

**AN INVESTIGATION ON APPLICATION
OF AI TECHNIQUES
ON GIS**

THESIS SUBMITTED FOR THE
AWARD OF THE DEGREE OF

DOCTOR OF PHILOSOPHY

C.RAVINDRA KUMAR



DEPARTMENT OF COMPUTER & INFORMATION SCIENCES
SCHOOL OF MATHEMATICS & COMPUTER/INFORMATION SCIENCES

UNIVERSITY OF HYDERABAD

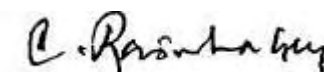
HYDERABAD - 500 046

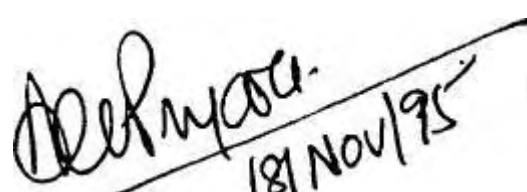
NOVEMBER 1995

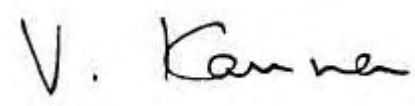
DEPARTMENT OF COMPUTER & INFORMATION SCIENCES
SCHOOL OF MATHEMATICS & COMPUTER/INFORMATION SCIENCES
UNIVERSITY OF HYDERABAD

This is to certify that I, C.RAVINDRA KUMAR, have carried out the research embodied in the present thesis for the full period prescribed under the Ph.D ordinances of the university.

I declare to the best of my knowledge, that, no part of this thesis was earlier submitted for the award of any research degree of any university.


C.RAVINDRA KUMAR


Prof. Arun Kumar Pujari
Supervisor &
Head of the Department


Prof. V.Kannan
Dean of the School

ACKNOWLEDGEMENTS

I am grateful to my guide Prof. A.K. Pujari, Head, Department of Computer and Information sciences for having initiated me in taking up this work and without his guidance, support and help this thesis would not have come to this stage.

My profound gratitude to my guide for having spared his valuable time for this work inspite of his busy schedule. During the course of this work, he has encouraged me a lot and put me in the right track. The suggestions given by him were indispensable and the entire work has proceeded as per his directions. I am grateful for the efforts of my supervisor in acquiring and providing pertinent data for the completion of this thesis.

My profound gratitude to Prof. P.G. Reddy, for having given permission to do Ph.D in Department of Computer/Information Sciences.

My sincere thanks to Director, computer centre for providing necessary resources for doing certain tasks.

I am especially grateful to the Vice chancellor, Dean, School of Mathematics and Computer/Information Sciences, Controller of examinations, and other university authorities for granting extension of time for submitting the dissertation.

During the course of this work, my family members have encouraged me and hence my special thanks to them also.

Finally, thanks to all computer centre operators for extending their help during the course of this work.

(C.Ravindra Kumar)

A B S T R A C T

A Geographical Information System (GIS) is a system used for collecting, storing, retrieving, transforming, and displaying spatial and non-spatial data about geographical objects. Need was felt to use mathematical models, or expert system in order to solve environmental, socio-economic tasks, by making use of spatial and non-spatial data in GIS. So, Artificial Intelligence (AI) is being used in deriving decision support system with GIS acting as spatially oriented database. In this dissertation, AI is used for solving three tasks which are relevant in the context of GIS arena.

An Evidential reasoning technique is developed in this thesis to reveal the information content of a target scene, using multiple sources of information. To deal with uncertain, imprecise information, multiple sources of information viz., topomaps, aerial photographs, satellite images, geologists knowledge which constitute possibilistic information, is to be combined with probabilistic information of a target scene and Evidential reasoning is to be performed by moving through frames, converging on spaces where target questions can be answered, by making use of various evidential operators such as gisting, fusion, summarisation etc., depending on the structure of the evidence domain, to identify features on the target scene. The knowledge-base or gallery is a body of evidence is a collection of frames and compatibility relations among them, and act as an intelligent tool. Basic elements like Tone/Colour, Texture, Shadow etc., are used in case of satellite image to establish gallery for

identifying features like planar surfaces, mountains etc., on a target scene. Evidential reasoning is applied in a typical case study using geologists knowledge, for identifying geomorphological features in topomaps and satellite images. Certain types of features whose presence or absence supports the presence of certain geomorphological features. The frames include the list of features to identify, drainage, area etc.. Various geomorphological features to identify are mountains, irregular hills etc..

Due to changes taking place in environment, there is a necessity for image to map registration of spatial objects of a typical map with the same portion of a satellite image, which will enable to have latest information about spatial objects in the map. In this process map updation, image interpretation, and data acquisition tasks are accomplished. A logic-based approach is developed for the purpose of image to map registration. Spatial object details are arrived by digitization process. Spatial objects chains and regions are considered. Spatial relationships like tee, chi, bounds etc., are arrived using various logics. Axiom set is built using "Prolog" by making use of information about spatial objects, spatial relationships among spatial objects, and rules applicable in a real world scene like rivers do not cross each other, shorelines form closed loops etc.. The possible combination features i.e., spatial reasoned details which are generated and proved "true" in the "Prolog" program or knowledge-base are used for image to map registration. There is

likely of arriving at a single interpretation if more facts are added to knowledge-base.

An Object-Oriented gateway to CIS is designed in this thesis, which consist of object-oriented frontend and graphical user interface. Object-Oriented data model has features of traditional database system, semantic data models and object-oriented programming. Features of object-oriented programming includes complex object, object identity, class and methods, encapsulation, inheritance and extensibility. Geometric entities are complex in nature and inherit features of other entities. Modelling of structural objects, integration of efficient access methods, use of proprietary GIS database have resulted to use object-oriented DBMS for GIS databases. So, object-oriented GIS supports class hierarchy, behavioral object orientation, and structural object orientation. The approach followed include providing object-oriented frontend which uses object-oriented data modelling concepts which the underlying GIS lacks. The object-oriented features are augmented to GIS by the frontend. Underlying GIS is used for data management. The first phase of object-oriented design is analysis, which include object-oriented concepts inclusion, establishing graphical user-interface (GUI), providing communication between GIS and frontend via parser and high level abstraction for an application by providing 'schema-building' facility. System design describes the approach to solve the problem, and has four sub-systems viz., schema manager, class manager, object manager and system manager. During object design various objects are designed with emphasis on data structures and

algorithms to implement each class by making use of analysis model and strategies of system design. The last phase in object-oriented design is implementation in which classes and relationships described are translated in to programming language, database. System manager receives requests from object manager, schema manager, and transaction part and are transferred through parser to GIS. The replies from CIS are communicated to object manager and schema manager. To reduce training effort on the part of GIS user, GUI is provided by which graphic objects can be dragged, scroll bars can be scrolled etc., and interaction with objects is by making use of pointing devices such as mouse.

TABLE OF CONTENTS

CHAPTER-I

1.0 INTRODUCTION AND OVERVIEW

1.1 Spatial information	1
1.2 Characteristics of spatial data	3
1.3 Geographical information system (GIS)	5
1.4 Components of GIS	7
1.5 GIS features	10
1.6 Expert system	11
1.7 Expert database system	12
1.8 Reasoning	13
1.9 Evidential reasoning	17
1.10 Motivation	19
1.11 Aim of the dissertation	20
1.12 Present work and organisation	21

CHAPTER-II

2.0 REASONING USING MULTIPLE SOURCES OF INFORMATION

2.1 Introduction	33
2.2 Spatial reasoning	34
2.3 Earlier work on spatial reasoning	35
2.4 Need for evidential reasoning	42
2.5 Theoretical basis	43
2.6 Implementation	45
2.7 Case study of evidential reasoning using expert's knowledge	66
2.8 Geomorphological features to be identified	66
2.9 Knowledgebase for the identification of Geomorpholo- gical features for satellite images and topomaps	71
2.10 Gallery	73

CHAPTER-III

3.0 IMAGE TO MAP REGISTRATION PROBLEM

3.1 Introduction	85
3.2 Logical formulation	86
3.3 Spatial objects and spatial relationships	87
3.4 Axiom set building	92
3.5 Implementation	95

3.6 Outline of implementation	97
3.7 Steps followed for spatial reasoning	98
3.8 Conclusions	99
CHAPTER-IV	
4.0 OBJECT-ORIENTED GATEWAY TO CIS	
4.1 Introduction	100
4.2 Theoretical basis	101
4.3 Need for object-oriented GIS (OOGIS)	107
4.4 Object-Oriented CIS (OOGIS) vs GIS	108
4.5 Need for cooperative environment	110
4.6 Need for graphical user interface (GUI)	111
4.7 Aim of Object-Oriented gateway to GIS	112
4.8 Object-Oriented data model	113
4.9 Different approaches to OOGIS	124
4.10 Approach followed for OOGIS	126
4.11 Design	128
4.12 Analysis	133
4.13 System design	142
4.14 Object design	148
CHAPTER-V	
5.0 CONCLUSIONS AND FUTURE DIRECTIONS	156
REFERENCES	165

CHAPTER-I

1.0 Introduction and Overview

1.1 Spatial information:

Since the introduction of computers four or five decades ago the meaning of the word computation has kept expanding, whereas computation traditionally with numbers, today we routinely compute pictures, texts and many other types of objects. When classified according to the types of objects being processed, three types of computer applications stand out as having shaped the development of computer science.

The first generation involved numerical computing, applied mainly to scientific and technical problems. Data to be processed consisted almost exclusively of numbers or, set of numbers with a simple structure, such as vectors and matrices. Programs are characterised by long execution time but small sets of input and many new numerical algorithms, were invented. Lasting achievements of this first phase of computer applications include systematic study of numerical algorithms, error analysis, the concept of program libraries and the first high level programming languages i.e Fortran and Algol.

The second generation, hatched by the needs of commercial data processing, led to the development of many new data structures. Business applications thrive on record keeping and updating text, processing and report generation: there is not much computation in the numeric sense of the work, but a lot of reading, storing, moving and printing of data. In other words, these applications are data intensive rather than computation

intensive. By focusing attention on the problem of efficient management of large, dynamically varying data collections, this phase created one of the core disciplines of computer science: data structures and corresponding algorithms managing data, such as searching and sorting.

We are in a third generation of computer applications, dominated by computing with geometric and pictorial objects. This change of emphasis was triggered by the advent of computers with bit map graphics. In turn, this leads to the wide spread use of sophisticated user interfaces that depend on graphics, and to a rapid increase in applications such as computer aided design (CAD), image processing and pattern recognition (in medicine, cartography, robot control) that deal with data embedded in space. The young discipline of computational geometry has emerged in response to the growing importance of processing of geometric objects and a variety of spatial data structures have been designed to support the kind of proximity based access that dominated spatial data processing.

At the risk of over simplification, the dominant impact on the development of data structures [9] due to these three phases are as follows:

1. Arrays for central memory, single key sequential files on external storage media.

2. List structures, primary balanced trees for single access, both for central memory and disk. Inverted files, based on the concept of one primary key and a number of secondary keys, provide an expensive form of multi-key access.

3. Multi-key access structures that treat all keys in a symmetric manner, in particular metric data structures that organise the cartesian product space in which objects to be stored are embedded.

1.2 Characteristics of spatial data:

A spatial data model [14] is a collection of well defined concepts to express both static and dynamic properties of data intensive applications. Static properties are spatial objects, attributes and relationships among spatial objects. Dynamic properties are operations on spatial objects and relationships among operations etc.. Static properties are expressed using database schema, dynamic properties are specifications of transactions and query language. The standard approach for spatial data modeling is to model spatial and non-spatial data separately. So the data models are required to represent spatial information, relationships of spatial objects and support spatial operations on the spatial objects.

The characteristics of spatial data structures have evolved in response to novel requirements [40] that distinguish spatial data from other kinds. The two most important characteristics are:

1. Objects are typically not accessed via a unique identifier such as a part number, but because they lie in or near some region of space.

2. Object representation is an integral part of the data structure.

If we only had to store points in space, the tuple (x,y,z)

of coordinates would make a point behave like the keys of any other kind of data stored. But spatial data applications introduce not only points, but many kinds of primitives (such as line segments, triangles, aligned rectangle etc.), and composite objects made up from these primitives. Objects typically overlap cell boundaries introduced by the data structure, and the difficult question "where *is an object anchored?*" is characteristic of spatial data. The same spatial object will be treated in different ways in different applications. The same spatial object content can be used differently at different periods. Spatial object view of one person may be different from others. The semantic meaning to spatial object and relationships will change from user to user for using them in respective area of work. There are no standard techniques for standardising spatial object contents.

It took time to recognise the impact of the unique characteristics of spatial data on data structures that handle them efficiently. In the 1970s and 1980s, the database community lumped every kind of data other than fixed format records (text, pictures, voice and spatial data). It was felt to extend relational database technology, with its simple conceptual structure, to handle all kinds of data. However, relational data is not just a way to represent data, it also implies or suggests certain access algorithms that are particularly efficient on data naturally represented by rows and columns. Indeed, much business data is of tabular form that lends itself to such regular access patterns. However, if spatial data is tried to be represented in

tabular form, for example by introducing relations polyhedral faces, edges and vertices, two harmful consequences follow.

First, geometric proximity is not reflected by proximity in memory. In the example above, all vertices no matter how far apart in space are stored contiguously in the same relation, whereas a vertex and its incident edges and faces are scattered all over the storage. This has grave consequences when data is stored on disk where instead of accessing one entire object as unit we may have to gather bits and pieces of this object in many separate disk accesses.

Second, the embedding space is not directly represented, but only indirectly through objects that may happen to be there. For example, the query find an empty region of a given size and shape is awkward and inefficient to answer if the data structure only knows explicitly about objects and not about cells (empty or inhabited) of a space partition that exists independently of the objects present.

As these lessons are slowly learnt, a variety of spatial data structures emerged over the past two decades that specifically aim to represent euclidean space as well as the objects that may inhabit it.

1.3 Geographical information system (GIS):

A geographical information system is a set of computer tools for collecting, storing, retrieving, transforming and displaying spatial data about geographical objects and non-spatial attributes of these objects. Geographical objects include natural phenomena

(such as lakes, rivers and forests), man made structures (such as dams, buildings and high ways) and other convenient objects that may define the location and extent of a geographical phenomena (such as a particular soil type),

A spatial information system [36] is a special kind of CIS and may be viewed as a database system in which most of the data are spatially indexed and upon which a set of procedures operates in order to answer queries about spatial entities represented in the database. Spatial characteristics are to be included in Geographical Information System (GIS) in order to answer queries involving geometric procedures.

Spatial data specifies the location and relative positions of objects. A vector representation uses points, lines and areas or polygons and a raster representation uses an (x,y) [19] grid to define the spatial location and relative position of objects. _

Non-spatial attributes, such as the assessed value of a building, the acidity level in a lake or the soil strength of a particular type of soil, are associated with objects. In a vector based CIS, non-spatial attributes are associated with a point, line or polygon used to represent the object of interest. In a raster based CIS, non-spatial attributes are associated with a grid cell. A group of grid cells representing a particular type of soil can be considered an object in the sense that spatial information and non-spatial attributes are stored in separate databases within the CIS. The database allows the attributes to be queried and objects associated with the attributes to be displayed. Selective display of features is a major capability of

a CIS and distinguishes a CIS from a computerised map. This relationship between objects and attributes gives CIS, powerful capabilities for analysing natural resources problem.

In the conventional information system where the information is retrieved from relational tables using certain simple logical operations controlled by relational algebra or, calculus, whereas in a GIS, the information is not available explicitly and is required to be computed by the help of certain algorithms. It is worthwhile at this point to note the structure of a knowledge-based system where the implicit information is derived by the help of certain inference from the explicit information in conjunction with the domain knowledge. Knowledgebase GIS aims at providing a framework in which explicit information can be retrieved from the database, the implicit information can be obtained by proper combination of two processes, one is the computation and the other is derivation.

1.4. Components of GIS:

The components of a GIS include data collection, data management, retrieval, transformation and display [38].

1.4.1 Data collection:

Data collection deals with objects and attributes. Information on objects is gathered by scanning or, digitizing existing maps, from aerial surveys, remote sensing and ground survey data. Capturing spatial data in a format suitable for GIS that are cartographically and topologically correct is a tedious task. Spatial data building requirement add complexity to data

management. Attribute data may be gathered about spatial objects from a number of sources.

1.4.2 Data management:

The data management component of a CIS involves the storing of spatial data and attribute data about objects. Separate mechanisms are used to store the spatial and attribute data. Objects have a spatial dimension and require spatial handling systems that are designed for spatial data representation and manipulation. Spatial data structures can be organised around grids (rasters) [19] or vectors (topological) forms. Object attribute data is normally managed using relational model. The attributes for an object are contained within a tabular structure and given a unique identification (ID), object is stored using the same unique ID, so that spatial and attribute data associated with an object can be related to the object with this ID. Object spatial data and attribute data are stored separately.

1.4.3 Retrieval:

The retrieval component deals with selecting objects based on spatial data or, attribute value. Much of the spatial analysis capability of a GIS is contained in this component. For example, a user of a GIS may derive to find all objects that are within X km of a given position and have a particular value for an attribute. To be more concrete, assume the user of a GIS wants to determine the assessed value of all commercial buildings located within the flood plain area as determined by a water surface profile model. First, assuming that commercial buildings are objects included in

the CIS database, a search can be performed to determine which commercial building objects are located within the confines of the flood plain. The query would require a point in polygon analysis. The relational database management system can then be queried to find the assessed value attribute of the commercial building objects that were identified as being in the flood plain and calculate the sum of the assessed value.

1.4.4 Transformation:

The transformation component allows GISs to create new information by combining object and attribute data into a new configuration. The transformation component gives CIS the capability to create new information by combining spatial objects based on the value of attributes. Each layer of information depending upon single attribute will be stored. Such multiple layers of attributes information will be available in CIS. Spatial analysis capabilities in these components allow CIS, to combine information from different layers to create new layers of information. This capability separates CIS from manual maps or digital mapping procedures.

1.4.5 Display:

The display component provides spatial representation of geographic components and textual lists of attributes. The display component can selectively display objects and attributes based on the results of a query as opposed to a static map or list. This dynamic capability allows CIS greater flexibility in displaying spatial and attribute information.

1.5 CIS features:

The retrieval and transformation of data in CIS are done with a primary and compound tools. The primary transformation tools include simple spatial analysis tools, such as distance between objects and point-in-polygon operations, database queries and boolean manipulations of data layers. Most GISs use only primary transformation tools when manipulating features and attribute data.

Compound transformation tools include mathematical models or, expert systems [42] that can operate on and interact with object and attribute data to produce new information. A compound transformation tool example would be the use of a mathematical model for determining the extent of flood plain. Geographic information systems (GISs) offer spatial data management and analysis tools that can assist experienced and skillful users in organising, editing, analysing and displaying positional and attribute information about geographical data. As yet, GISs offer little modeling capability with most offering only primary modeling tools such as map-overlays and buffering. This is especially unfortunate in the context of natural resource planning and management, because mathematical models are important tools for analysing these issues. The models provide insight into the problems by representing physical, environmental, economic and/or social processes. These models include simulation models where these processes are simulated to test alternative scenarios and optimisation models where objects are specified and parameters are adjusted to meet the objectives. However few such models have a

well developed capabilities to analyse and display spatial information. Many models work around the spatial aspects of a problem by simplifying assumptions and parameterization. Clearly, CIS applications could benefit from modeling capability of these models, and these models could benefit from the spatial analysis and display capability of CIS.

1.6 Expert system :

Growing interest to store, retrieve and manipulate symbolic data along with attribute data led to the development of various techniques for solving varieties of problems which will arise to different spatial information systems. This idea is responsible for development of expert systems or knowledge-based systems [37]. Expert Knowledge is a collection of principles and procedures used by human experts in solving a certain domain of problems which are complex and ill-structured. Knowledge is represented using logic, production rules, semantic nets and frames etc..

Semantic nets, frames are declarative methods in which knowledge is represented as a static collection of facts accompanied by a small set of general procedures for manipulating them. In procedural method the bulk of knowledge is represented as procedures for using it for systems using probabilistic, default and heuristics. In some applications declarative and procedural methods of knowledge representations are combined. The main goal is to select a right method that allows all the knowledge to be represented and that facilitates its use in solving a task at hand. The search in knowledge-base is done by using inference

rules. Interactive messages are also possible.

1.7 Expert database systems [1]:

An expert database system involves a combination of database management system and expert System (ES). The Expert System is used to perform intelligent processing of information being stored in a Data Base (DB) . So expert database system can be defined as a system for developing applications requiring knowledge directed processing of shared information. The user will access expert database system via expert system interface.

The categories of expert database system are given here.

1.7.1 Intelligent database:

In the first class of systems, involving ES-DB interaction, set of routines are used to arrive at a particular component of DB resulting in an intelligent- database. The purpose of intelligent database is to improve both efficiency and functionality of the DBMS. Efficiency can be achieved by query optimization. AI techniques are available for handling incomplete data to get multiple user views to improve the functionality of the system.

