space such as orientation and distance by defining the BSIP and the FIP.

[Museros and Escrig 02e] Museros L., Escrig M.T., “Modeling Motion Qualitatively: Integrating Space and Time”, *Lecture Notes in Artificial Intligence num. 2504 Topics in Artificial Intelligence. 5th Catalonian Conference on AI, CCIA 2002.* Eds. M. Teresa Escrig, Francisco Toledo and Elisabet Golobardes. Springer. ISBN: 3-540-00011-9, pgs. 64-74, Octubre 2002

This paper is a first step towards the development of the qualitative movement theory as a CSP model, and the refined and final version of this work is presented in:

[Museros and Escrig 02f] Museros L., Escrig M.T., “Modeling Motion Qualitatively: Integrating Space and Time”, *Boletín del ACIA núm.28.* ISSN : 1577-1989, pgs. 115-120, Octubre 2002.

The next paper presents in a journal, specifically the journal *“Inteligencia Artificial. Revista IberoAmericana de Inteligencia Artificial”*, the integration of several qualitative spatial models integrating space (2-D and 3-D) and time, including the qualitative movement model, developed by the research group C4R2 (Cognition for Robotics Researc) of the UJI, to whom we belong.

[Escrig, Museros et al. 02g] M. Teresa Escrig, Lledó Museros, Julio Pacheco y Francisco Toledo, “Several Models on Qualitative Motion as instances of the CSP”, *Inteligencia Artificial. Revista IberoAmericana de Inteligencia Artificial Número 17/Otoño 2002*. ISSN: 1137-3601, pgs. 55 –71, Noviembre 2002

Finally, given the quality of the article [Museros and Escrig 02d] we were invited to extend the paper and to publish it in a journal, specifically in the *“Journal of Universal Computer Science”*. The result of this extension is the next article:

[Museros and Escrig 03] Museros L., Escrig M.T., ”Modeling Motion by the Integration of Topology and Time”, *Journal of Universal Computer Science. September Issue. Vol. 9, No. 9, Volumen :9*, pgs.1096-1122, 2003

**From 2004 to 2005**

During this period of time, we have corried out the following tasks of this PhD thesis:

1. Development of the State of the Art on Qualitative Shape Representation Theories.
2. Development of the Qualitative Shape Representation and Matching Theory.
3. Development of the Applications presented in this thesis.

From these tasks the following papers have appeared:

[Museros and Escrig 04a] Museros L. , Escrig M.T., “A Qualitative Theory for Shape Description”, *Inteligencia Artificial. Revista IberoAmericana de Inteligencia Artificial num. 23, Volumen 8*, 2004. ISSN: 1137-3601, 2004

This paper describes the initial Qualitative Shape Representation and Matching Theory, which is only able to describe and match shapes of polygonal non-regular objects.

[Museros and Escrig 04c] Museros L., Escrig M. T., “A Qualitative Theory for Shape Representation and Matching”, *Proceedings QR 2004, Edited by Johan de Kleer and Keneth D. Forbus*., pgs. 3-10, 2004.

This paper describes the Qualitative Shape Representation and Matching Theory extended to be able to represent and match shapes of polygonal non-regular objects, and objects with curves or completely curvilinear objects.

[Museros and Escrig 04d] Museros L., Escrig M. T., “Qualitative Recognition of objects without holes, with holes and curves”, *WorkShop Notes del en Spatial and Temporal Reasoning WorkShop*, en el ECAI 2004, 16th European Conference on Artificial Intelligence 2004, 2004.

This paper describes the Qualitative Shape Representation and Matching Theory extended to be able to represent and match shapes of polygonal non-regular objects, objects with curves or completely curvilinear objects, and with one or several holes.

Finally there are several papers which describe the applications developed in this thesis:

[Museros and Escrig 04b] Museros L., Escrig M. T., “A Qualitative Theory for Shape Representation and Matching for Design”, *Proceedings ECAI 2004, IOS Press ISSN 0922-6389*, pgs. 858-862, 2004.