1.7.2. Enhanced expert system:

Systems using the enhanced expert system (ES) approach incorporate extended data management facilities into ES. One way is to extend the programming language in which the ES is written and another way is to either down loading the data as a snapshot to ES from the database prior to the working with ES, or the system can be integrated in which case there is a dynamic link

between them and data is retained.

1.7.3. Inter system communication:

The two systems ES and DB are allowed to exist as independent systems and provide the same form of communication between them. An independent intermediate system can be introduced in between ES and DBMS to provide the interaction between them. The primary concerns are secondary storage search pattern for database management system and the search cycle of expert system.

1.7.4 Deductive database management system (DDBMS) [24]:

The DDBMSs are important links between conventional databases and fact and rule base knowledge management. Logic provides some of the powers of deductive database management system. Typical database contributors are the notion of the schema, the automatic maintenance of integrity constraints and the concern about efficient secondary storage management.

1.8 Reasoning:

Inferring new information from available knowledge or proving an information in the set of knowledge is called reasoning. An efficient reasoning system must reveal the information content of an observed event based on available knowledge.

1.8.1 Reasoning using logic [12]:

Knowledge is represented in the language of logic as a set of formulae (axioms). Reasoning in this context is the methodology used to prove any formula or predicate (expressed using logic) in

the available knowledgebase. Matching and substitution are the processes involved in proving a predicate which will be given as goal statement. Resolution which attempts to show that the negation of a statement produces contradiction is also used in order to prove the statement.

Systems developed using logic are monotonic in nature in the sense that the number of statements can be added and new logics can be proved, but this will not cause earlier statements or predicates invalid. Logic programming is based on procedural interpretation of set of clauses. It executes procedure calls left to right and tries alternative procedures once at a time. Order is decided on the way they are written. Logic databases can deal with incomplete information using disjunction and existential quantifiers without relaxing the underlying formalism. Selective back tracking, loop detection, limited use of bottom up execution are some control methods. Recursion can be conveniently used in logic databases. Free variables are instantiated during the process for evaluating a true interpretation. The query to be proved in logic database must be a logical consequence of the set of clauses.

1.8.2 Reasoning using rules [7]:

Rules are used to represent strategies or directives. In a rule based expert system, the knowledge is represented as sets of rules that are checked against a collection of facts about current situation. When the IF portion of a rule is satisfied by the facts, the action specified by the THEN portion is performed. When this happens the rule is said to execute or fire. The action taken

when the rule fires may directly provide the needed result or may modify the set of facts in the knowledgebase (ex: adding a new fact).

The matching of IF portions of rule to the facts will produce inference chain. The two ways in which rules can be used in a rule based expert system are forward chaining and backward chaining reasoning techniques. In forward chaining reasoning technique the search for new information proceeds in the direction of the arrows separating the left hand and right hand sides of the rules. The system uses information on the left hand side to derive information on the right.

If the goal is to infer one particular fact, forward chaining reasoning technique could waste lot of time and backward chaining reasoning might be more effective. The difference in these two approaches is the way the facts and rules are searched.

1.8.3 Reasoning using frames and semantic nets:

Hierarchically structured knowledge can be represented in frames and semantic nets. Distinct types of knowledge can be easily distinguished and handled, and explicit hierarchical organisation of knowledge is obtained. Inheritance is one method used for inference purpose. For non-trivial applications it is necessary to use a rich frame formalism including procedural components. Frames can be developed to invoke rules or horn clauses also.

1.8.4 Nonmonotonic reasoning [6]:

Sometimes the information gathered or arrived is not sufficient to meet the needs of the users for reasoning purpose*. The addition of one piece of information may force the deletion of another i.e., statements that are derived in this way depend on the lack of belief in certain other statements. Some conclusions can be made as long as no contradictory evidence is present. The construction of these conclusions is known as default reasoning [35]. Autoepistemic logic [23] using belief values and ignorance using self knowledge also come under this. Conclusions are assumed to be tentative. It may have to be retracted after new information is added. As nonmonotonic reasoning systems require additional information for proving theorems based on changes which may take place from time to time, storage space and processing time will be more when compared to monotonic systems, where theorems once proved need not be verified again.

1.8.5 Reasoning under uncertainty [17]:

For AI systems to produce adequate assessments of the state and behavior of the real world, they must cope with information and knowledge that is characterised by varying degrees of uncertainty, ignorance and correctness. Thus in any knowledge-based system, the importance of the knowledge acquisition and the organisation of the required knowledge can in no way be over emphasised. Similarly, reasoning process using the knowledgebase is another important component in any successful knowledge-based system. The management of uncertain information in

the first generation expert systems, when addressed at all, has largely been left to adhoc methods. This practice has been effective only because operational expert systems normally assume that the knowledge is complete, precise and unvarying. This fundamental assumption is a principal source of limitation. The uncertainty in knowledge arises out of two phenomena:

- 1) subjectivity of observation
- 2) incompleteness in knowledge

For example, consider the rule *if the object looks red then the object is red*. There is some kind of uncertainty in saying that the object looks red. This rule can also be defeated as it is not always true that only objects looking red are actually red.

The existing approaches to represent uncertainties can be subdivided into two basic categories according to their quantitative or qualitative characteristics. There are three distinct approaches of qualitative methods, which differ in the semantics of their numerical representation, namely:

- single valued (ex: classical probability theory, Bayer's rule)
- interval valued (ex: Dempster shafer theory, evidential reasoning) and
- multi-valued (ex: fuzzy logic theory)

1.9 Evidential reasoning [29] [33]:

Evidential reasoning supports reasoning with interval valued uncertainties using knowledge from multiple sources. The knowledgebase module can be interfaced to any CIS or, image

processing systems. In fact it provides a tool for knowledge representation where the independent pieces of knowledge are expressed by a network that describes the interrelationships among several bodies of evidences.

The choice of an approach based on the Dempster-Shafer theory was not arbitrary. It is established that this theory confers important methodological advantages, such as its ability to represent ignorance in a direct and straight forward fashion, its consistency with classical probability theory, its compatibility with boolean logic and its manageable computational complexity.

Evidential reasoning is used to assess the effect of all available pieces of evidence upon a hypothesis, by making use of domain specific knowledge. In order to apply evidential reasoning to a given task, the first step is to delimit a prepositional space of possible situations and in the theory of belief functions this prepositional space is termed as the frame of discernment. A frame of discernment delimits a set of possible situations exactly one of which is true at any one time. Once a frame of discernment has been established, prepositional statements can be represented by subsets of elements from the frame corresponding to those situations for which the statements are true.

Domain specific knowledge is defined in terms of compatibility relations that relate one frame of discernment to another. A compatibility relation describes which elements from two frames can simultaneously true. Various evidential operators such as fusion, summarisation, discounting, translation, gisting, and interpretation are used depending on the structure of evidence

domain, to assess evidence by operating on evidence domain. The evidential reasoning approach focuses on a body of evidence which is a collection of interrelated beliefs.

1.10 Motivation:

More recently expert systems and GISs have been combined to provide distributed, knowledge-based design and analysis tools for use in environmental and natural resources planning. Robinson et al [4], provided a review of expert system application in GIS and conclude that lack of tools for building expert systems and formalism in GIS are substantial obstacles to the development of large scale integrated expert GIS application. The major areas of interest in expert GIS reviewed are map design, terrain analysis and geographic database management. Kofran [22] reviews the literature related to Artificial Intelligence (AI) and GIS, and finds that there has been very little use of AI in GIS applications, but that many developers are interested in the possibilities of such integrated software systems. Two major uses of AI related to GIS are discussed.

- 1) AI as an analytical tool integrated into GIS
- 2) Knowledge-based GIS where AI plays a predominant role in deriving a decision support system with GIS acting as a spatially oriented database.

The latter mode of application, which places AI techniques at the center of the problem solving method and GIS in the position of spatial data analysis and management is the type of system described in the dissertation.

1.11 Aim of the dissertation:

AI is playing a predominant role in CIS arena and it is being used as a problem solving technique. In this dissertation AI is used in solving tasks with respect to three areas which are relevant in the context of GIS. They are described below as three aims of this thesis.

The first aim of this dissertation is to design an evidential reasoning methodology for revealing the information content of a target scene. In this process features like mountains, domal hills, plateau etc., can be known on the scene. For this purpose knowledge is represented in the form of a gallery of frames and compatibility relations among frames which delimits the space of possibilities. The knowledgebase is a body of evidence which is a collection of inter related beliefs and act as an intelligent tool. As the information is not clear and ambiguous, the reliable information obtained already is to be linked with puzzling features, and evidential reasoning is to be carried out by moving through frames in the gallery by establishing paths converging on spaces where target questions are answered using various evidential operators such as fusion, summarisation, gisting etc..

The second aim of this dissertation is to develop a methodology for solving image to map registration problem. The information content in a map may not be fully true due to changes taking place in environment to various image primitives. So, the characteristics of various features of image primitives present in an image are to be properly registered to map primitives of the relevant map. In this process image interpretation, map

updatation, and data acquisition tasks are solved.

The final aim is to design an object-oriented gateway to existing CIS which includes object-oriented frontend to existing CIS (as it is much better if existing CIS is converted to object-oriented CIS (OOGIS), as developing OOGIS is costly), so that a fully functional, flexible, user friendly and application independent GIS can be established by making use of benefits of semantic data modelling concepts such as aggregation and generalisation, object-oriented programming features like object, class, polymorphism, encapsulation etc., apart from traditional database features such as persistence, sharing etc., and providing graphical user interface (GUI).

1.12 Present work and organisation:

The present work and organisation is described in the following paragraphs.

The first chapter titled 'Introduction and Overview' starts with explaining about evolvement of spatial information finally explaining about the present day tasks which involves bit map graphs. In the next section the characteristics of spatial data are explained in which the importance of spatial data modelling, its concepts are described, highlighting the deficiencies to represent spatial data in tabular form.

A GIS is a set of tools for collecting, storing etc., of spatial and non-spatial data about spatial objects and spatial database is a special kind of GIS, is described along with components of GIS viz., data collection, data management.

manipulation etc.. The next section deals with features of CIS, which include use of primary and compound tools for retrieving and transformation of data in GIS. The definition of expert system, expert database system, and deductive database management system are explained next. The purpose of reasoning, reasoning using logic, reasoning using rules, reasoning using frames and semantic nets etc., are described in subsequent sections of chapter I. The sections following this explains about nonmonotonic reasoning, reasoning under uncertainty, and evidential reasoning. The motivation in taking up solving the tasks in this dissertation using AI as a problem solving method and GIS in the position of spatial data analysis and management is described in the next section followed by aim of the dissertation viz., reasoning using multiple sources of information, image to map registration problem, design of an object-oriented gateway to GIS.

Chapter II deals with reasoning using multiple sources of information. It starts with 'Introduction' to this chapter. Meaning of spatial reasoning, definition of spatial reasoning by various authors, earlier work on spatial reasoning viz., algebraic approach to spatial reasoning, design of *fuzzy* logic based expert system, intelligent GIS, and logical frame work are explained in the subsequent sections of chapter II. To deal with imprecise, uncertain information, multiple sources of information viz; topomaps, aerial photographs, geologists knowledge etc., are to be properly merged with the probabilistic information of the target scene and reasoning has to be carried out using this information to identify various features on the scene. The knowledgebase will

act as an intelligent tool. Knowledge is represented as a gallery of frames and compatibility relations among frames that delimits the space of possibilities. This knowledgebase or gallery is used to reveal the information content of a scene. So, the need for evidential reasoning is explained in the next section. The architecture and block diagram is shown in the next section. Details of terms used in this chapter viz., frame of discernment, basic probability assignment, belief function, focal element are explained in the subsequent section.

Next section deals with implementation details of evidential reasoning. The framework for implementing, involves specifying a set of distinct propositional space (frame of discernment), arriving at compatibility relations, assigning a quantitative belief for the propositional space, and reasoning by establishing paths for these bodies of evidence to move through frames using various evidential operators (summarisation, gisting, fusion etc.) converging on spaces where target questions are answered. The major steps involved are knowledgebase creation and reasoning. The next section of chapter-II deals with building knowledgebase. Frames with compatibility relations among them constitute a gallery or knowledgebase. The steps involved in creation, modification etc. , of gallery or knowledgebase are viz., create a gallery, update a gallery, work as a pre-defined gallery, delete a gallery. The reasoning for identifying features of a scene using the possibilistic information in the gallery and probabilistic information in the target scene is described in the next section. The basic elements like tone/colour, texture, shadow

etc., aid in identification of features on satellite images are described. The options available for evidence domain, and types of evidences are also explained.

Dempster's rule, optimistic rule of combination, Bayesian approximation, Barnett's technique for singleton hypotheses etc., algorithms are used while using fusion operation operating on the way the evidence domains and evidences are used. A case study of evidential reasoning by making use of expert's knowledge is described in the next section in order to identify the feature information on topomaps and satellite images. Various geomorphological features to be identified are mountains, inselbergs/born hads, irregular hills etc.. Certain types of features whose presence or absence in topomap or satellite image supports the presence of certain geomorphological features. Geologists who are experts in this area provide the information, which consists of features in topomaps and corresponding hypotheses (eg: closely spaced contour lines, widely spaced contour lines etc.), features from FCC image (tone is light grey, tone is dark grey etc.) with corresponding hypotheses, features from band-4 image (uniform tone with texture, grey tone etc.) with hypotheses and some more evidences supporting geomorphological features. Then this information is organised suitably to build knowledgebase or a gallery which consists of frames and compatibility relations. Alternative way is described in next section in which frames are described with each frame for one feature such as sand, sand dunes etc., and are used to identify features. Sand (s_1, s_2) is a frame where s_1 indicates vast areas of

light to medium yellow tone on FCC and s_2 indicates absent of vast areas of light to medium yellow tone on FCC. Compatibility relations for each frame with hypotheses are described in the next section.

Chapter III discusses the image to map registration problem. It starts with 'Introduction' to the chapter. In this chapter, logic based approach is followed for the purpose of image to map registration. The map may not contain the latest information, due to changes taking place to the the geographic features. So, need is felt to register the characteristics of a typical feature in image with same feature in a map which helps in arriving at correct inferences from map. Also interpreting an image from the bare lines present on it, play a crucial role in order to perform spatial reasoning, for example to know all roads leading to a certain destination. Logical formulation is explained in the next section. Spatial objects viz., chain, region and spatial relationships tee, chi, interior etc., among various spatial objects can be arrived using various logics is described in the next section. The various facts in real world scene i.e., rules that are applicable in real world scene, such as rivers do not cross each other, rivers can not form closed loops etc., are explained. Axiom set building which constitute various spatial objects and their relationships and rules applicable in real world scene is described in next section. The implementation details are described next. The possible combinations of features are generated and used as goal statements in the prolog program (facts (spatial objects and spatial object relationships) + rules) in

order to check the feasibility. The interpreted features set proved 'true' in the prolog program are updated in to a dynamic database and there is likely arriving at single interpretation if more facts are appended to the fact set in the knowledge base. These spatial reasoned information is used for map updation, image interpretation, and data acquisition purposes. Steps followed for spatial reasoning are described next. They are defining processing/mapping rules to check feasibility, input the image primitive coordinates or digitizing the image, arrive at image object relationships, generate all possible spatial reasoned or interpreted details, and test the feasibility of various combinations and output the results. Conclusions are described in the last section of chapter-III.

Chapter IV deals with developing an object-oriented gateway to existing CIS. This chapter deals with the theoretical basis giving details of object, object identity, class, polymorphism etc.. The need for Object-Oriented CIS (OOGIS) is explained. Geometric entities are complex structured in nature and possess inherited features of other entities. Integration of efficient access methods, modelling of structural objects, use of proprietary CIS database resulted to use object-oriented DBMS for CIS databases. Object-Oriented CIS supports class hierarchy, structural object orientation which provide efficient means to model complex geographical objects. All topological relationships can be stored for each instance in object-oriented databases (OODB). Behavioral object orientation allows to facilitate support of user defined data types and operators for a particular

application. So, using object-oriented database design real world models of spatial phenomena in CIS can be created, because it offers sophisticated tools. The OOGIS vs CIS is explained in the next section. The comparisons such as generic concepts, adhoc query facility etc., are described. Large amount of data from multiple sources viz., satellites, ground observation, aerial photographs etc. , is to be accessed, processed and transformed. This necessitates need for cooperative environment is described next. Need for graphical user interface (GUI) to have user friendly system and to reduce training effort is explained next.

The Object-Oriented data model (OODM) which uses traditional database features such as persistence, sharing, concurrency etc., semantic data modelling concepts such as aggregation, generalisation along with object-oriented programming concepts such as complex object, object identity, class, encapsulation etc., is dealt in next section. Semantic data models are explained next, which have more expressive power and more powerful abstraction over other models. Aggregation which refers to abstraction represented in hierarchical structure and generalisation refers to an abstraction enabling a class of individual objects to be thought of generally as a single named object. Class hierarchy and inheritance is described. Because of multiple super classes multiple inheritance and class lattice exist, and these features are explained.

Additional features of OODM are described. A composite object is defined as an object with a hierarchy of exclusive objects that form a tree structure. The record of evolution of data type i.e..

version control is another feature of OODM. Object can have different representations that are equivalent, and this feature is available in OODM. Data operations in OODM are dealt separately. This includes schema definition, database creation, data retrieval and data update. An interface language can be used to perform these operations. The two methods for building object-oriented databases (OODB) are extending a relational system to support the concepts of objects, and extending object-oriented programming language to include persistence, sharing etc., features to databases. In the next sections OODM which support construction of composite objects and support user defined types and operations are discussed.

Success of CIS lies on, proper modelling of real world phenomena and this is discussed in separate section. So the use of accurate models and efficient access methods are vital in modelling real world phenomena. Different approaches to OOGIS such as the extension of an existing CIS with OODB functionality (capture, analysis etc., features of spatial data are availed by CIS, and storing management and other object-oriented features are to be enhanced over existing CIS), extension of an existing DBMS by geometric and geographical functionality.

Approach followed to OOGIS is explained in next section and this is an extension to an existing CIS. Underlying CIS is used for spatial database management. Object-Oriented frontend implements the object-oriented modelling capabilities. Parser is responsible for communication between frontend and underlying CIS. The frontend will augment features of object identity, structure

etc., to the underlying CIS. Environment is provided for schema building and GUI (graphical user interface). In GUI, graphic objects can be dragged, scroll bars can be scrolled etc., and pointing devices such as mouse can be used to interact with objects. The first phase in object-oriented design is analysis, which is nothing but building real world phenomena model that describes the objects, their relationships and dynamic flow of control i.e., it is an abstraction of what the system should do. The next section is system design and during this phase overall structure is determined by making use of analysis structure and proposed architecture. The third stage is object design uses analysis model and strategies of system design to arrive at design of various objects and main emphasis is on data structures and algorithms to implement each class. The last stage in object-oriented design is implementation in which object classes and relationships developed are translated into programming language, database or hardware implementation. Also GUI which act as user friendly is provided, and this is also described in next section.

The tasks of analysis phase are described in the next section viz., object-oriented concepts inclusion, establishing GUI, providing communication between GIS and frontend via parser are described in the next section along with high level abstraction for an application by providing Schema building facility. Operations for modifying a schema are explained in next section. They are given below.

1. Schema building: It includes organisation of entities (objects) and definition of their properties for a particular application. Concept of class is proposed to classify objects. Operations which can be applied in each object are also defined. Class hierarchy i.e., super class-sub class relationships among classes are to be defined.

2. Schema-modification: It includes:

a. Operations for class hierarchy viz: add a class, delete a class, adding super/sub class relationships etc..

b. Operations for aggregation hierarchy viz: add attribute, change name, add/delete a method or operation etc..

c. Propagation of changes to class definition and object instances

3. Communication between the two layers (underlying GIS and schema): It is established by making use of parser. System manager is used for communication between the two layers. This is described in the next section.

The next section discusses the system design aspects. The, object-oriented modelling capabilities are distributed among the four sub systems viz: schema manager, class manager, object manager and system manager. Approach to problem solving is selected in system design phase.

Responsibilities of four sub systems are described in the subsequent sections. The major responsibilities of schema manager are schema creation for each application, schema listing and view

schema, conflicts resolving etc.. The responsibilities of class manager includes class creation, view class or class list, modifying the class, and propagation changes to object manager, responding messages from schema manager. The responsibilities of object manager are creating objects, modification and retrieval of objects. Transaction includes analysis and query processing the underlying CIS will provide. Map modules consists of specifying boundaries and objects in a map and using analysis functions like add maps, cut or zoom etc. , present in the underlying GIS. The responsibilities of system manager is that it will have overall control and is described in the next section. The importance of parser is described. Object design is the section which is a mapping process i.e., the ideas, strategies and models developed earlier are mapped on to objects, data structures and algorithms. Object diagrams of CIS-schema, CIS-class, CIS-object etc., are discussed.

Finally in the last chapter (chapter-V), conclusions and future directions are explained. The future directions includes:

1. Inclusion of additional features to identify geomorphological features using evidential reasoning like sub-ways, canals etc..
2. In image to map registration problem more image primitives like culverts, railway lines etc., can be added, more attributes like length, region adjacent to which region etc., can be included in image primitives, more topological relations like points inside a region can be added. These will enhance the capability to interpret more features in a better way. Object-Oriented concepts can be extended to knowledgebase.

3. In object-oriented gateway to existing CIS, equivalent object concept i.e., to keep track of different representations of same object can be added, and client-server model which will be beneficial to CIS can be designed with little changes. In client-server model a single server act as kernel which is rich in functionality and will be used by different clients working on client-GISs placed wide apart.

CHAPTER-II

2.0 Reasoning using multiple sources of information:

2.1 Introduction

Most of the CIS applications require the reasoning task to be carried out using multiple sources of information, as they will deal with imprecise, uncertain and inaccurate information. When the information of interest is clear and unambiguous, single line of reasoning can be established and primitives are then instantiated to arrive at solutions, over different situations. In, some application domains, conclusions can be drawn by linking irrelevant and unrelated pieces of information in certain ways. When the information is not clear and ambiguous, it is necessary to connect puzzling features observed on the scene with the reliable information that has been obtained from multiple sources. Thus the knowledge is from multiple sources viz., aerial photographs, geologists knowledge, satellite images, ground observation, is a gallery of frames and compatibility relationships among frames that delimits the space of possibilities. In this process the evidences (multiple sources of information) constitute knowledgebase. Evidential reasoning [28] is to be carried out using various evidential operators (gisting, summarisation etc.) by moving through frames in the gallery by establishing paths converging on spaces where target questions are answered. The knowledgebase represents a store house of the knowledge primitives available to the system which will act as an intelligent tool.

This chapter explains first about spatial reasoning and the earlier work on spatial reasoning viz., Algebraic approach to spatial reasoning, design of fuzzy logic for CIS, intelligent CIS and a logical framework. In subsequent sections the need for evidential reasoning, definitions and terms used in this chapter, the implementation aspects, and a case study by making use of evidential reasoning technique is described.

2.2 Spatial reasoning:

Reasoning procedure [47] which uses spatial and non-spatial attributes, to know all the implicit relationships along with explicit relationships is called as spatial reasoning. For knowing implicit relationships along with explicit relationships, the set of possible spatial objects with their properties and relationships are to be represented in a formal language and inference procedures are used to perform spatial reasoning. Each spatial object is mapped into a distinct node in the network and binary relationships (covers, disjoint, interior etc) among them are represented by the direction and the label of the edges. The explicit relationships are represented and implicit relationships can be derived to have complete knowledge about spatial objects which is of interest to a CIS user.

The information content in the database about spatial objects is extracted for use along with rule based systems for spatial reasoning.

Spatial reasoning [30] is performed for correctly unifying remote sensing information with the existing CIS. Spatial

knowledge is also used to arrive at spatial information such as distance, area, direction and height

2.3 Earlier work on spatial reasoning:

Different strategies/ techniques were followed by various researchers in order to solve tasks involving spatial reasoning, which they encountered in their domain of study. Some of the contributions related to CIS are described below.

2.3.1 Algebraic approach to spatial reasoning [47]:

We assume that we are given a database in which a set of spatial objects and their interrelationships are explicitly represented. We are often faced with a situation in which such a database contain implicit information about many of the objects and spatial relationships that are referred to in the database. Hence we may ask:

1) Whether we may correctly infer all the facts about the objects and their interrelationships that are implicitly represented in the database.

2) whether the current database of objects and relations is consistent and whether it will remain consistent if we add new objects or relationships.

These two questions are closely related and clearly require a reasoning process of some form which is sound and complete. The soundness of a reasoning procedure guarantees that the procedure leads only to correct inference. Hence, for example, any inferred spatial relationships are always logically implied by the initial set of relationships. The completeness of a

reasoning procedure guarantees that the procedure leads to all of the correct inferences. Hence, for example, it will eventually produce all of the correct inferences about relationships that are logically implied by the initial set of relationships.

The main task is to make complete inferences about the set of all spatial relationships that hold between the spatial objects, given prior information about a subset of relationships, constraint satisfaction procedures are used for implementation by formalising the spatial inferences within the framework of relational algebra. The database and knowledgebase systems may be viewed as systems that contain models of some set of phenomena in the real world. So procedures can be developed to access such system by deriving inference about the phenomena represented in the model. In the particular case of Geographical Information System (GIS), such inferential capabilities typically relate to entities that occupy space and change over time, and involve both the spatial and non-spatial attributes and relationships of the entities. The steps involved are representing set of spatial objects, properties of the spatial objects using formal language with well defined syntax and semantics and sets of inference procedures which perform spatial reasoning and these procedures are axiomatized for the domain of spatial objects and relationships.

We assume that a spatial object is defined as some entity possessing an essential projection on to some geometrical space and that this projection may be represented as a point set with a well defined interior and a connected boundary. We also assume

that any spatial relationship of interest between objects may be defined in terms of eight binary topological relationships. We consider only binary spatial relationships as n-ary relations may be equivalently represented in terms of conjunction of binary relationships. The eight relationships are given in the following table.

Relation	Notation	converse
A and B are disjoint	d	self converse
A meets B	m	self converse
A and B overlap	o	self converse
A equals B	e	self converse
A covers B	c	c
A is inside of B	·	i
A is covered by B	c	c
A is properly contains B	⌢	i

We may view each of the eight relations as being atomic i.e each may be viewed as a singleton set containing the smallest non-zero element in the appropriate relation algebra.

Consider a pair of spatial objects and assume that we have reason that the relations holding between these objects i.e in some subset of this atomic relations. We may interpret such a subset as a disjunction of relations, may hold between the pair of objects and as the relationships are disjoint, no more than one relationship in the disjunction will hold.

We use infix notation $O_i(R)O_j$ to represent the fact that the topological relationship R holds between the spatial objects O_i and O_j . The disjunction of the relations R and S over the domain of spatial objects is denoted by $O_i(R+S)O_j = O_iRO_j \ \vee \ O_iSO_j$ for example, $O_1(C+O)O_2$ represents either O_1 covers O_2 or O_1 overlaps O_2 . In general, there are 2^n different disjunctions of eight atomic relationships between a given pair of spatial objects.

We may represent our knowledge about a given set of objects in terms of a graph which we may term as a binary spatial constraint network (BSCN). Each spatial object is mapped into a distinct node in the network and the binary relationships among them are represented by the direction and label of the edges. An edge is directed to distinguish between a binary relation and its converse. The fact that the relations are expressed as a disjunction indicates that our explicit knowledge of the relationships is incomplete. Within this framework, reasoning may be viewed in terms of pruning the set of feasible relations or tightening the constraints on the labels of the under specified edges. We may perform such reasoning if the combined information of the labels of the edges R_{ij} and R_{jk} implies some information about the edge R_{ij} that is not explicitly encoded in the label.

2.3.2 FLESS:

Design of fuzzy logic based expert system for CIS [26]:

Current GISs are predominantly based on boolean logic, a rigid two valued mathematical system which gives no room for

imprecise information. The employment of boolean logic in CIS design causes the following problems.

1) It arbitrarily screens out intrinsic impression in the database and forces artificial precision on imprecise information, graded spatial phenomena and processes. It is especially inadequate in data retrieval and overlay operations. Systems based on boolean logic can only entertain inflexible queries and perform incomplete or untruthful retrieval.

2) It fails to determine and communicate to users the extent of imprecision and error.

3) It is inappropriate to model human cognition, perception and thought processes which are generally embedded with impression. Such a deficiency impedes the implementation of effective and human like inferential procedures in CIS.

4) It is inadequate to model natural languages which are imprecise in nature.

With respect to the level of intelligence, human knowledge and expertise have not been effectively integrated into data input, retrieval, analysis and output in present day CIS. These operations are usually driven by menus or procedures which leave very little room for flexibility and the automation of human intelligence. Thus available CIS suffer from the following acute problems.

1) A low level of automation in the capturing of information from conventional map sources and remote sensing systems.

2) A low level of automated human intelligence in tasks such as feature recognition and recreation.

3) A severely limited ability in formulating and handling complex queries, user interface, data representation and inferences.

4) Poor interfaces with procedural knowledge and rule based knowledge.

Building large ES from scratch for the present purpose is, however costly and time consuming. The availability of flexible and user friendly ES shells is thus important to the construction of intelligent GIS, such that the shell directs the flows of information in and out of GIS and provides inferences. To be able to handle the uncertainty that is due to randomness and imprecision, this shell is also designed to support analyses and inferences in fuzzy environment with some probability arguments.

FLESS facilitates the construction of rule based geographical expert system with intelligence and decision making capabilities.

2.3.3 Intelligent Geographical information system [8]:

In Geographical information systems, new entities or new attributes can be created from existing exact and inexact entities and their attributes in many ways. This process can be generalised as follows. For any given location x , the value of a derived attribute U . can be given by one of the three following functions.

1) When no spatial contiguity is taken into account

$$U_i = F(A, B, C, \dots)$$

2) When spatial contiguity is taken into account

$$U_i(x) = f(A_x, B_x, C_x, \dots)$$

3) When **attribute vary over time**

$$U_i(x,t) = f(A_x, B_x, C_x, \dots)$$

These can again be classified into 9 different classes. Clearly, with such a plethora of options that can be used both simply and in any possible set of combinations, it is not easy for users to learn how and when to use the spatial analysis tools and desired result.

2.3.4 A logical framework [39]:

Spatial analysis systems, no less than any other area, requires representations of knowledge that are complete, correct, flexible and efficient. In pursuit of that goal researchers have exploited a wide variety of knowledge representation schemes including grammar, semantic nets, logics, frames, rules etc. A logical frame work is provided to carry out the knowledge representation and reasoning task.

In order to carry out reasoning task knowledge is represented in first order logic. The representation starts with description of image and specifying it involves considering the image primitives, taxonomy of image objects, relationships among image objects, taxonomy of scene information and general facts about scene objects also have to be represented. Finally the depiction mapping from image to scene which will also involves taxonomic mapping and relational mapping has to be represented. All these representations stated above is called an axiom set, and this set is finally transformed or refined into a set of propositional formulas. Interpretation or reasoning details can

be arrived from the model containing set of axioms by proving the instantiated values to image primitives in the predicate set.

A logical specification of relevant knowledge and underlying assumptions for such an application will consist of image axioms, scene axioms and mapping axioms.

2.4 Need for evidential reasoning:

The various reasoning techniques explained above are useful in situations where reasoning is applied by making use of knowledge which is clear and unambiguous. CIS user will come across situations in which he/she may be interested to identify features like deltaic plains, irregular hills etc., in a target scene under consideration having uncertain or probabilistic information. To solve such problems it is necessary to use multiple evidences [13] (possibilistic information) with basic elements involving tone/colour, texture, shadow etc., with respect to satellite images to be merged with probabilistic or uncertain information of the scene. Once the evidence domain and evidences are known, a gallery is established which consist of frames and relations among frames and evidential reasoning is performed to reveal the information content of a target scene. The target scene may be a topomap or a satellite image.

2.4.1 Architecture and block diagram:

The block diagram of the evidential reasoning system is given below.

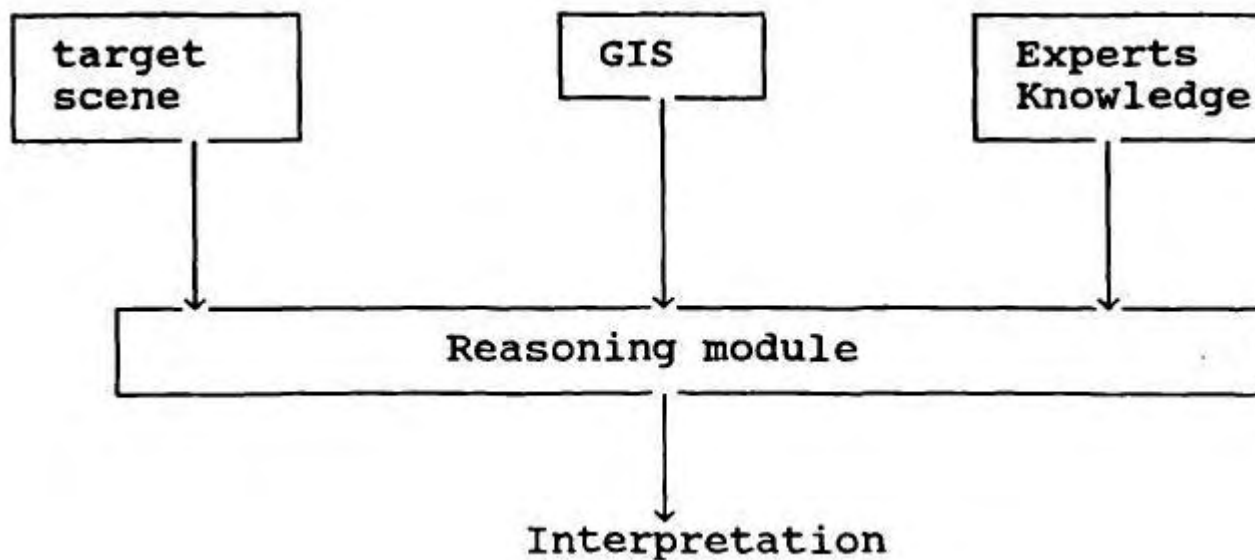


Fig 2.1: Evidential reasoning system

2.5 Theoretical basis [3] [46]: The various terms used in this chapter are given below.

1. Frame of discernment: Suppose Θ denotes a set of possible answers to some question, and assume that only one of these answers can be correct. We call Θ a frame of discernment.

2. Basic probability assignment (bpa): The impact of each distinct piece of evidence on the subsets of Θ is represented by a function called a basic probability assignment (bpa). A bpa is a generalisation of a probability mass distribution, the latter assigns a number in the range $[0,1]$ to every singleton of Θ such that the numbers sum to 1.

3. Belief function [48]: A function Bel that assigns a degree of belief $Bel(A)$ to every subset A of Θ qualifies as a belief function if and only if there is a random non-empty subsets S of Θ

such that $\text{Bel}(A) = \Pr[S \subseteq A]$ or a belief function denoted by Bel , corresponding to a bpa m , assigns to every subset A of \circ , the sum of the beliefs committed exactly to every subset of A by m .

$$\text{Ex: Bel}((a,b)) = m(a,b) + m(a) + m(b)$$

So, $\text{Bel}(A)$ is a measure of the total amount of belief in A and not the amount committed precisely to A by the evidence corresponding to the bpa m .

$$\text{Bel}(A) = m(A) \text{ if } A \text{ is singleton}$$

$$\text{Ex: If } A = \{(a)\}, \text{ then } \text{Bel}(A) = \text{Bel}(\{a\}) = m(\{a\}), \text{ Bel}(\circ) = 1.$$

The information in a belief function Bel can also be expressed in terms of the plausibility function Pl given by

$$\text{Pl}(A) = 1 - \text{Bel}(A) = \sum_{B \subseteq \circ, B \cap A = \emptyset} m(B), \text{ where } A \text{ denotes the complement of } A, \text{ Pl}(A) \text{ is the plausibility of } A \text{ in light of the evidence - a measure of the extent of which the evidence fails to refute } A. \text{ Bel}(A) = 1 - \text{Pl}(A).$$

4. Focal element: A subset S of \circ is called a focal element of Bel if $m(S) > 0$ or $\text{pr}(S=S)$ is positive. The simplest belief function is the belief function whose only focal element is the whole frame \circ , in this case $\text{Pr}(S=\circ) = 1$, and this belief function is called the vacuous belief function. If Bel is vacuous then $\text{Bel} \circ \text{Bel} = \text{Bel}$, for any other belief function Bel . A belief function is called a simple support function if it has at most one focal element not equal to \circ . If a simple support function does have a focal element not equal to \circ (i.e., if it is not vacuous), then this focal element is called the focus of the simple support function. The union of all focal elements of Bel is called the

core of Bel. If all focal elements of Bel are singletons, then Bel is termed as Bayesian [10], The focal elements for $Bel_1 \circledast Bel_2 \circledast \dots \circledast Bel_n$ will consist of all **non-empty** intersections of the form $F_1 \cap \dots \cap F_n$ where F_i is a focal element of Bel_i . The orthogonal sum of simple support functions with a common focus will be another support function with that focus. Similarly, the orthogonal sum of dichotomous belief functions with a common dichotomy will be another dichotomous belief function with that dichotomy. The information in Bel or Pl is also contained in the commonality function Q which is defined as $Q(A) = \sum_{B \subseteq A} (-1)^{|B|+1} Pl(B)$ for every subset A of Ω , $Q(\emptyset) = 1$ and $Pl(\emptyset) = 0$ for any belief function.

2.6 Implementation:

The framework [29] for implementing includes the following steps:

1) Specifying a set of distinct prepositional spaces namely, frame of discernment each of which delimits a set of possible world situation.

2) Arriving at compatibility relation specifying the interrelationships among these set of prepositional spaces.

3) Assigning quantitative belief for the prepositional space.

4) Reasoning by establishing paths for these bodies of evidence to move through these frames by evidential operations, converging on spaces where target questions can be answered. Some initial

implementation was carried-out in [54],

The major steps involved are knowledgebase creation and reasoning.

2.6.1 Knowledgebase:

It consists of a gallery of frames and their compatibility relations among them. This involves delimit a prepositional space of possible situations and one of which is true at any given time. The frame of discernment consists of the set of all features and the task is to identify a single feature which holds good for that particular location. This frame can be denoted by the set $\circ_D = \{ d_1, d_2, \dots, d_{n-1}, d_n \}$ and each element d_i corresponds to a particular feature of a scene under consideration.

Examples of frames are Area (upland area, plain land, piedmont zone), Tone (red, yellow, ash, grassy-green) etc..

Example of a compatibility relation Area-features ((upland area, mountains), (upland area, hills)).

Prepositional statements can be represented by disjunctions of elements from the frame. Frames can be represented as graphs consisting of nodes connected by direct edges. For each element a node is included. Nodes have edges pointing to other elements of other frame. For example D_i might correspond to the statement, the area is plain in such case D_i would be represented by the subset of elements from \circ_D that represents different types of plain.

$$D_i \subseteq \circ_D$$

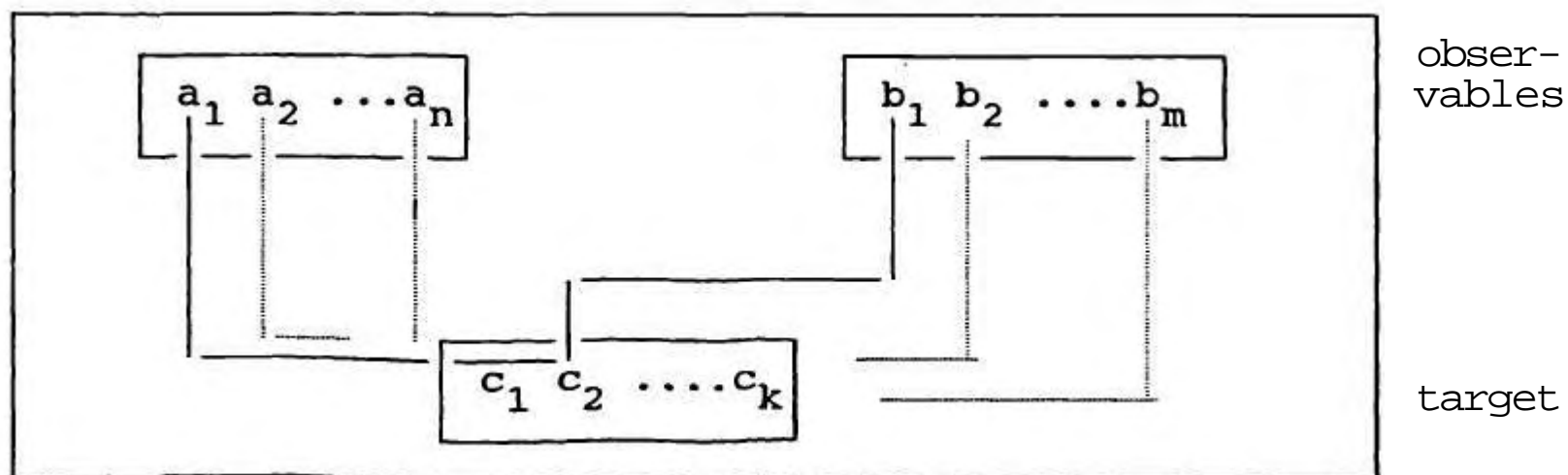
Belief values are assigned [48] to the individual propositions in space. Additional frames of discernment may be defined. In order to identify the drainage pattern present in the given area, a frame of discernment can be established which represents different drainage patterns ω_E . Using compatibility relations the frames can then be related. This relation includes the elements from the related frames which can be true simultaneously,

$$D, E, \dots, N-1, N \quad D^x \circ E^x \dots x \circ N$$

Using this compatibility relation $\omega_{D, E, \dots, N}$ a compatibility mapping $C_{D \rightarrow E}$ can be defined for translating statements [32] expressed relative to frame ω_D to the statements relative to ω_E .