This paper presents the application of the complete Qualitative Shape Representation and Matching Theory for an industrial application, which is to achieve the automatic and intelligent assembly of mural ceramic mosaics.

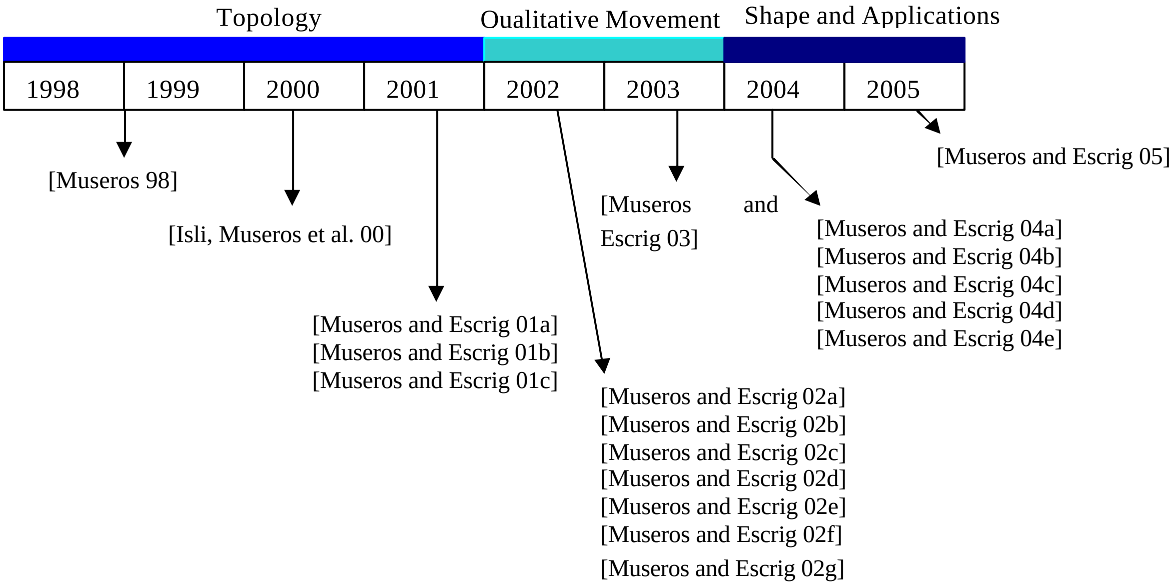
[Museros and Escrig 04e] Museros L., Escrig M. T., “A Qualitative Theory for Shape Matching applied to Autonomous Robot Navigation.”, *Frontiers in Artificial Intelligence and Applications. Recent Advances in Artificial Intelligence Research and Development.* ISBN 1 58603 466 9, 2004.

This paper presents the application of the initial Qualitative Shape Representation and Matching Theory for polygonal non-regular objects together with the Movement Theory for the autonomous robot navigation of a Khepera[2] robot.

[Museros and Escrig 05] Museros L., Escrig M. T., “Extracting Relevant Features of an Image for Qualitative Shape Matching.”, *Proceedings VII Jornadas de Trabajo ARCA, JARCA’05,* 2005

This paper presents how we process the images coming from a vision system in order to obtain the reference points to create the qualitative description of the objects in the image. This process has been applied to the management of pictures of tesseraes for the industrial application developed.

All this work can be represented in a graphic form by the following diagram:


Topology
Qualitative Movement
Shape and Applications
1998
1999
2000
2001
2002
2003
2004
2005
[Museros 98]
[Isli, Museros et al. 00]
[Museros and Escrig 01a]
[Museros and Escrig 01b]
[Museros and Escrig 01c]
[Museros and Escrig 02a]
[Museros and Escrig 02b]
[Museros and Escrig 02c]
[Museros and Escrig 02d]
[Museros and Escrig 02e]
[Museros and Escrig 02f]
[Museros and Escrig 02g]
[Museros and Escrig 03]
[Museros and Escrig 04a]
[Museros and Escrig 04b]
[Museros and Escrig 04c]
[Museros and Escrig 04d]
[Museros and Escrig 04e]
[Museros and Escrig 05]