A compatibility relation is represented as a graph that includes the nodes from the frames that it relates with connecting to compatible elements. Directed edges define the compatibility mapping from one frame to another frame moving forward along the edges. Image understanding system is made use of for analysing the image in order to arrive at belief values to each of the observable features.

GALLERY



(a₁ b₁ c₂) are compatible

Fig 2.2: Knowledge representation

A good knowledge representation scheme must be flexible to include or modify features, serve as a tool with which a space of alternative formulations can easily be explored. Once a modification is complete the argument should automatically react to the changes, updating any conclusions that depend upon it.

A common frame can also be established that captures all the information relating to translation of statements between various sets of frames.

As more aspects of interest are added, the number and complexity of frames and compatibility mappings increases. One can establish a single complex frame that includes all aspects of interest or establish a network of frames that includes a distinct frame for each aspect of interest. Deriving conclusions using network of interrelated frames and single complex frame may not be same. However reasoning based on a well formed inter connected frames constituting a gallery is sound.

The steps involved in creation, modification etc., of knowledgebase or gallery are given below:

1. Create a gallery: Gallery with different frames having elements and their compatibility relations among them is to be created first. For each frame of discernment, a database of its elements and a set representation is created, so that given an element of a frame, its set representation can be known. Set union and set intersection can be easily implemented.

2. Update a gallery: Frames in the gallery, elements of a frame of discernment, the compatibility relations of the frames in the

gallery can be updated whenever any change take place.

3. Work as a pre-defined gallery: Once the user establishes a gallery, he need not create it every time instead he can use the appropriate gallery file already created, from the database.

4. Delete a gallery: Gallery can be deleted whenever such need arises. It will automatically delete the associated frames, compatibility relations and all related files.

2.6.2 Reasoning for interpreting a target scene:

The aim is to establish a line of reasoning based upon both the possibilistic information from evidence domain or multiple sources of information in the gallery and probabilistic information from the target scene for identifying the likely features on the scene. The gallery delimits the space of possible situations and this evidential information establishes the likelihood of these possibilities. Bodies of evidence (multiple sources of information) are expressed relative to frames in the gallery and compatibility mappings are used for establishing paths to move through frames where target questions can be answered, for identifying features on the scene.

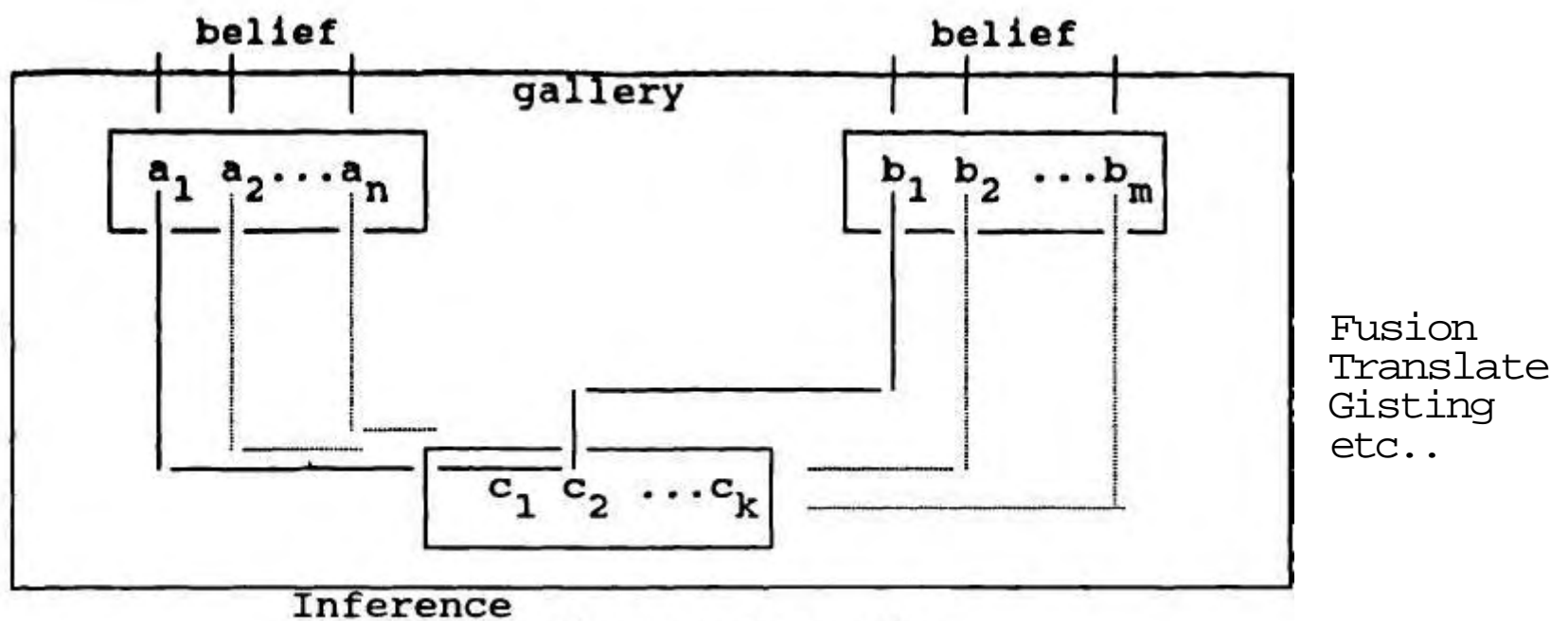


Fig 2.3 Reasoning

Each body of evidence is expressed in terms of a frame in the gallery and is represented as a mass distribution (eg: m_A) over propositional statements discerned by a frame (eg: ϕ_A)

$$m_A = 2^{\phi_A} \rightarrow [0,1] \text{ such that}$$

$$\sum_{A_i \subseteq \phi_A} m(A_i) = 1 \text{ and}$$

$$m_A(\phi) = 0$$

Intuitively, mass is attributed to the most precise propositions a body of evidence supports.

The basic elements which aid in the identification of different features in satellite image [51] are given here under.

a. Tone/colour: Tone/colour is used by the expert to identify the vegetative cover, snow, clouds, water bodies, soils etc.. Evidence relating to various features can be computed using the frame tone and the compatibility relation between the frames tone and features .

b. Textures Texture is created by tonal repetition in groups of objects that are often too small to be discerned as individual objects. Texture is the usual impression of roughness or smoothness, is a valuable clue in interpretation. Mostly, the plains have a smooth texture. The given area should be processed to compute the roughness of the area.

c. Shadow: Shadow provide information not apparent from other sources and are particularly helpful for objects which are very small or lack of tonal contrast with their surroundings. Shadows usually indicate steeper tones and hence, evidence that the area is an upland can be supported.

d. Context: In the land form analysis, context is very important. We can identify a plain as a coastal plain, if it is bordering a sea, and as a playa if it is surrounded by uplands.

e. Pattern: Pattern or repetition is characteristic of many man made/ natural objects. Pattern or spatial arrangement of objects is important clue to their origin or function or both. Drainage pattern gives some evidence supporting certain features.,

f. Shape: The shape or form of some objects is so distinctive that they can be identified solely based on this parameter.

g. Size (area): The size of an object is one of the most useful clues to its identity. A location is an isolated hill if its aerial dimension is less and is present on an extensive plain.

h. Association: This clue is most useful or helpful in identification of land forms. A flood plain is associated with several features such as river terraces, back swamps, natural levees, abandoned channels etc.. A sandy plain in a desert is

associated with several types of sand dunes such as parabolic, longitudinal dunes, barchams etc..

Different types of evidence domain and evidences are to be selected such that reasoning can be applied to specific types of problems. Following options are available for an evidence domain.

Option 1. All: i.e. the domain can be any subset of the frame of discernment \circ

Option 2. Singleton elements: i.e. the domain is the individual prepositions of a frame of discernment.

Option 3. Hierarchical hypotheses space: only some sub-sets are considered, which form a strict hierarchy.

Option 4. Partition: The evidence domain is the field p^x where P is a partition and P^* is the set of all unions of elements of P .

Appropriate evidence domain for a frame of discernment in the gallery, and the type of evidences used are selected to arrive at solutions. Following types of evidences can be selected.

1. All: The evidence is carried by the power set of a frame of discernment, \circ .

2. Simple support functions: A belief function is called a simple support function if it has at most one focal element not equal to whole frame, \circ .

3. Dichotomous belief function: A belief function is called dichotomous belief function with dichotomy $\{A, A^c\}$ if it has no focal elements other than A, A^c and \circ .

Evidences are analysed using different evidential operators converging them on spaces related to which the target questions can be answered.

2.6.3 Operations on evidences:

Various operations on evidence domain and evidences are described below.

2.6.3.1 Fusion: This operation is used to determine consensus (single body of evidence) from several bodies of evidence obtained from independent sources. Emphasis is on points of agreement and deemphasises on points of disagreement. Once multiple bodies of evidence representing independent opinions are expressed relative to the same frame of discernment, they can be fused or combined [45] to represent the consensus of the original disparate opinions. Select different combination rules based upon the type of evidence domain and the type of evidences selected relative to a frame of discernment. Following rules can be applied for this purpose depending on evidence domain and type of evidences.

a. Dempster's rule [50]: Dempster's rule [DS rule] of combination can be used when the evidence domain is the power set of Θ and the evidence is also carried by the power set of Θ . It is defined as follows:

$$m_1 \oplus m_2 (c) = k^{-1} \left(\bigwedge_{A_i \cap B_j = c} m_1(A_i) m_2(B_j) \right) \text{ if } c \neq \emptyset; m_1 \oplus m_2 (\emptyset) = 0$$

$$\text{where } k = \sum_{A_i \cap B_j \neq \emptyset} m_1(A_i) m_2(B_j)$$

Dempster's rule both commutative and associative and therefore multiple bodies of evidence can be combined in any order with out affecting the result. This rule is based on the conjunction of evidences.

b. Optimistic rule of combination [27]: When the evidence domain and the evidence are carried by the power set of Ω , optimistic rule of combination is selected which is based on the disjunction of evidence and provides a good approximation for combining bodies of evidence which disagree with each other or contradict with each other. The optimistic rule of combination is defined as follows:

$$m(A) = K^{-1} \left[\sum_{F_i \in \Omega_i} m_1(F_i) + \sum_{G_j \in \Omega_i} m_2(G_j) - \sum_{\substack{F_i \cap G_j \neq \emptyset \\ F_i, G_j \in \Omega_i}} m_1(F_i) m_2(G_j) \right]$$

$$K = \sum_{F_i \cap G_j \neq \emptyset} m_1(F_i) m_2(G_j)$$

Ω_i = The union of the elements in the maximal set of interacting elements.

F_1, F_2, \dots, F_p are focal elements of the mass distribution m_1 , and G_1, G_2, \dots, G_q are the focal elements of the mass distribution m_2 . Ω_i is maximal set of interacting elements i.e. every element of this set interacts with every other element directly or indirectly. 'A' is a focal element of a resulting mass distribution which is defined as:

$$A = \left[\bigcup_{F_i \in \Omega_i} F_i \right] \cup \left[\bigcup_{G_j \in \Omega_i} G_j \right]$$

c. Bayesian approximation [25] [50]: When the evidence domain and evidence carried by the power set Ω , one can compute the bayesian approximation Bel of the original belief function Bel for each body of evidence and then combine these bayesian approximations using Dempster's rule of combination.

The bayesian approximation Bel of Bel is induced by the basic

probability assignment (bpa) \underline{m} defined by

$$\underline{m}(A) = \sum_{A \subseteq B} m(B) / \sum_{C \subseteq \Theta} m(C) |C|, \text{ if } A \text{ is a singleton}$$

$$m(A) = 0 \text{ otherwise}$$

The factor $[\sum_{C \subseteq \Theta} m(C) |C|]^{-1}$ is called the bayesian constant of Bel which represents measure of precision.

Since the bayesian approximation Bel is bayesian, the combination of bayesian approximations of belief functions is computationally less involving than combining the belief functions themselves using DS rule.

d. Barnett's technique for singleton hypotheses [43]: When the evidence domain is singleton elements and their negations, and the evidence being combined are simple support functions, focused on singletons or their negations Barnett's formula can be used to combine the belief functions for the different elements of the frame of discernment.

Step 1: For each of $e \in \Theta$, Bel_1 is a simple support function focused on the singleton $\{e\}$ and Bel_2 is a simple support function focused on its complement $\{e\}^c$. Bel_1 and Bel_2 are then combined using Dempster's rule to obtain dichotomous belief function Bel ($Bel_1 \oplus Bel_2$) focused on $\{e\}, \{e\}^c$.

Step 2: For each $e \in \Theta$, $Bel_{e \in \Theta}$ can be computed using the following formula.

$$Pl(A) = k \left(1 + \sum_{e \in A} e^+ / (1 - e^-) - \prod_{e \in A} e^- / (1 - e^+) \right) \text{ where}$$

$$k^{-1} = 1 + \prod_{e \in \Theta} e^+ / (1 - e^+) - \prod_{e \in \Theta} e^- / (1 - e^-)$$

$$\text{and } e^+ = Bel_e \{e\}, e^- = Bel_{e^c} \{e\}$$

The number of computations required by σ increases only linearly with the number of elements in σ .

e. Barnett's technique applied to partition [43]: A partition of a frame of discernment σ is a set of disjoint nonempty subsets of σ whose union equals σ . p^* is used to denote the set consisting of all unions of elements of p : p^* is the field of subsets of σ . When the evidence domain is the partition p and the evidence is the simple support functions for and against elements of p , Barnett's technique can be applied to combine the belief functions. $\sigma(\text{Bel} \mid p \in p)$, Where Bel^p is dichotomous with dichotomy $\{p, p^c\}$.

As before the simple support functions for and against elements of partition p can be combined to obtain the dichotomous belief function Bel .

$$Pl(A) = k \left(1 + \sum_{\substack{p \subseteq A \\ p \in p}} p^+ / (1-p^+) - \prod_{\substack{p \in p \\ p \not\subseteq A}} p^- / (1 - p^+) \right)$$

for every element A of p^* where

$$k^{-1} = 1 + \sum_{p \in p} p^+ / (1-p^+) - \prod_{p \in p} p^- / (1-p^+)$$

where $p^+ = \text{Bel}_p(p)$, $p^- = \text{Bel}_p(p^c)$

where Bel is carried by p , its value for elements of p determine its values for the other subsets of σ .

$$\text{Bel}(A) = \max \{ \text{Bel}(B) \mid B \subseteq A, B \in p^* \}$$

$$Pl(A) = \min \{ Pl(B) \mid B \supseteq A, B \in p^* \}$$

f. Gordon and Shortliffe algorithm [15] [34] [44]: When the evidence domain is a hierarchical hypothesis space and the evidence to be combined are simple support functions for and against hypotheses that can be arranged in a tree like structure.

Gordon and Shortliffe's technique can be applied.

The algorithm and the corresponding formulas are as follows:

Let T denote the set of subsets (except for \emptyset itself) in the hierarchy of hypotheses and T' denote the set of all complements of subsets in T (this is also a subset of 2^{Ω} but entities in T' will not be generally in T i.e. negation of hypotheses).

Step 1: For each subset X_i in T combine all confirmatory evidence to obtain a bpa m_{X_i} using the formula

$$m_{X_i}(x_i) = (1-s_1) (1-s_2) \dots (1-s_k)$$

Where s_1, s_2, \dots, s_k are the simple support functions confirming x_i . Combine all **disconfirmatory** evidence to obtain another bpa $m_{\bar{x}_j}$ (or $m_{\bar{x}_i}$) using the above formula for confirmatory evidence except that now s_1, s_2, \dots, s_k are simple support functions representing the evidence **disconfirming** x_i .

Both m_{x_i} and $m_{\bar{x}_j}$ (or $m_{\bar{x}_i}$) are simple support functions focused on x_i and \bar{x}_i respectively.

Step 2: It is required to compute

$$m_T = m_{x_1} \circledast m_{x_2} \circledast \dots, x_i \in T, \text{ and } m_{T'} = m_{\bar{x}_1} \circledast m_{\bar{x}_2} \circledast \dots, x_i \in T'$$

The values of $m_m(A)$ for A can be obtained using the following formulas $m_T(A) = k \prod_{x_i \in T} m_{x_i}(A)$ if $A \in T$, $m_T(A) = k \prod_{x_i \in T} m_{x_i}(\emptyset)$ if $A = \emptyset$

Where $k = \frac{1}{(1-k')}$, $1-k' = \prod_{x_i \in T} m_{x_i}(\emptyset)$.

Step 3: Combine disconfirmatory evidence by step-wise combination of the $m_{\bar{x}_i}$'s in T' and calculate $m_T \circledast m_{\bar{x}_1}$ then $(m_T \circledast m_{\bar{x}_1}) \circledast m_{\bar{x}_2}$ etc., for all x_i in T' in such a way that $(m_T \circledast m_{\bar{x}_i})$ is an

approximation to $(m_A \ominus m_{\bar{A}})_i$. Belief assigned to a subset A by e is instead assigned by \odot to the smallest super set of A in T if A itself is not in T.

There are different formulae in step 3 depending upon which of the three relationships hold between X and A: $x \in A$, $x \cap A = \phi$ or $x \supset A$, where X is a subset of $T \cup \emptyset$ and A is a subset of T.

In all cases $k = 1 / (1 - k')$, where $k' = \frac{m_{\bar{x}}(A) \wedge m_m(x)}{\sum_{x \in T} m_{\bar{x}}(A)}$.

Case 1; $x \subseteq A$:

$$m_T \ominus m_{\bar{A}}(x) = k m_T(x) m_{\bar{A}}(\emptyset)$$

Case 2: $x \cap A = \phi$ { i.e., $x \cap A = x$ }:

i. If $x \cup A$ is a set in $T \cup \emptyset$:

$$m_T \ominus m_{\bar{x}}(x) = k (m_T(x) + m_T(x \cup A) m_{\bar{A}}(A))$$

ii. If $x \cup A$ is not in $T \cup \emptyset$:

$$m_T \ominus m_{\bar{x}}(x) = k m_T(x)$$

Case 3: $x \supset A$:

i. If $x \cap A$ is not a set in T:

$$m_T \ominus m_{\bar{x}}(x) = k m_T(x)$$

ii. If $x \cap A$ is in T:

$$m_T \ominus m_{\bar{A}}(x) = k m_T(x) m_{\bar{A}}(\emptyset).$$

Gordon and Shortliffe claim that combining evidence in a breadth first fashion, from higher to lower levels will result in an optimal approximation.

The algorithm computes belief to only subsets in T. Thus for A in T, Bel(A) can be computed by summing net belief in A with belief assigned to all its descendants. However, it will not in general be possible to compute Bel(A), since A will usually be in T' but not in T. Thus the notion of the belief interval

$(\text{Bel}(A), 1-\text{Bel}(A))$ is lost in this scheme. Competing hypotheses would need to be compared based upon Bel alone with out regard to the width of the belief interval,

g. Shafer and Logan's algorithm for hierarchical evidence [43][44]:

When the evidence domain can be reduced to hierarchical hypotheses space and the evidence being combined are simple support functions with evidence for and against the hypotheses in the tree, Shafer and Logan's algorithm can be used to combine the belief functions which is linear in its computational complexity. It computes belief intervals for the hypotheses in the tree. The algorithm is as follows.

Algorithm:

Let A denote the set of all nodes in the tree except e . Let Bel_A for each node A in A be the single dichotomous belief function with dichotomy $\{A, \bar{A}\}$.

For any node A in the **tree**, Bel_A^\downarrow is the orthogonal sum of Bel_B for all nodes B in A that are strictly below A .

For each node A in Bel_A denotes the orthogonal sum of Bel_B for all nodes B in A that are neither below A nor equal to A .

sA denotes the set of all daughters of a nonterminal node A .

For each node A in A , we set

$$\begin{aligned} A_0 &\approx \text{Bel}_A(A), A_0^- = \text{Bel}_A(\bar{A}) \\ A_\downarrow^+ &\approx \text{Bel}_A^\downarrow(A), A_\downarrow^- = \text{Bel}_A^\downarrow(\bar{A}) \\ A &= (\text{Bel}_A \oplus \text{Bel}_A) (A) \quad A^- = (\text{Bel}_A \oplus \text{Bel}_A) (\bar{A}) \\ A_\Delta &\approx (\text{Bel}_A \odot \text{Bel}_A^\Delta) (A), \quad A_\Delta^- = (\text{Bel}_A \odot \text{Bel}_A^\Delta) (\bar{A}) \end{aligned}$$

$$A_i^+ = \text{Bel}_i^\downarrow(A), \quad A_i^- = \text{Bel}_i^\downarrow(A)$$

If A is a terminal **node**, then Bel_i^\downarrow is **vacuous**, and therefore

$$A_i^\downarrow = A_i^- = 0, \quad A_i^+ = A_i^- \quad \text{and} \quad A_i^+ = A_i^-.$$

For each node B other than e and its daughters, we set

$$B_i^+ = \text{Bel}_i^\downarrow(B), \quad B_i^- = \text{Bel}_i^*(B)$$

$$B_{A_i}^* = \text{Bel}_{A_i}^\downarrow(B \cup A) \quad \text{where } A \text{ is } B \text{'s mother,}$$

Step 1: To compute $(\text{Bel}_{A_i}^\downarrow)(A, A)$ for mothers of sibs of terminal nodes. This can be done using the following formula:

To calculate A_i^\downarrow and A_i^\sim from B and B for B in S

$$A_i^+ = 1 - k$$

$$A_i^\sim = k \pi_A B^\sim / (1 - B^+) \quad \text{where} \quad k^{-1} = 1 + \sum_{B \in S_A} B^+ / (1 - B^+)$$

This is followed by the calculation of $\text{Bel}_{A_i}^\circ(\text{Bel}_{A_i}^\downarrow)(A, A)$ by the following formulas:

$$A_i^+ = 1 - k (1 - A_0^\sim) \cdot (1 - A_i^+)$$

$$A_i^\sim = 1 - k (1 - A_0^+) (1 - A_i^\sim) \quad \text{where} \quad k^{-1} = 1 - A_0^+ A_i^\sim - A_0^\sim A_i^+$$

After these operations have been completed for every node A, whose daughters are all terminal nodes, we pretend to prune all these terminal nodes from the tree and we repeat the process with the new sibs (sets of all daughters) of terminal nodes and so on, until we calculate $(\text{Bel}_{A_i}^\downarrow)(A, A)$ for the daughters A of e.

Step 2: We calculate $\text{Bel}_e^\downarrow(A)$ and $\text{Bel}_e^\circ(A)$ for each A in S_e using the following formulas:

$$A_e^+ = 1 - k \left(1 + \sum_{\substack{B \in S_e \\ B \neq A}} B^+ / (1 - B^+) - \pi_e B / (1 - B^+) \right)$$

$$A_e^- = 1 - k (1 - A_e^\sim) / (1 - A_e^+)$$

where $k^{-1} = 1 + \sum_{B \in S_e} B^+ / (1 - B^+) - \sum_{B \in S_e} B^- / (1 - B^-)$

Step 3: We go back down the tree, when we go from A to its daughters, we find $Bel_{\Delta}^{\Delta}(A, A)$ using the following formulas:

$$A_{\Delta}^+ = 1 - k (1 - A_{\Delta}^+) / (1 - A_{\Delta}^+)$$

$$A_{\Delta}^- = 1 - k (1 - A_{\Delta}^-) / (1 - A_{\Delta}^-)$$

Where $k^{-1} = (1 - A_{\Delta}^+) / (1 - A_{\Delta}^+) + (1 - A_{\Delta}^-) / (1 - A_{\Delta}^-) - (1 - A_{\Delta}^+ - A_{\Delta}^-) / (1 - A_{\Delta}^+ - A_{\Delta}^-)$

Then we calculate $Bel_{\Delta}^{\Delta}(B)$, $Bel_{\Delta}^{\Delta}(\bar{B})$ and $Bel_{\Delta}^{\Delta}(B \cup \bar{A})$ for each B in S_A using the following formulas:

$$B_{\Delta}^+ = k (A_{\Delta}^+ (B^+ - A_{\Delta}^+) + (1 - A_{\Delta}^+ - A_{\Delta}^-) B_{\Delta}^+)$$

$$B_{\Delta}^- = 1 - k (1 - A_{\Delta}^-) (1 - B_{\Delta}^-) \text{ where } k^{-1} = 1 - A_{\Delta}^- / A_{\Delta}^+ - A_{\Delta}^- / A_{\Delta}^+$$

2.6.3.2 Discounting: It is an evidential operation that adjusts a mass distribution to reflect its source's credibility (expressed as a discount rate $r \in (0,1)$). If a source is completely reliable ($r=0$) discounting strips away all apparent information content i.e discounting has no effect, if a source is a unreliable ($r=1$), discounting strips away all apparent information content; otherwise discounting lowers the apparent information content in proportion to the source's unreliability. It has the effect of widening the evidential intervals, reflecting increased ignorance. Discounting is defined as follows:

$$m_{\Delta}^{\Delta}(A_i) = (1 - r) m_A(A_i), \quad A_i \neq \emptyset_A \\ = r + (1-r) m(\emptyset) \text{ otherwise}$$

2.6.3.3 Translation: Using this evidential operation a body of evidence can be moved away from its original context to a related one, to assess its impact on dependent hypotheses. If a body is to be interpreted to a question expressed over a frame different from the one over which the evidence is expressed, a path of compatibility relations connecting the two frames is required. The mass distribution expressing the body of evidence is then repeatedly translated frame to frame, by way of compatibility mappings until it reaches the ultimate frame of question. In translating m_A from frame \circ_A to frame \circ_B by way of compatibility mapping $C_{A \rightarrow B}$, the following computation is applied to derive the translated mass distribution m_{\circ_B}

$$m_{\circ_B}(B_j) = \sum_{\substack{C_{A \rightarrow B}(A_i) = B_j \\ A_i \subseteq \circ_A \vee B_j \subseteq \circ_B}} m_{\circ_A}(A_i)$$

where $C_{A \rightarrow B}(A_i) = \{b_j \mid (a_i, b_j) \in \circ_{A,B}, a_i \in A_i\}$

Intuitively, if we (partially) believe A. and A. implies B., then we should have the same (partial) belief in B.. This method when applied to move mass distribution among frames that represent states of the world at different times is called projection. Projection operation is used to move a body of evidence away from its original temporal context, to a related one.

2.6.3.4 Summarisation: Summarisation eliminates extraneous details from a body of evidence or mass distribution by collecting all the extremely small amounts of mass (determined by a threshold $t \in$

[0,1]), attributed to propositions and attributing the sum to the disjunction of those propositions. The resulting body of evidence or mass distribution is slightly less informative than the original, but it remains consistent with the original.

$$m(A_i) - m(A_i), A_i * S \\ = S + m(S), \text{ otherwise}$$

$$S = \cup A_i \\ 0 \neq m(A_i) < t$$

$$S = \sum m(A_i) \\ 0 \neq m(A_i) < t$$

2.6.3.5 Gisting: General sense of a body of evidence can be derived using this operation which produces a single statement that captures the general sense of a body of evidence, without reporting uncertainty. Gisting produces a (boolean valued) statement that attempts to capture the essence of mass distribution. In other words, it attempts to summarise the contents of a body of evidence in terms of a single statement from the frame, void of any uncertainty or ignorance. The gist of a mass distribution is the most pointed statement from the frame whose support meets or exceeds a selected level. The gist G of a mass distribution, m_A is defined relative to a gist level $g \in (0,1)$

$$G = \bigcup_{A_i \in \mathcal{A}} A_i, \quad G \in \mathcal{A} \text{ for all } A_i \in G, A_k \notin G$$

$$\text{spt}(A_i) = \text{spt}(A_j) > g$$

$$|A_i| = |A_j|$$

$$\text{Spt } (A_i) > \text{Spt } (A_k) \text{ or } |A_k| > |A_i|$$

2.6.3.6 Interpretation: The truth of a given statement, based upon a given body of evidence can be determined using this evidential operation. The positive and negative effects of the truth of the given statement using the body of evidence are produced during the process. To interpret a body of evidence relative to a frame Θ_A , we calculate belief intervals for the various subsets of Θ_A . The belief interval for a proposition A_j is computed as follows:

$$\text{Bel } (A_j) = \sum_{A \subseteq A_j} m_A [A_i], \text{ pl } (A_j) = 1 - \text{Bel } (\Theta_A - A_j)$$

$$[\text{Bel } (A_j), \text{Pl } (A_j)] \subseteq [0, 1]$$

The lower bound of an evidential interval indicates the degree to which the evidence supports the proposition, while the upper bound indicates the degree to which, the evidence fails to refute the proposition i.e the degree to which it remains plausible.

Although evidence may point to subsets of the frame of discernment Θ without pointing to any particular element, one is often, interested in final conclusions about the elements of Θ . When belief intervals of elements of Θ are given there is no unique, way to order them with respect to their degree of certainty. There are different orderings [50] possible, on these belief intervals.

1. minimal ordering \leq_{\min} is defined by

$$[x, y] \leq_{\min} [x', y'] \text{ iff } y \leq x'$$

Let $c, d \in \Theta$, we write $c \leq_{\min}^d$ for $\text{Bel}(c),$

$\text{pl}(c) \leq_{\min} [\text{Bel}(d), \text{pl}(d)]$ and $c \leq_{\min}^d$ for $c \leq_{\min}^d \wedge d \leq_{\min}^c$.

2. average ordering \leq_{av} is defined by

$$[x, y] \leq_{av} [x', y'] \text{ iff } (x+y)/2 \leq (x'+y')/2.$$

Let $c, d \in \mathcal{O}$ we write $c \leq_{av} d$ for $[\text{Bel}(c), \text{pl}(c)] \leq_{av} [\text{Bel}(d), \text{pl}(d)]$

and $c =_{av} d$ for $c \leq_{av} d \wedge d \leq_{av} c$.

3. plausible ordering \leq_{pl} is defined by

$$[x, y] \leq_{pl} [x', y'] \text{ iff } y < y'.$$

Let $c, d \in \mathcal{O}$ we write $c \leq_{pl} d$ iff $\text{pl}(c) \leq \text{pl}(d)$ or $\underline{m}(c) \leq \underline{m}(d)$

4. belief ordering \leq_{Bel} is defined by

$$[x, y] \leq_{Bel} [x', y'] \text{ iff } x < x'.$$

Let $c, d \in \mathcal{O}$, we write $c \leq_{Bel} d$ iff $\text{Bel}(c) \leq \text{Bel}(d)$

Reasoning using multiple sources (evidential reasoning) has been successfully applied to problems in computer vision, intelligence analysis etc., with different options for evidence domain and evidence available and this reasoning system provides a good framework for automated reasoning for a variety of problems.

2.7 Case study of evidential reasoning using expert's knowledge:

In order to identify geomorphological features in a target scene (topographic, satellite image) using evidential reasoning, knowledge of geologists who are experts in this field is used to build gallery or knowledgebase. This gallery consists of knowledge from multiple sources and it contains frames and compatibility relations among them. This gallery will reveal the information content of the target scene. Certain types of features whose presence or absence support the presence of certain geomorphological features. The gallery is organised properly to represent various frames and is used to know exactly the features present on the satellite image or topographic map. This methodology is described in the following sections.

2.8 Geomorphological features to be identified:

Mountains = h1

Inselbergs/Bornhardts = h2

Irregular hills = h3

Domal hills = h4

Plateau = h5

Mesa = h6

Butte = h7

Asymmetrical ridges = h8

Symmetrical ridges = h9

Deltaic plains = h10

Pedi plains = h11

Planar surfaces = h12

Marine platforms - h13
River terraces - h14
Rolling plains - h15
Piedmont plains = h16
Coastal plains = h17
Terrace plains = h18
Outwash plains = h19
Sandy plains = h20
Playa = h21
Alluvial plains = h22
Sea = h23
River = h24
Paleoriver course = h25
Alluvium = h26
Uplands = (h1, h2-h4, h5-h7, h8-h9)
Plains = (h10-h22)
Hills = (h2-h4)
Plateau lands = (h5-h7)
Ridges = (h8-h9)
Sea = h23
River = h24

Once the features to be identified are determined information is collected about the process of manual image interpretation. In other words we attempted to understand the manual process in which the geologists identify the above set of features from a satellite image. Certain specific types of features whose presence or absence in map/image supports the presence of certain

geomorphological feature. The information as collected from experts is summarised below.

In the first list the features in the toposheets and the corresponding hypotheses that are being supported are as follows:

1) Closely spaced contour lines (h1-h9)

a) 1) spreading over large area (h1)

2) closely spaced in some areas and around these areas they are widely spaced (h1)

3) widely spaced contour lines suddenly group together indicating foothills (h1)

b) contour lines form concentric ellipses (h8,h9)

1) closely spaced concentric ellipses are close together on one side and farther apart on the other side (h8)

c) contour lines are circular (h2,h4)

d) contour lines are oval shaped (h3)

e) closely spaced contour lines around an area where there are no contour lines at all (h5,h6,h7)

1) area of the region with no contour line is less than 4 sq cm on 250000:1 topo sheet (h7)

2) area of contourless region is between 4 to 16 sq cm (h6)

2) Widely spaced contour lines (h10-h22)

3) Thin drainage lines join up to form thicker lines (h1-h9)

1) radial type of drainage pattern (h2,h5,h6,h7)

2) annular drainage pattern (h4)

4) Very few drainage patterns (h10-h22)

1) shallow drainage channels (h11)

2) drainage dendritic (h15)

- 3) drainage rectangular (h15)
- 4) drainage barbed (h15)
- 5) dichotonous drainage pattern (h16)
- 5) No drainage patterns (h10-h22)

In the second list the features from satellite images and the correspondences that are being supported are as follows:

Features from FCC image:

- 1) Tone is light grey (h1-h9, h11)
- 2) Tone is dark grey (h1-h9, h15, h17)
- 3) Tone is light brown (h1-h9, h12, h14, h15)
- 4) Tone is dark brown (h1-h9)
- 5) Bright to red mottled tone (h10, h22)
- 6) Tone is bluish colour (h11)
- 7) Tone is light to dark or angish shade (h11)
- 8) Tone is grassy green (h11, h12)
- 9) Dark green (h11)
- 10) Thick lash grey (h13)
- 11) Medium grey (h15, h17)
- 12) Snowy white patches (h19)
- 13) Muddy colour (h18)
- 14) Big black patch (h23)
- 15) Red colour (h16, h22)
- 16) Yellow colour (h15, h21, h22)
- 16)a) Very light yellow (h21)
- 16)b) Light yellow (h15)
- 17) Light blue (h22)
- 18) White (h17)

19) Whitish grey (h16)

20) FCC image shows up slender black/blue/white coloured lines which are braiding/meandering/interlaced (h24)

21) FCC image shows some very bright spots indicating salt deposits (h20)

22) Tone darker than that of the surrounding area (h22)

Band 4 image:

1) Uniform tone and texture (h10-h22)

2) Grey tone (h2, h12-h14, h16-h19)

2) a) Grey tone light grey (h19, h16, h2, h12)

2) b) Grey tone medium grey (h2, h14, h16, h17, h18)

2) c) Grey tone dark grey (h12-h14, h17, h19)

3) Whitish (h17)

4) Tone very bright (h20, h21)

Some more evidences supporting geomorphological features:

1) Prominent crests or peaks (h2-h4)

a) Crest/peak is off center (h3)

2) Broad convex form with smooth textured surface (h4)

3) Located between uplands and rivers (h10-h22)

4) Presence of rock out crops or close to uplands (h11)

5) Level summits in extensive upland areas (h12)

6) Rocky sloping surfaces in sea or close to it (h13)

7) Bench like/stepped structures beside river (h1-h9)

8) Number of ponds/tanks across (h15)

9) Fan like features at base of uplands (h16)

10) Near to sea/lake (h10)

11) Triangular/fan like in shape (h10)

- 12) Hashed lines/shading in toposheet (h20)
- 13) Surrounded by uplands (h21)
- 14) Apron like or triangular in shape (h19)
- 15) Stepped area beside a river (h18)
- 16) Associated with elongated water bodies coming into the land from sea/lake (h17)

The information collected from the experts is essentially unstructured and hence not suitable for implementation using computer. The following is the list of knowledge as acquired from the experts is converted suitably in the form of gallery as discussed earlier. The structured knowledgebase is given below.

2.9 Knowledgebase for the identification of Geomorphological features from satellite images and topomaps:

2.9.1 Topomaps:

A) Drainage pattern

The phrase x(y) indicates y if x is present

- 1) Dichotomous drainage pattern (piedmont plain)
- 2) Annular drainage pattern (domal hills)
- 3) Radial type of drainage pattern (hills, inselbergs, bonhardts, domal hills, plateaus)
- 4) Centripetal drainage pattern (playa)
- 5) Reticulate drainage pattern (Manshy tidal flats)->(deltaic plain)
- 6) Dendritic drainage pattern (fluvial plain, pedi plain)
- 7) Subparallel drainage pattern (piedmont plain)

B) Contour lines

- 1) Very closely spaced contour lines(steeper slopes)->(upland area)

- 2) Moderately spaced contour lines (piedmont zone)
- 3) Very widely spaced contour lines (plain land)
- 4) Oval/circular contour lines in toposheet (domal hills, bornhardts)
- 5) Contour lines from closely spaced concentric ellipses (ridges)
- 6) Contour lines from many small semi circular arcs (aluvial fans)
- 7) Contour lines with a wavy pattern with a outward convex segment
i.e away from hill side (piedmont plain)
- 8) Hashed lines/shading in topomap (sandy plain)

2.9.2 Satellite images:

A) Shadow

- 1) Common shadows (steeper slopes) (upland area)
- 2) No shadows (plain land)

B) Texture

- 1) rough texture (upland area, pedi plain)
- 2) medium to smooth texture (plain land)

C) shape

- 1) triangular/fan like in shaped area (alluvial fan, deltaic plain)
- 2) lobate (palm like) shaped area (deltaic plain)

D) Tone

- 1) yellow tone (in FCC light) (sandy plain, coastal plain, playa)
- 2) light to dark red tone (in FCC light)

(vegetative cover) -> (mountains, hills, inselbergs, bornhardts, domal hills, ridges, fluvial plain, deltaic plain, coastal plain, piedmont plain, alluvial fans)

- 3) Grassy green to dark green tone (scrubby vegetation) ->
(mountains, hills, inselbergs, bornhardts, domal hills, ridges.

plateaus, pedi plain, piedmont plain)

4) Light to dark ash colour tone

(alluvium) -> (fluvial plain, coastal plain, deltaic plain, piedmont plain, alluvial fans)

E) Context

1) area adjacent to sea (coastal plain/ deltaic plain)

2) area adjacent to stream/ river (fluvial plain, deltaic plain)

3) piedmont area (alluvial fans, piedmont plain)

^A) upland area (mountains, hills, inselbergs, bornhardts, ridges, plateaus)

5) plain land area (fluvial plain, deltaic plain, coastal plain, sandy plain, pedi plain, playa, piedmont plain)

2.10 Gallery:

The list of frames constructing the gallery is listed below. The prepositional elements of each frame is also listed against the respective frame name.

2.10.1 Frames:

Features (mountains, hills, inselbergs, bornhardts, domal hills, ridges, plateaus, playa, alluvial fans, piedmont plain, fluvial plain, coastal plain, deltaic plain, sandy plain, pedi plain)

Drainage (dendritic, dichotomous, annular, radial, sub parallel, centripetal, reticulate)

Area (upland area, piedmont zone, plain land)

Shape sat (fan like, triangular, lobate like)

Shape map (concentric ellipses, concentric circles, concentric

ovals, wavy pattern, semi circular arcs)

Tone (red, yellow, ash, grassy green)

Texture (smooth, rough, coarse)

Shadow (dense, absent)

The compatibility relations among the frames is listed below.

Relation between two frames F1, F2 is denoted by F1-F2. All elements are listed against their respective relation.

2.10.2 Compatibility relations:

Drainage-Features

((dendritic, fluvial plain), (dendritic, pedi plain),
(dichotomous, pediment plain), (annular, domal hills), (radial,
hills), (radial, inselbergs), (radial, born hardts), (radial,
plateaus), (sub parallel, piedmont plain), (centripetal, playa),
(reticulate, deltaic plain)),

Area-Features

((upland area, mountains), (upland area, hills), (upland area,
inselbergs), (upland area, born hardts), (upland area, domal
hills), (upland area, ridges), (upland area, plateau), (piedmont
zone, alluvial fans), (piedmont zone, piedmont plain), (plain
land, fluvial plain), (plain land, deltaic plain), (plain land,
coastal plain), (plain land, sandy plain),
(plain land, pedi plain), (plain land, playa))

Shape sat-Features

((fan like, alluvial fans), (triangular, deltaic plain),
(lobate like, deltaic plain))

Shape map-Features

((concentric ellipses, ridges), (concentric circles, domal hills),
(concentric ovals, domal hills), (wavy pattern, piedmont plain),
(semi circular arcs, alluvial fans))

Texture-Features

((smooth, fluvial plain), (smooth, deltaic plain), (smooth, sandy
plain) , (smooth, coastal plain) , (smooth, pedi plain) , (smooth,
playa) , (smooth, piedmont plain) , (smooth, alluvial fans) ,
(coarse, pedi plain) , (coarse, alluvial fans) , (rough, mountains) ,
(rough, hills) , (rough, domal hills) , (rough, bornhardts) ,
(rough, ridges) , (rough, plateaus))

Shadow-Area

((dense, upland area), (absent, piedmont zone), (absent,
plain land))

Tone-Features

((yellow, sandy plain), (yellow, coastal plain), (yellow, playa),
(red, mountains) , (red, hills) , (red, inselbergs) , (red, born
hardts) , (red, domal hills) , (red, ridges) , (red, plateaus) , (red,
fluvial plain) , (red, deltaic plain) , (red, piedmont plain) , (red,
alluvial fans) , (ash, coastal plain) , (ash, fluvial plain) , (ash,
deltaic plain) , (ash, pedi plain) , (ash, piedmont plain) / (ash,
alluvial fans) , (grassy green, pedi plain) , (grassy green,
pediment plain) , (grassy green, mountains) , (grassy green, hills) ,
(grassy green, ridges) , (grassy green, plateaus) , (grassy green,
insel bergs) , (grassy green, bornhardts) , (grassy green, domal
hills))

An alternative approach is to define each feature as a frame, such an alternative list is given below.

2.10.3 Description of observable frames in gallery:

1) Sand; This frame identifies the sandy deposits on a satellite image.

Sand (s1,s2), s1 = Vast areas of light to medium yellow tone on FCC, s2 = Absence of vast areas of light to medium yellow tone on FCC

2) Sand dunes: This frame observes the imagery for various types of sand dunes

(Sand dunes: A heap or mound of sand shaped by wind)

Sand dunes (sd1, sd2)

sd1 = dunes with crescent/ longitudinal/ parabolic patterns are present, sd2 = absence of dunes with crescent/ longitudinal/ parabolic patterns

3) Desert: This frame identifies the desert surfaces on the imagery.

Desert (d1, d2) , d1 = Vast areas of light brown/ dark grey/ yellow brown tone on FCC, d2 = absence of vast areas of light brown/ dark grey/yellow brown tone on fee.

4) Saltpan: This frame identifies the salt pan lakes found in arid environments on a satellite image.

Saltpan (sp1, sp2) , sp1 = A lake with blue brown (deep purple) tone surrounded by bright white tone areas and uplands, sp2 = absence of a lake with blue brown (deep purple) tone surrounded by bright white tone areas and uplands

5) Centripetal: This frame identifies a centripetal drainage pattern on the imagery

Centripetal (c1, c2) , c1 - presence of a centripetal drainage pattern, c2 = absence of a centripetal drainage pattern

6) Volcanoes; This frame observes the imagery for the presence of volcanoes.

volcanoes (v1, v2) , v1 = an upstanding land mass with a prominent radial type of drainage pattern and the peak is either eroded or jilled with lakes, v2 = absence of an upstanding land mass with a prominent radial type of drainage pattern and the peak is either eroded or jilled with lakes.

7) Alluvial fans ;

This frame identifies alluvial fans at the base of uplands.

Alluvial fans (af1, af2) , af1 = a series of fan shape forms combining at the base of an upstanding land mass, af2 = absence of a series of fan shape forms combining at the base of an upstanding land mass.

8) Dichotomous; This frame observes a dichotomous drainage pattern on the imagery.

Dichotomous (del, dc2), del = a series of dichotomous drainage patterns at the base of an upstanding land mass, dc2= absence of a series of dichotomous drainage patterns at the base of an upstanding land mass.

9) Sub parallel: This frame observes for a parallel drainage pattern on the imagery.

Sub parallel (sbl, sb2), sbl = a series of sub parallel drainage pattern at the base of an upstanding land mass, sb2 = absence of a

series of sub parallel drainage pattern at the base of an upstanding land mass.

10) Inselbergs: This frame observes the imagery for the presence of inselbergs.

Inselbergs (isl, is2), isl = small areas of upstanding land mass over a vast plain, is2 = absence of small areas of upstanding land mass over a vast plain.

11) Close to uplands: This frame observes the imagery for closeness of the plain to uplands.

Close to uplands (cull, cu12) , cull = The plain land is at the base or adjacent to an upstanding land mass, cu12 = The plain land is not at the base or adjacent to an upstanding land mass.

12) Lakes: This frame observes the imagery for the presence of lakes on a rocky terrain

Lakes (l1, l2), l1 = a series of lakes with alternating upstanding land mass between the lakes, l2 = absence of a series of lakes with alternating upstanding land mass between the lakes.

13) Dendritic; This frame observes the imagery for the presence of a dendritic drainage pattern.

Dendritic (drl, dr2), drl = presence of a dendritic drainage pattern, dr2 = absence of a dendritic drainage pattern.

14) River pattern: This frame observes the imagery for the presence or absence of a meandering or braided river (meandering:- sinuous bends of a river, braided:- river with multiple threads of channel subdividing and rejoining),

River pattern (rpl, rp2), rpl = presence of a meandering/ braided river, rp2 = absence of a mending/ braided river.

15) Oxbows: This frame observes the imagery for the presence of oxbows lakes. Oxbows (ob1, ob2), ob1 = presence of crescent shape water bodies with light blue/ deep blue tone on FCC adjacent to a river, ob2 = absence of crescent shaped water bodies with light blue/ deep blue tone on FCC adjacent to a river.

16) Back swamps: This frame observes the imagery for the presence of back swamps.

Back swamps (bs1, bs2) , bs1 = presence of small water bodies or patches of dark red tone areas adjacent to a river, bs2 = absence of small water bodies or patches of dark red tone areas adjacent to a river.

17) Reticulate; This frame observes the imagery for the presence of a reticulate drainage pattern

Reticulate (n1, n2) , n1 = presence of reticulate drainage pattern, n2 = absence of reticulate drainage pattern

18) Estuaries; This frame observes the imagery for the delta information.

Estuaries (es1, es2), es1 = river joining sea/lake with a number of distributerics joining a delta / birds foot/ estuaric shapes, es2 = river is not joining sea/ lake with more than one distributory

19) Adj to sea: This frame observes the imagery to see whether the plain land is adjacent to the sea.

Adj to sea (ajsl, ajs2), ajsl = land is adjacent to a sea (i.e the adjoin area to the sea up to the point where there is significant change in the relief) , ajs2 = land is not adjacent to the sea.

20) Lagoons: This frame observes the imagery for the presence of lagoons.

Lagoons (lg1, lg2), lg1 = presence of elongated or elliptical water bodies connected to a sea by means of tidal inlets, lg2 = absence of elongated or elliptical water bodies connected to a sea by means of tidal inlets.

21) Mud flats; This frame observes the imagery for the presence of mud flats.

Mud flats (mf1, mf2) , mf1 = small water bodies or patches of red areas adjacent to a sea, mf2 = absence of small water bodies or patches of red areas adjacent to a sea.

22) Coastal dunes: This frame observes the imagery for the presence of coastal dunes.

Coastal dunes (cd1, cd2), cd1 = presence of small areas of white patches or light yellow tone adjacent to a sea, cd2 = absence of small areas of white patches or light yellow tone adjacent to a sea.

23) Beaches; This frame observes the imagery for the presence of beaches.

beaches (bc1, bc2) , bc1 = presence of a narrow zone of white/ light yellow tone adjacent to a sea, bc2 = absence of a narrow zone of white/ light yellow zone tone adjacent to a sea.

24) Drainage: This frame observes the imagery for the presence of drainage.

Drainage (dp1, dp2), dp1 = presence of small streams, dp2 = absence of small streams.

25) Vegetation: This frame observes the imagery for the vegetation.

Vegetation (vg1, vg2, vg3, vg4) , vg1 = densely vegetated i.e presence of light red/ dark red tone on FCC, vg2 = sparse or confined vegetation i.e presence of small areas of red tone on FCC, vg3 - grass or scrob vegetation i.e presence of light red or yellowish green tone on FCC, vg4 = absence of vegetation i.e absence of red or yellowish green tone on FCC.

2.10.4 The set of geomorphological plains (features) to be identified from satellite image:

- 1) sandy plain (p1)
- 2) deserts (p2)
- 3) playa (p3)
- 4) volcanoic plains (p4)
- 5) pediment plain (p5)
- 6) pedi plain (p6)
- 7) rolling plain (p7)
- 8) fluvial plain (p8)
- 9) deltaic plain (p9)
- 10) coastal plain (p10)

1) sand -> plains

s1 -> p1

S2 -> (p2, p3, p4, p5,p6, p7, p8, p9, p10)

2) sandunes -> plains

sd1 -> p1

Sd2 -> (p1, P2, p3, p4, p5, p6, p7, p8, p9, p10)

3) Desert -> plains

dl -> p2

d2 -> (p1, p3, p4, p5, p6, p7, p8, p9, p10)

4) Salt pan -> plains

sp1 -> p3

sp2 -> (p1, p2, p4, p5, p6, p7, p8, p9, p10)

5) Centripetal -> plains

cl -> p3

C2 -> (p1, p2, p4, p5, p6, p7, p8, p9, p10)

6) Volcanoes -> plains

v1 -> p4

V2 -> (p1, p2, p3, p5, p6, p7, p8, p9, p10)

7) Alluvial fans -> plains

af1 -> p5

af2 -> (p1, p2, p3, p4, p6, p7, p8, p9, p10)

8) Dichotomous --> plains

dc1 -> p5

dc2 -> (p1, p2, p3, p4, p5, p6, p7, p8, p9, p10)

9) Sub parallel -> plains

sb1 -> p5

sb2 -> (p1, p2, p3, p4, p5, p6, p7, p8, p9, p10)

10) Inselbergs -> plains

isl -> (p6, p7)

is2 -> (p1, p2, p3, p4, p5, p6, p7, p8, p9, p10)

11) Close to uplands -> plains

cull -> p7

CU12 -> (p1, p2, p3, p4, p5, p6, p8, p9, p10)

12) Lakes -> plains

l1 -> p7

l2 -> (p1, p2, p3, p4, p5, p6, p8, p9, p10)

13) Dendritic -> plains

dr1 -> (p6, p7, p8)

dr2 -> (p1, p2, p3, p4, p5, p6, p7, p8, p9, p10)

14) River pattern -> plains

rp1 -> (p8, p9)

rp2 -> (p1, p2, p3, p4, p5, p6, p7, p10)

15) Oxbows -> plains

ob1 -> (p8, p9)

ob2 -> (p1, p2, p3, p4, p5, p6, p7, p8, p9, p10)

16) Back swamps -> plains

bs1 -> (p8, p9)

bs2 -> (p1, p2, p3, p4, p5, p6, p7, p8, p9, p10)

17) Reticulate -> plains

rtl -> p9

rt2 -> (p1, p2, p3, p4, p5, p6, p7, p8, p10)

18) Estuaries -> plains

es1 -> p9

es2 -> (p1, p2, p3, p4, p5, p6, p7, p8, p10)

19) Adj to sea -> plains

ajsl -> (p1, p9, p10)

ajs2 -> (p1, p2, p3, p4, p5, p6, p7, p8)

20) Lagoons -> plains

Igl -> (p9, p10)

Ig2 -> (p1, p2, p3, p4, p5, p6, p7, p8, p9, p10)

21) Mudflats -> plains

mf1 -> (p9, p10)

mf2 -> (p1, p2, p3, p4, p5, p6, p7, p8, p9, p10)

22) Coastal dunes -> plains

cd1 -> p10

Cd2 -> (p1, p2, p3, p4, p5, p6, p7, p8, p9, p10)

23) Beaches -> plains

bc1 -> p10

bc2 -> (p1, p2, p3, p4, p5, p6, p7, p8, p9, p10)

24) Drainage -> plains

dp1 -> (p1, p2)

dp2 -> (p2, p3, p4, p5, p6, p7, p8, p9, p10)

25) Vegetation -> plains

vg1 -> (p4, p8, p9)

vg2 -> (p1, p2, p4, p6, p7)

vg3 -> (p3, p5, p7, p10, p2)

vg4 -> (p1, p2, p3, p10)

A case study in which use of evidential reasoning to identify geomorphological features in topomaps and satellite images is explained in this chapter. The various steps are viz., list of geomorphological features to be identified, features on topomaps and satellite images with corresponding hypothesis, structuring the details gathered to build a gallery having frames and compatibility relations, identifying the geomorphological features by making use of the gallery.

CHAPTER-III

3.0 Image to map registration problem:

3.1 Introduction:

Logic is used as one of the important tool for knowledge representation and reasoning. The advantage of logic based knowledge representation is that it has a sound theoretical foundations, hence has unified technique for representation as well as reasoning. AI researchers have been following first order logic as well as propositional logic for these purposes. There has been several techniques and algorithms to carry on reasoning tasks using logic as the medium of expression. Many researchers have also proposed extension of the logic to temporal logic, default logic, nonmonotonic logic etc.. Logic based knowledge representation and reasoning has been successfully applied to many applications in the areas of diagnostic, expert systems, VLSI design and soon.

In this chapter a logic based method is proposed for the purpose of image to map registration. Most often it is required to identify the same portion on the map in order to match the features of satellite images with the map features. This registration of image to map is useful in many contexts such as map updating, data acquisition as well as image interpretation. A map updating is concerned with incorporating additional time varying features which are acquired by satellite and which are not available in the map. In order to carry out the map updating, it is necessary to register the satellite image with the relevant map and then work out a correspondence between these features. In

the context of image interpretation, CIS user notices only bare lines starting from one point and ending at another point in a satellite image, or aerial photograph. It is very difficult to infer or interpret these lines i.e what each shall mean. Suppose the requirement of a user is to know all roads starting from a particular location and leading to a specific location. To solve such problems only spatial information depicted by a satellite image may not be sufficient. It is necessary to employ techniques of spatial reasoning in order to know all roads which may also aid in successful AI base map registration. A reasoning based feature matching may prove to be useful tool in long term CIS research.

3.2 Logical formulation:

The problem of image to map registration can be formulated as a reasoning problem in logic. The formulation is discussed here as a problem in first order logic and then for simplicity, it is reformulated in the following section in the form of clauses of propositional logic. The formulation is based on features such as point, line or chain of a edge pixels, extracted from satellite image using image processing techniques. This involves essentially two important stages, first is to find the edge pixel using image operation and the second is to link a contiguous pixels to identify point or linear feature.

Once the linear features are extracted from the satellite image a higher level feature depicting the interrelationship between a pair of lines is identified. The interrelationships between pair of lines include a *chi crossing* which means each line

crosses the other line, a *tee meeting* means each line touches the other line (starting or ending pixel of one line is on the other line). If starting pixel is same as end pixel of a chain or line then such line or chain forms a closed relation. The other relations interior, exterior, bounds can be arrived by means of finding out whether a chain or line is closed or not and the region with which the relationship is to be arrived fall suitably within, outside or adjacent to it.

To achieve this goal logic programming "prolog" is used to build knowledgebase which consists of facts (spatial objects and spatial relationship details) and rules and free variables in a given goal statement are instantiated with different values and proved true or false in the knowledgebase. So all interpretations which are proved 'true' will be saved in a dynamic database.

3.3 Spatial objects and spatial relationships :

Spatial objects are represented as point, chain, polygon or region. In such representation point will be represented by a single vertex. To represent symbol for display purpose point is used which will not have length or area. Stream of coordinates constitute a chain, which will have beginning and end points. It has length but no area, polygon or region is represented as a set of limited chains having an area and perimeter with starting point is same as end point. Sometimes line will be used as a spatial object which will be represented by one or more number of chains. Polygon is bounded by chains that enclose an area. Such information can be extracted from a map by the process of vector

digitization and the similar information from satellite image can be extracted by the help of library of image processing routines. Spatial relationships are to be known in order to perform spatial reasoning task. The spatial relationships proposed by Reiter [39] are used for the purpose of analysis here.

If a chain c_1 meets chain c_2 at a t junction these two will have tee relationship and if c_1 and c_2 meets at x junction then these two have chi relationship. If a chain is a closed one then the relationship can be named as closed. If the region (being list of chains surrounding it) includes that chain in its list, then that chain bounds that region. The relationships like interior, exterior can also be established based on the constraints imposed to derive them. As in the case of extracting spatial entities, spatial relationships can also be derived using a set of programs using the following logics in case of a satellite image.

3.3.1 Tee and Chi: If some point in one chain is the same as a point in another chain, then both chains will obviously meet. If this point is the starting or ending point of one of the chain, it means that the chain just touches the other. This is a tee relationship, otherwise it is a chi relationship. Once a chi has been detected between two chains, it is not necessary to check for other chi meetings between the same two, we need have to check the two only for a possible tee. For determination of tee and chi relations, line file serves as input.

3.3.2 Bounds: If the region (line or polygon) being a list of chains surrounding it includes a chain in its list, then that chain bounds that line (polygon). Polygon file is the input for this purpose.

3.3.3 Loop: If the first point in the first chain of a line (or polygon) is same as the last point in the chain or line (or polygon) it forms a loop. The input for arriving at loop relations is the polygon file.

Spatial database is created for spatial analysis purpose, hence the non-spatial information has to be linked with spatial information.

We consider chains and regions only to arrive at image axiom set. Also we will arrive at these axioms by considering image primitives for image object. The image primitives are chains and regions. Chains in the image depicts linear scene objects in the scene. Relations tee, chi, bounds, closed, interior and exterior in image maps on to joins, cross, beside, loop, inside and outside respectively in scene.

Various facts in a real world scene are

1) Shore lines form closed loops

$(\forall x) \text{ shore } (x) \supset \text{ loop } (x)$

Elements in shore line set are also in loop set.

2) The inside area of a shore line is land iff its outside is water, its inside is water iff its outside is land.

$(\forall x,y,z) \text{ shore } (x) \wedge \text{ inside } (x,y) \wedge \text{ outside } (x,z) \supset [\text{land } (y) - \text{water } (z)] \wedge [\text{water}(y) \equiv \text{land } (z)]$

Elements x are in shore relation set which forms an inside relation set with elements y and outside relation with elements z , then elements y are in land set if elements z are in water set or elements y are in water set if elements z are in land set.

3) Rivers can not form loops

$$(\forall x) \text{ river } (x) \supset \neg \text{ loop } (x)$$

River set elements x can not be in loop relation set.

4) If a road or river is beside an area then that area is land

$$(\forall x) \text{ beside } (x,y) \wedge (\text{road } (x) \vee \text{ river } (x)) \supset \text{ land } (y)$$

Elements x which lies either in road or river set forms a beside relation with elements y which are on land set.

5) Rivers flow into other rivers or shores.

$$(\forall x) \text{ river } (x) \supset (\exists y \text{ river } (y) \wedge \text{ joins } (x,y)) \vee (\exists z \text{ shore } (z) \wedge \text{ joins } (x,z))$$

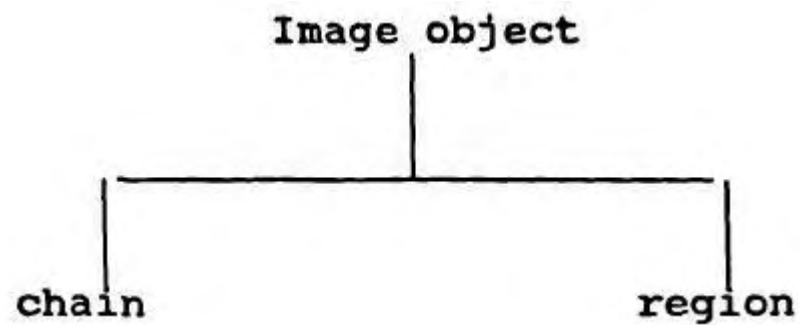
Elements x in river set are also in join relation set with elements y in another river set or with elements z in another shore set.

6) Rivers do not cross each other

$$(\forall x,y) \text{ river } (x) \wedge \text{ river } (y) \supset \neg \text{ cross } (x,y)$$

Set having elements arrived by combining elements x,y in river set does not contain the elements of cross relationship set obtained by using elements x,y .

Taxonomy of image objects can be represented by chains and regions.



Following relations exist between these image primitives.

Relation name	Meaning	Figure
a) Tee (c1,c2)	Chain c1 meets Chain c2 at a t junction	
b) Chi (c1,c2)	Chain c1 meets Chain c2 at a x junction	
c) Bounds (c,r)	Chain c bounds region r	
d) Closed (c)	Chain c is a closed figure	
e) Interior (c,r)	An interior of closed chain c is region r	
f) Exterior (c,r)	An exterior of a closed chain c is region r	

3.4 Axiom set building:

For every image object i

\neg road (i), \neg river (i), \neg shore (i), \neg land (i), \neg water (i).

Image object i will not depict directly road, river, shore, land and water.

Taxonomy hierarchy of scene objects is represented by the following formulae.

$(\forall x)$ scene object (x) \equiv linear scene object (x) \vee area (x)

$(\forall x)$ \neg (linear scene object (x) \wedge area (x))

$(\forall x)$ linear scene object (x) $=$ road (x) \vee river (x) \vee shore(x)

$(\forall x)$ \neg (road (x) \wedge river (x))

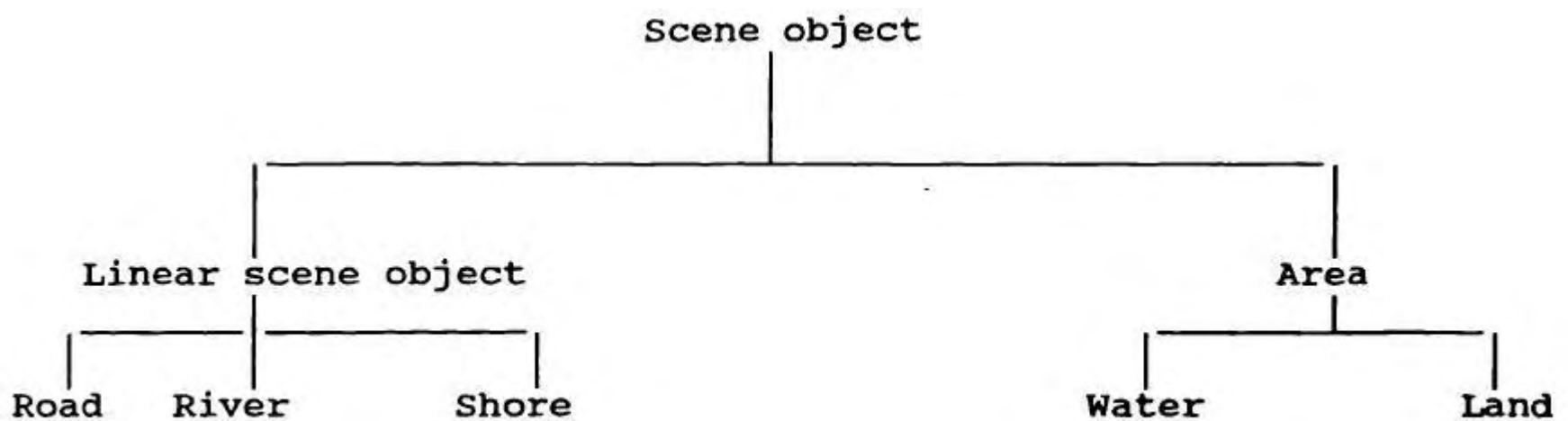
$(\forall x)$ \neg (road (x) \wedge shore (x))

$(\forall x)$ \neg (river (x) \wedge shore (x))

$(\forall x)$ area (x) \equiv land (x) \vee water (x)

$(\forall x)$ \neg (land (x) \wedge water (x))

Linear scene object and area are the two main scene object primitives. Road, river, shore are primitives for linear scene object. Land and water are primitives for area.



Each image relation instance depicts an instance of scene relation and each image object primitive depict scene object

primitive. Chains and regions depict linear scene objects and areas respectively. So tee, chi, bounds, interior, exterior relations maps on to joins, cross, beside, loop, inside and outside respectively. Unique names formulae together with domain closure formulae.

Some more facts to build knowledgebase are given below.

a) \neg road (i) , \neg river (i) , \neg shore (i) , \neg land (i) , \neg water (i)

Image object i will not directly depict road, river, shore, land, water.

b) For every area s

\neg road (s) , \neg river (s) ,

\neg shore (s) ,

land (s) \vee water (s) ,

\neg land (s) \vee \neg water (s) .

linear scene object area can have land or water as primitives but not road, river, or shore.

Following relations exist between scene objects.

a) For every loop (s)

road (s) \vee shore (s) ,

\neg road (s) \vee \neg shore (s) ,

\neg river (s).

For all elements s in loop set lies either on road set or shore set and not on river set. Road or shore form loop, but river can not form loop.

b) For every linear scene object s which is not loop
road $(s) \vee$ river (s) ,
 \neg road $(s) \vee \neg$ river (s) ,
 \neg **shore** (s) .

For all elements on linear scene object which are not in loop set,
lies on road or river set but not on shore set. Road and river
will not form Loops. Shore line form a loop.

c) For every cross relationship (x,y)

\neg river $(x) \vee \neg$ river (y)

Elements x and y in two sets and forming cross relationship will
never lies on river sets. Rivers will not cross each other.

d) For every loop relationship (x,y,z) with loop (x) , y inside of
 x and z outside of x

shore $(x) \supset$ (land $(y) \equiv$ water (z))

For elements x in shore set which are also in loop relation form
outside relation with elements z in water set and inside relation
with elements y in land set.

e) For every beside relationship (x,y) , x not loop
land (y)

Elements x on chain which are not on loop relation and form a
beside relation with elements y in land set.

f) For every beside relationship (x,y) , x being loop
road $(x) \supset$ land (y)

Elements x in road set which are on loop relation also form a
beside relation with elements y on land set.

g) For every linear scene object x not loop

$$\text{river (x)} \supset \left[\begin{array}{l} \left\{ (y \mid (x,y) \in |\text{joints}| \text{ and } y \notin |\text{loop}|) \right\} \text{ river (y)} \\ \vee \left[\left\{ x \mid (x,z) \in |\text{joints}| \text{ and } z \in |\text{loop}| \right\} \text{ shore (z)} \right] \end{array} \right]$$

For all elements x on river and not on loop relation can have joins relation either with elements y in river but not in loop set or elements z in shore but not in loop set.

For the purpose of implementation, image objects chains and regions are used in both (image,scene) domains. In mapping we will treat direct mapping say for example chain to a suitable scene object namely road, river, shore. The relations tee, chi, bounds, interior and exterior are one and same in both image and scene.

3.5 Implementation:

PROLOG representation of spatial objects, spatial relationships and rules (facts in a real world scene) are given here under.

Input lists:

- 1) Chain-list ([c1, c2, c3,...])
- 2) Region-list ([r1, r2, r3,...])
- 3) Loop-list ([c5, c6,...])
- 4) Tee-list ([c2, c1], [c2,c3],...)
- 5) Chi-list ([c3, c4],...)
- 6) Bounds-list ([c1, r1], [c2, r2],...)

Predicates to check for various scene features;

- 1) Bounds-Water (x) :- Water-list (y) , bounds (x,A) , member (A,y) .


```

2) Bounds-road-or-river (x):- Road-list (y)' Bounds (A,x), member
(A,y);
River-list (z), Bounds (A,x), member (A,z).
3) Tee-with (x,y) :- Tee (x,A) , member (A,y) ;
Tee (A,x), member (A,y) .
4) Chi-with (x,y) :- Chi (x,A) , Member (A,y) ;
Chi (A,x), member (A,y) .
5) Both (A,B,x) :- member (A,x), member (B/x) .
6) Shore-Water (A,y) :- Shore-list (B) , not (member (A,B));
Water-list (z) , member (y,z).
Rules set to infer scene object :
Road :- Chain (x) , Road-list (l),
      Road-list (q) , not (member (x,q)),
      not (Bounds-Water (x) ) ,
      not (Tee-with (x,q)),
      decided, append (l, [x] , k) ,
      retract (road-list (l)), asserta (road-list (3c) ) -
River :- Chain (x) , River-list (l),
      road-list (q) , not (member (x,q)),
      Shore-last (r) , not (member (x,r)),
      not [Bounds-Water (x) ] ,
      not (loop (x) ) ,
      Tee (x,-) ,
      not [Tee-with (x,q)],
      not [Chi-with (x,l)],
      decided, append (l, (x) , k) ,
      retract (River-list (l) ) , asserta (River-list (k) ) .

```

```

Shore :- Chain (x), Shore-list (l),
        Road-list (q), not (member (x,q))/
        River-list (r), not (member (x,r)),
        Bounds (x,A), Bounds (x,B), not (A - B),
        Land-list (y) , Water-list (z) ,
        not (both (A,B,y), not (both (A,B,z))),
        Loop (x),
        not (Tee (x,-)),
        not (Tee-with (x,l)),
        not (Chi (x,-)), not (chi (-, x) ,
        decided, append (l, [x], k) ,
        retract (Shore-list (l)), asserta (Shore-list (k))-

Land :- region (x) , Land-list (l),
        Water-list (q), not (member (x,q)),
        Bounds (A,x), Bounds (A,y), Shore-Water (A,y),
        decided, append (l, [x], k),
        retract (Land-list (l)), asserta [Land-list (k) ].

Water :- region (x), Water-list (l),
        Land-list (q), not (member (x,q)),
        not (Bounds-Road-or-River (x)) ,
        Bounds (A,x), Bounds (A,y), not (member (Y,l)),
        decided, append (l, [x], k) ,
        retract (Water-list (l)), asserta (Water-list (k)).

```

3.6 Outline of implementation:

The chains and regions form input data. Each chain is represented with a set of coordinates or contiguous pixels and

region is represented by list of all chains that bound the region.

Set of spatial relationships between spatial objects are computed next. The next step is to build axiom set as mentioned above. These two i.e rules and facts (set of spatial objects and spatial relationships) are to be merged. The process of interpreting or spatial reasoning is to know possible representations of spatial objects (chains and regions) in the set of scene objects (road, river, shore, land, water) for image to map registration. The spatial reasoned information if desired can be highlighted with a typical or specific colour along with non-spatial information. Some initial implementation was carried-out in [55],

3.7 Steps followed for spatial reasoning:

a) Define processing / mapping rules to check feasibility:

This is implemented using PROLOG. All axioms which are constraints on the image, scene and mapping are used as predicates or rules. These rules will govern what could be what in the image and scene.

b) Input the image :

Using a 'C' program by inputting the coordinates through key board or digitizing the image we get chain and region lists.

c) Image object relationships :

The lists of chains and regions are to be checked to see the relationships they can form between themselves. In this process tee, chi etc., lists will be generated. A 'C' program is used for this purpose.

d) Generate all possible spatial reasoned or interpreted details :

The possible combinations of all spatial objects as to what the chains could be (road, river, shore) and what the regions could be (land, water) are arrived using a 'C' program.

e) Test the feasibility of various combinations and output the results:

Each combination stated in step (d) , is used as a goal (possible interpretation) statement in the PROLOG program to check whether it is feasible i.e., proved 'true'. Such possible interpretations or spatial inferred details are updated into a dynamic database.

3.8 CONCLUSIONS:

The approach followed in this study can be very effectively used by any CIS user for his/her benefit for image to map updation, image interpretation and data acquisition. As selective back tracking was not possible in TURBO PROLOG, a 'C' program is written to generate possible interpretations one by one to be checked in PROLOG program. Recursion was also tried but same solution had repeated.

Due to deficiencies in TURBO PROLOG, certain modules were developed using 'C' language. Each time single likely interpretation detail will be generated using a 'C' program for proving it in PROLOG program. The interpreted or spatial reasoned details (which are proved true in the PROLOG code or knowledgebase) will be saved in a dynamic database. At the end the dynamic database will contain all possible interpretations to be available to a GIS user for image to map registration.

CHAPTER-IV

4.0 Object-Oriented gateway to CIS:

4.1 Introduction:

Data about spatial objects (spatial and non-spatial data) is stored in spatial database, which is the central element of CIS. Spatial database must be an useful abstraction of reality and it must support operators for the management and analysis of database. The data in a GIS comes from various sources viz. satellites, aerial photographs, ground observation etc.. New applications require customised data types and interfaces which the conventional GIS is lacking, because they are static and inflexible. Present day applications uses complex structured geometric entities, which inherit features from other entities also. So, need has been felt to develop a system which support complex spatial objects. Hence, modelling involves structural objects.

Due to various reasons stated , need has been felt to develop an object-oriented GIS, which uses class hierarchy and structural object orientation which provide efficient means to model complex geographical objects. Behavioral object orientation feature in object-oriented data model allows user to create specific data types and operators for a particular application. Object-Oriented data model (OODM) permitts spatial and non-spatial characteristics of one class of feature to be inherited by related feature types. All spatial and non-spatial data related to a class of feature are stored together using object-oriented approach, which will help in creation of complex data models.

CIS user is interested to access data (i.e external aspects) irrespective of its internal representation which is hidden from the user. To improve performance, fix a bug etc., it is essential to change the implementation of spatial object. The feature of encapsulation (information hiding) of object-oriented data model can be made use of in developing a CIS. Need has been felt to call a variety of functions i.e., to have specific behavior at run time using same interface. This can't be achieved in traditional CIS. So, feature of polymorphism of object-oriented data model can be used in CIS.

The various sophisticated tools available with object-oriented data model has motivated in developing an object-oriented CIS, which will enable in creating complex real world spatial phenomena with ease. As, developing new CIS using object-oriented concepts is costly, an object-oriented frontend (which is application independent) to existing CIS is designed. In order to facilitate system to be user friendly, graphical user interface (GUI) is to be provided, and this is also incorporated. So, object-oriented gateway to CIS consists of object-oriented frontend to CIS, and GUI modules.

4.2 Theoretical basis:

The terms/features used are briefly explained below [16],

4.2.1 Object:

It is a convenient collection of data that represents a meaningful entity in application. It represents an individual, identifiable item, unit or entity, either real or abstract, with a

well defined role in the problem domain. objects serve two purposes. One, they promote understanding of the real world and two, they provide a practical basis for computer implementation. Objects may be any thing existing, like book, person, polygon, forest etc., and not existing physically but well defined, such as symbol table, binary tree, a chemical equation. An object has a state, behavior, and identity in object-oriented data model. The structure and behavior of similar objects are defined in their common class; the terms instance and object are interchangeable.

The state of an object encompasses all the (usually static) properties of the object plus the current (usually dynamic) values of each of these properties. A property is one inherent or distinctive characteristic, quality or feature that contributes to making an object uniquely that object. All properties have some value. This value might be a simple quality or it might denote another object. Behavior describes how an object acts and reacts in terms of its state changes and message passing. The behavior of an object is completely defined by its actions.

4.2.2 Object identity:

Each object should have a unique identity to denote the object independently of its behavior or state. This implies that all objects have identity and are distinguishable by their existence and not by descriptive properties that may have. Most programming and database languages use variable names to distinguish temporary objects. Most database systems use identifier keys (primary keys) to distinguish persistent objects, mixing data value and identity.

4.2.3 Class:

An object class describes a group of objects with similar properties (attributes), common behavior (operation), common relationships to other objects, and common semantics. Person, cycle and process are all object classes. The abbreviation class is often used instead of object class. Most objects derive their individuality from differences in their attribute values and relationships to other objects. However, objects with identical attribute values and relationships are possible.

We can abstract a problem by grouping objects into classes. Common definitions such as class name and attribute names are stored once per class. All objects of a class can benefit from the code reuse by writing all operations once for each class.

4.2.4 Complex object:

It is used to represent objects that are composed of other objects as parts (components) and/or containing other objects as values/reference to their attributes. Two types of reference semantics exist between a complex object and its components at different levels. The first type, which is called 'ownership semantics' applies when the sub-objects of a complex object are encapsulated within the complex objects and are hence considered part of the complex object. This is referred to as the 'is-part-of' or 'is-component-of' relationship. The second type which we call 'reference semantics' applies when components of the complex object are independent objects but some times may be considered as part of complex object. This is called

'is-associated-with' relationship, since it describes an equal association between two independent objects. The desire of current database users is to abstract these complex objects without disturbing the structure (in natural way),

4.2.5 Encapsulation:

Encapsulation (also known as information hiding) consists of separating external aspects of an object, which are accessible to other objects, from the internal implementation details of the object, which are hidden from others. Encapsulation prevents a program from being so interdependent that a small change has massive ripple effects. The implementation of an object can be changed without affecting the applications that use it. One may want to change the implementation of an object to improve performance, fix a bug, consolidate code, or for porting.

Encapsulation is not unique to object-oriented languages, but the ability to combine data structure and behavior in a single entity makes encapsulation clearer or more powerful than in conventional languages they separate data structure and behavior.

4.2.6 Hierarchy and inheritance:

Another important characteristic of object-oriented system is that they allow type or class hierarchies and inheritance. These two are interrelated. Type hierarchies and class hierarchies are conceptually different, but because in the end result they often lead to mean the same thing. Hierarchy is often explained in terms of class. Classes may form a hierarchy (with super class -

sub class relationship) such that objects in any given class are automatically members of its super class. All methods and attributes that apply to a class, apply to its sub classes as well and this is called inheritance.

Inheritance is the sharing of attributes and operations among classes based on a hierarchical relationship. Each sub class incorporates, or inherits, all of the properties of its super class and adds its own unique properties. Class hierarchy is the relationship among classes where one class shares the structure or behavior defined in one (single inheritance) or more (multiple inheritance) other classes. This inheritance defines a kind of hierarchy among classes in which a sub class inherits from one or more super classes, a sub class typically augments or redefines the existing structure and behavior of its super classes.

An object class a inherits the attributes and methods of an ancestor class b if and only if they do not conflict with the attributes and methods that have been defined for a directly or for any ancestor class of c that lies between a and b.

4.2.7 Binding:

Binding is the act of associating name to a type. It is generally seen in terms of time when the action takes place. Static binding or early binding means that the behavior of an object is fixed at the time of compilation. Dynamic binding or late binding means that the actual behavior of an object is known only at run time. So, binding is not made until the object designated by name is created.

4.2.8 Polymorphism:

The binding of specific behavior, among the different implementations, at run time is known as polymorphism. It is also known as over loading. It can be facilitated to both objects and operators. For objects, it represents a concept where a single name may denote objects of many different classes, that are related by some common super class. It can be termed as object polymorphism. For, operators, termed as operation polymorphism, same operator name or symbol (i.e., an operator or a function) is bound to more than one different implementations, depending on the type of object or the type of the parameters passed. Polymorphism is the result of interaction of inheritance and dynamic binding. It is one of the most powerful features of object-oriented programming (OOP) and distinguishes OOP from traditional programming.

4.2.9 Extensibility:

It assembles the pre written modules and classes into new tailor made systems. The technological improvements can be quickly incorporated into the reusable modules and thus the system can always remain up to date. The development of new systems become rapid and economical. New techniques can be evaluated without making significant modifications.

4.2.10 Object-Oriented DBMS [20] [52];

This is a DBMS with an object-oriented data model (OODM) . object-oriented database management system (OODB) apart from

supporting the object-oriented features explained above (4.2.1 to 4.2.9) should also provide DBMS features such as persistence, secondary storage management, transaction management, concurrency control, recovery and query along with structural object-orientation (support of composite objects) and support user defined data types

4.3 Need for object-oriented CIS (OOGIS) [41]:

The conventional GISs are static and inflexible with regard to new applications that may require customised data types and interfaces. Geometric entities which are complex structured in nature, possess inherited features of other entities. Also there is a need to support complex objects. Modelling of structured objects, adoption of emerging standards and integration of efficient access methods for large spatial databases are falling into the challenging areas in development of any GIS. In conjunction with RDBMS, proprietary GIS databases are also to be used which resulted to use OODBMS for GIS databases [31].

So, keeping this in view, in object-oriented GIS, class hierarchy and structural object orientation provide efficient means to model complex geographical objects. In addition, behavioral object orientation (support of user defined types and operators) allows to facilitate support of specific data types and operators for particular application. Moreover the object-oriented approach implements a system where all spatial and non-spatial data relating to a class of feature are stored together permitting the creation of complex data models. OODB permits spatial and

non-spatial characteristics of one class of feature to be inherited by related feature types. Object-Oriented approach also encapsulates all applicable operations within each class of features so that operations on the feature are manifested equally in the spatial and attribute data. All topological relationships can be stored for each instance. Object-Oriented approach is having clear advantage in comparison with the fixed set of data types and operators that is typical for most commercial GISs that exist. So, using object-oriented database design which offers most sophisticated tools, real world models of spatial phenomena in a GIS can be created with ease at present.

4.4 Object-Oriented GIS (OOGIS) vs GIS:

Following are the comparisons between GIS and OOGIS.

4.4.1 Generic concepts:

The generic concepts of an object-oriented database design are object, classification, relationship, inheritance relationship and aggregation. Inheritance relationships among the various objects are embedded components of geographic entities. Complex real world entities can not fit in relational database, so artificial decomposition into several relational tables becomes necessary with common attributes (foreign keys) for linking. By decomposing efficiency is decreased and there is a possibility of losing the semantics associated with data. But in OOGIS, the user is able to manipulate the complex objects as if they were single units. So, the generic types mentioned above help the user to model/abstract the complex entities in natural way. In traditional

CIS, a complex object must be decomposed over different relations.

4.4.2 Adhoc query facility:

Adhoc query facility is not possible with CIS, but provide menu driven queries and restricted class of queries. Programming language interface to CIS provide user to query in adhoc manner and interactive interface to CIS is best used for pointing a geographic object on the display and put query centered on that object. CIS can offer more flexibility when it can handle searches both in spatial and object centered ways.

Object extensions to SQL (O-SQL) are proposed in TIGRIS object-oriented data model for CIS [18]. These extensions include:

The ability to invoke any method on any object directly from the query language for use either in predicates in selects, in relational operators or for updates.

The ability to define class sensitive language macros that interpret query statements in accordance to class context (a query over loading operators)

The ability to group query statements together in single units of execution, with inter query communication via temporary relations managed by the system.

4.4.3 Concurrency:

Most of the applications require operating in multi user environment. Database systems usually provide mechanisms to facilitate controlled sharing of data and resources. This controlled sharing means that certain data items can only be read

or updated by a selected group of users. Only authorised users who have ownership permissions are allowed to change the data. Due to data encapsulation and multiple inheritance features of OOGIS, the situation is less complex to support controlled sharing.

4.4.4 Distribution:

Large amount of data from different sources is common in GIS applications. The performance of the system can be improved if the data can be stored in different locations with-out affecting the integrity of the database. With out losing the integrity and consistency the updates/retrievals are to be performed and the distribution of data is transparent to the user. Distributed data management has made significant progress since the introduction of relational model and object-oriented models. Traditional GIS data models, on the other hand, do not lend themselves easily to the distributed data management. Also complex geographical objects and user defined data types will be supported in more natural way in OOGIS.

4.5 Need for cooperative environment [2]:

For any application of GIS, data .is generated by multiple sources (digitized, satellites, ground observation, weather station, aerial photographs etc..) and this data is accessed, processed and transformed by many users and available for use to other users also. Lack of coordination among all these different users of data may render large amounts of work useless.

Activities related to geographic applications involve modelling geographic phenomena (source of modelling reality). Modelling takes place by applying successive transformations to some input data, which results in creation of derived data. This derived data can not be interpreted correctly (probably by experts in other domains who may have to use this derived data) without the knowledge about how data was created and global model of which it is part. Furthermore, the complex models used require the collaboration of several people/experts and they may have to use the model which is designed by others. One more problem added to current situation is the model itself may undergo changes as more knowledge and precise data becomes available in future. Hence, there is need for considering cooperative environment in the design of more sophisticated CIS.

The main aspect of cooperative modelling is that, multiple agents interact with each other to create complex structures/models. Another aspect is temporal cooperation, the data sets may be used and accessed years after they are created. This raises the issue of defining cooperation between actions executed may be years apart. Existing CIS do not provide any support for the cooperative work. So, there is a need to facilitate cooperative environment, and still allowing users to take advantage of CIS.

4.6 Need for graphical user interface (GUI):

The performance and efficiency of any system can be looked and felt by user, only when proper interface is provided. In case

of information systems, interface in interactive manner or through programming languages avail user to build their application and to perform queries. Graphical user interface (GUI) is, no doubt, so user friendly and becomes must in case of CIS. The spatial analysis system involves large volumes of data, display of geographical objects like maps necessitates the option of GUI.

Main use of GIS is performing spatial queries in different ways before coming to conclusion for any decision making application. To support spatial query, GUI is one not to leave. So, GUI is also an important issue of consideration for the success of OOGIS.

4.7 Aim of object-oriented gateway to GIS:

It is much better if existing GIS is converted to OOGIS, as it is very costly to develop new OOGIS.

So, the aim of object-oriented gateway to GIS is two fold, one is to develop an object-oriented frontend to existing GIS, such that the flavor of object-orientation and the power of handling spatial data/objects of GIS are blended together to evolve a fully functional, flexible, application independent, user friendly GIS. Secondly, to provide GUI, schema building facility, automated change propagation, and version control in existing GIS. It is also possible to achieve cooperation between users working in two different underlying GIS, developing two sub models of a global application.

4.8 Object-Oriented data model (OODM):

Data model can be seen as a collection of conceptual tools for describing data relationships, data semantics, consistency constraints and operations. A data model describes on the abstract level objects and their behavior. Data structure is a specific implementation of data model and it fixes performance aspects such as storage utilisation and response time. Object-Oriented data models are proposed for taking care of characteristics which differ from those of traditional business applications such as complex data structures, longer duration transactions, newer data types for storing images or large textual items and need to define non-standard application specific operations for applications like CAD/CAM, image and graphic databases, CIS and multi media database.

Fig 4.1 below shows the concepts drawn from different areas towards the development of an object-oriented data model (OODM).

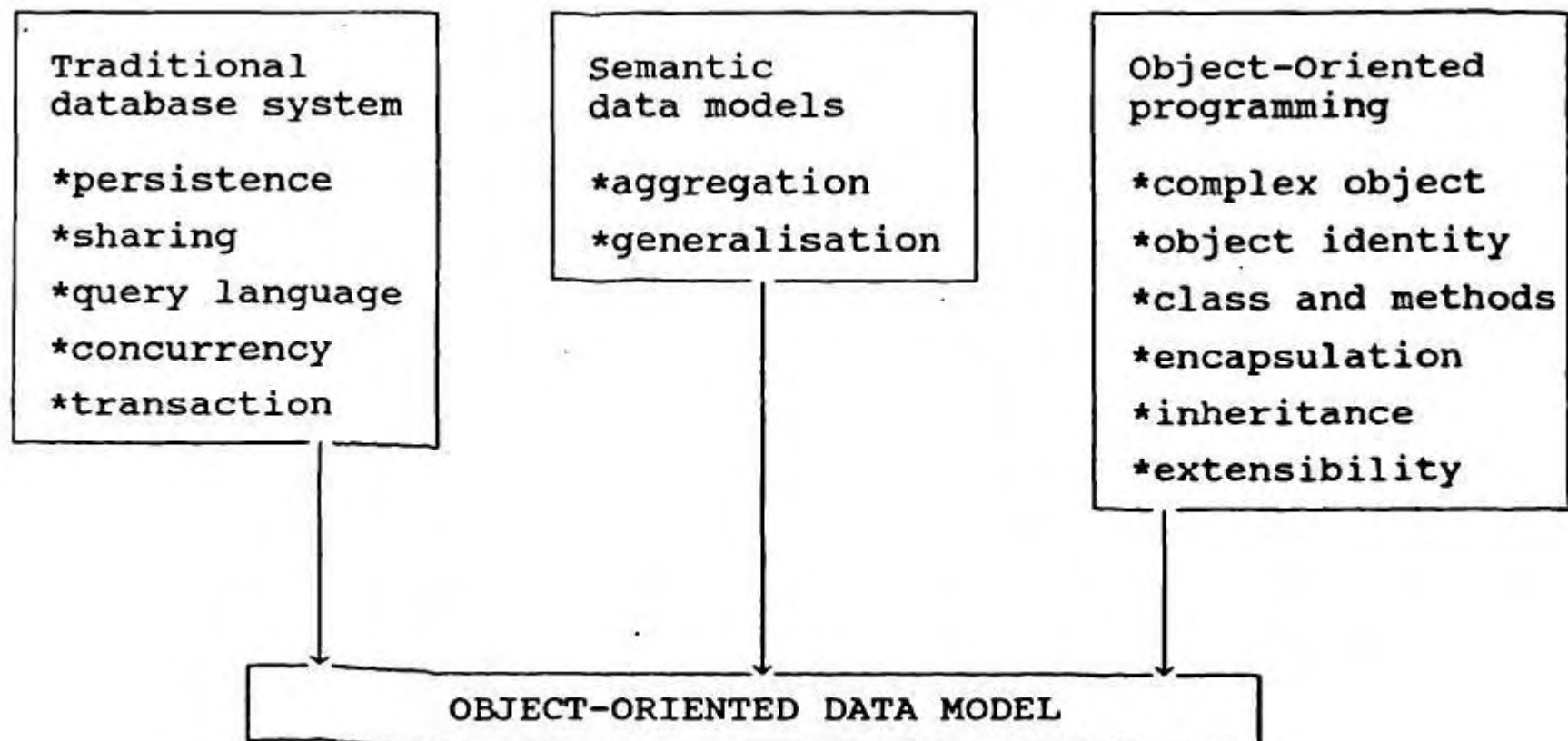


Fig: 4.1 Object-Oriented data model

As shown in Fig 4.1, various concepts from traditional database systems, semantic data models, object-oriented programming are put together to form an object-oriented data model. The database features viz., persistence, sharing etc., are required for any data model. The object-oriented data model originates from the tradition of semantic data modelling, but it takes view of data that is closely aligned with object-oriented programming languages.

The entities in an object-oriented data models uses single modelling concepts: the objects. The Object-Oriented paradigm is based on five fundamental concepts [4].

1. Object is used to model every real world entity and is associated with a unique identifier.
2. Each object has a set of instance attributes (instance variables) and methods. The value of an attribute can be an object or a set of objects. This characteristic permits arbitrarily complex objects to be defined as an aggregation of other objects. The set of attributes of an object and set of methods represents the object structure and behavior respectively.
3. Object's status is represented by the attribute values and can be accessed or modified by sending messages to the objects to invoke the corresponding methods. Methods define the operations to manipulate or return state of an object. Objects can also communicate with one another by sending messages. For each message understood by an object, there is a corresponding method that executes the message.

4. Objects sharing the same structure and behavior are grouped into classes. A class represents a template for a set of similar objects.

5. A class can be defined as a specialisation of one or more classes. A class defined as a specialisation is called a sub class and inherits attributes and methods from its super class.

Aggregation and generalisation mechanisms of semantic data models in addition let the user represent relationships among objects and among object collections.

4.8.1 Semantic data models:

The OODM uses semantic data models to represent complex objects because traditional relational models are inadequate such as limited expressive power (semantic content) and number of problems can not be naturally expressible in terms of relations. Spatial systems are a case where the limitations become clear [53]. In a typical spatial system a polygon is decomposed into rings, which are again decomposed into set of chains. The polygon will have attributes polygon ID, chain ID, ring ID, ring seq. Ring will have attributes ring ID, chain ID, chain seq. The attributes of chain are chain ID, start node, end node, left polygon, right polygon and attributes of node are node ID, point ID.

The attributes of point are point ID, x coordinate and y coordinate. Each of these is a relation. This model of polygon as a set of relations, though complete, is low level and depends on user's capability of decomposing the complex objects or entities into relations. Semantic models aim to provide more facilities for

the representation of the user's view of systems. These models (ex: entity-relationship model, functional data model, semantic data model) as well, decouple these representations from the physical implementation of the databases. These provide more powerful abstractions for database schema than the relational, hierarchical, and network models can support. Aggregation, generalisation, specialisation and association are various abstraction constructs with the first two being proposed by Smith & Smith (1977).

4.8.2 Aggregation and generalisation:

Aggregation refers to an abstraction in which a relationship between objects is regarded as a high level object. Aggregates are abstract entities that contain heterogeneous components. In relational terms, an aggregated tuple has attributes that are themselves tuples of different relations. This relationship can be nested to any depth. Aggregation is one way of representing hierarchical structure among tuples of different relations. The classes or types are constructed by attributes or components and there is likely some attribute belongs to two classes but corresponding objects do not share the instance of common attribute.

Generalisation refers to an abstraction which enables a class of individual objects to be thought of generally as a single named object. Elements of a class can be specialised and grouped into sub classes. A sub class inherits all the properties of its parent class. Elements of a sub class also belong to their parent

class. This sub class mechanism is called generalisation. Generalisation views a set of objects or a set of types/classes as one generic type/class. In case of objects we call it as 'classification' (i.e., 'class' in OODM) and in case of class we call it as 'type-type' generalisation or simply generalisation. All properties of the generalised type can be inherited downward to the constituent types. The arrows indicate the direction of generalisation.

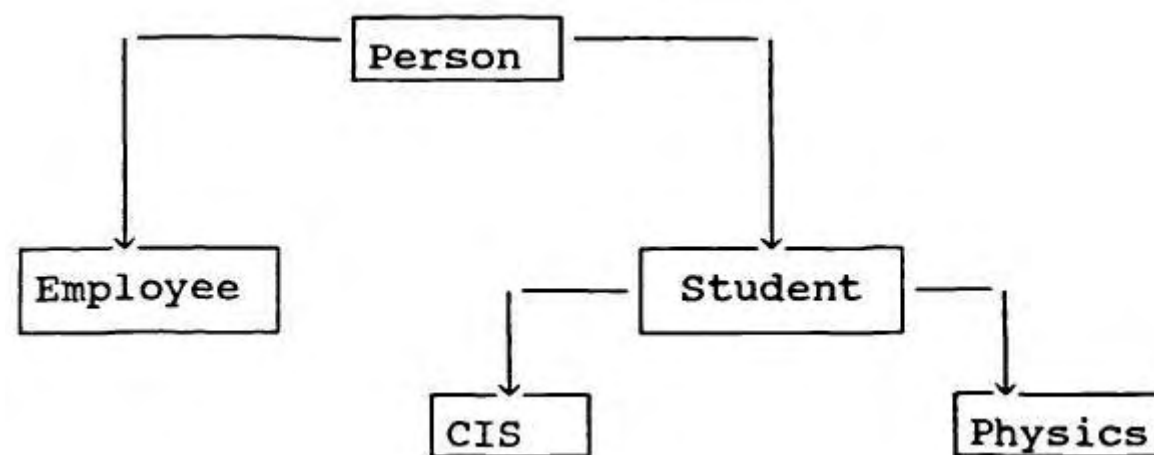


Fig 4.2 Generalisation hierarchy

In fig 4,2, employee and student together constitute higher level type person. Student is a generalisation of CIS and Physics. The attributes of person are inherited by each constituent (eg: employee and student). These abstractions can be modelled in OODB with the concepts hierarchy and inheritance.

4.8.3 Class hierarchy and inheritance:

The class hierarchy specifies the 'is-a' relationship among classes. If two classes c_1 , c_2 are related with 'is-a' relation, i.e c_1 'is-a' c_2 then all properties (including attributes and methods) defined on c_2 will also be defined on c_1 . c_1 is called a

sub class of c_2 and c_2 is a super class of c_1 . The inheritance concept says that the properties specified for a class are inherited by all its sub classes in the hierarchy. This is similar to the generalisation construct in semantic data modelling.

4.8.4 Class lattice and multiple inheritance:

If a class can have only one super class, the class hierarchy forms a tree. Sometimes, it is useful for a class to have multiple super classes. This generalises the class hierarchy tree to a directed acyclic graph (DAG), simply called a lattice. In a class lattice, a class can inherit properties from multiple super classes. This feature is known as multiple inheritance.

4.8.5 Additional features of OODM:

The following additional features, an OODM can support in addition to the basic concepts.

4.8.5.1 Composite object [21]:

Many applications find it useful to provide an 'is-part-of' relation between an object and the object it references in addition to 'is-a' relationship. This is similar to aggregation concept in semantic data modelling. A composite object can be defined as an object with a hierarchy of exclusive objects that form a tree structure. However, it can also form a DAG structure, if component objects can be shared. The composite objects are often treated as units in data storage, retrieval, and integrity enforcement.

4.8.5.2 Version control:

Version control captures the record of evolution for data objects. Users in CAD/CAM, CIS etc., environments often desirous to work with multiple versions of an object, before selecting a right one that satisfies as their requirements. An object version can have only one previous version. Two properties, previous version, and next version, may be used to express the appropriate temporal relationships among object versions. Next version property can be multi valued, thereby allowing a given object version to have multiple successors and this version history generally forms a tree structure.

4.8.5.3 Equivalent objects:

In some applications such as design databases, an object can have different representations that are equivalent, the purpose of which is to impose constraints on the databases. OODM uses generic objects to represent the semantics of equivalent objects explicitly. A generic object contains attributes that identify different representations of the same object. Modifications in one representation are reflected in others by adding constraints to the definition of generic objects.

4.8.6 Data operations in OODM:

So far we have seen conceptual schema part of OODM, i.e., how the real world is represented as objects and relationship between these objects. The second part 'data operation' in OODM include schema definition, database creation, data retrieval and data update.

The schema definition operations are high level operators/ used to describe application schema structure, various class definitions and relationship among the classes (class hierarchy). In CIS, CAD/CAM etc., applications the requirements of the user, source data, environment changes with time, hence the definition of schema is liable to be changed, which generally does not change in business applications. In order to modify the schema definition the operators similar to 'change class', 'add class', 'delete class', and 'change class hierarchy' are to be provided.

The operations for creating database have the form,
Create object (object name, object definition) or
Add object (object name, object definition) , are used to input the data in the required form (structuring the data) .

The operation for data retrieval is of the form
retrieve (object name, list of attributes, list of conditions)'
'List of attributes' specify the attribute names of interest specified as [all], [all but atb1], [atb3, atb4] etc..
'List-of-conditions' has the form [atb1 logical operator value], [default] etc..

The operation of data update is of the form
update (object name, object value).

All operations stated above can be performed by making use of an interface language.

4.8.7 Object-Oriented databases:

An object-oriented database system (OODB) is a DBMS with an object-oriented data model (OODM) , and the key feature of OODB is

the power they give the designer to specify both the structure of complex objects and the various operations which can be applied to these objects. Developers have selected one of the approaches mentioned here under:

Extending a relational system to support the concepts of objects

Extending an object-oriented programming language to include persistence, sharing etc., database features.

Many OODBs have been developed to meet the requirements of various applications viz., CAD/CAM, CASE, document processing etc., by making use of one of the above approaches. Knowledgebase representation schemes are to be incorporated in OODB to support AI applications. Orion developed by Mcc and Iris developed by HP will fall into the category of extending a relational system. Gemstone (by servo logic), Encore (by Brown univ), and object store (by object design) are the OODBs as an extension of object-oriented programming languages. The first one was developed using 'small talk' and next two were built on C . Although many OODBs have been proposed using these approaches, they differ in their interpretation of OODM. All OODBs (proposed/developed) support the concept of complex objects and object encapsulation at different levels. They can be categorised on the level of object orientation.

4.8.8 Structural object-orientation:

A data model supporting the construction of composite objects is called structurally object-oriented. So the object (tuple) do

not have to be atomic (as in the case of relational model), but may be composite in turn. A system that supports composite objects must provide specific operators, such as, duplication and deletion of objects with their sub structures, or navigation through an object structure. This is sometimes called operationally object orientation. It should provide atomic types (eg: integer, character) and certain type constructs (eg: tuple, set, list) which will enable users to construct application specific complex data structures. Structural object orientation can be used in a natural way to model built in objects (i.e., objects embedded in other objects in the sense of part of hierarchy) or to specify shared objects which are included in several other objects simultaneously. 'Class' construct is generally used to attain structural object-orientation in OODBs. To maintain reference between the objects and component objects object ID plays a vital role.

4.8.9 Behavior object-orientation:

A data model is called behaviorally object-oriented if it supports user defined types and the definition of operators (methods) that are applied to these types. Once the user defined type and its associated operators are specified, this new type can be used like a system defined type. Operators or methods are defined by an interface (name, input parameters, output parameters) and a program to compute it. The OODM will take care off operators not applying to inappropriate objects. To, apply an operator, one only needs to know its interface; no knowledge is required about its implementation.

In order to increase portability of an application, it should be generic and independent of the specific hardware configuration irrespective of the way the data model is defined and the success of object-oriented system design.

4.8.10 Modelling real world phenomena [11]:

The empirically real world verifiable facts referring geographical reality which constitutes a spatial information system is to be modelled. This data model is a limited representation of reality since the facts may not be certain, and is constrained by the finite, discrete nature of computing devices. The spatial databases representing real world phenomena is composed with logical units, spatially referenced entities must be abstracted, generalised or approximated in the process of creating the database. In spatial databases, modelling plays a major role and controls the view of the world which the user ultimately receives.

From the user's point of view, it is suitable to start with the real world. The world consists of fully defined and incompletely defined entities/objects. The incompletely defined spatial objects set is the source of the problem and includes spatial objects from different sources depending on the needs of any user. Euclidean geometry to reason about spatial arrangements and network topology when plan a trip or navigate any automobile are the mostly used methods to conceptualise space by which most of the applications can be addressed. It is not that reality changes, but the concepts used to structure perception of the

situation differs. In order to cope with the complexity of the realisation, we have to abstract details and concentrate on the aspects that are important for the task at hand.

The environment where the nature of the variations, the objects that can be identified and the identification may vary with the discipline, spatial scale and personal opinion gives few problems for exact models of real world phenomena or reality. The better model for complex real world objects is to regard them as some type of complex continua and because they can not be seen in their entirety, they can only be described by sampling and reconstructed by interpolation. A CIS will be successful only if it can present the user, an accurate view of the world, and to do so requires efficient access to a database, and the use of accurate data models.

4.9 Different approaches to OOGIS:

The object-oriented term has received attention recently in the CIS literature as many of the computer science concepts of object-oriented programming and databases have stimulated discussions in the spatial context. The object-oriented notion of object identity is clearly more compatible with the object view of reality than the field view, and the systems currently being marked as object-oriented, rather than layered, seem to be aimed at those applications in which the object view is more acceptable.

Following are different approaches to OOGIS, for which some prototypes have been developed as an attempt to use

object-oriented model in CIS technology. These approaches can also be used, to develop a non object-oriented CIS model.

1. The extension of OODB functionality to existing CIS:

In this approach capture, manipulation, analysis, presentation capabilities of the spatial data are availed by the CIS, and the storage, management and other object-oriented features are to be enhanced over the existing CIS. One example in this line is the commercial CIS SICAD of Siemens NIXDROF with its data management component GDB, offers numerous database functionality.

2. The extensions of an existing DBMS by geometric and geographical functionality: For this approach, OODB seems to be an interesting option. A kernel consisting of geographical and geometric building blocks can be built on underlying OODB as base classes. These building blocks are classes and methods for representation and management of spatial information. Other CIS functionality can easily be build with the help of kernel. The first prototypes have been developed in this line are based on O2 and Postgres Godot is the OOGIS developed based on this paradim.

3. The coupling of GIS with existing DBMS, is a hybrid approach, in which some or all geographical information are stored in specialised structures, and attribute information is stored in DBMS.

4.10 Approach followed for OOGIS:

As explained already, there are many advantages if CIS is developed on object-oriented concepts. The following are steps followed in achieving this goal.

4.10.1 An object-oriented frontend to any CIS:

This is an extension to an existing GIS. In this there is a clear separation between the underlying GIS and the extension part and this separation plays an important or vital role in achieving the portability, extensibility and (cooperative) modelling capability.

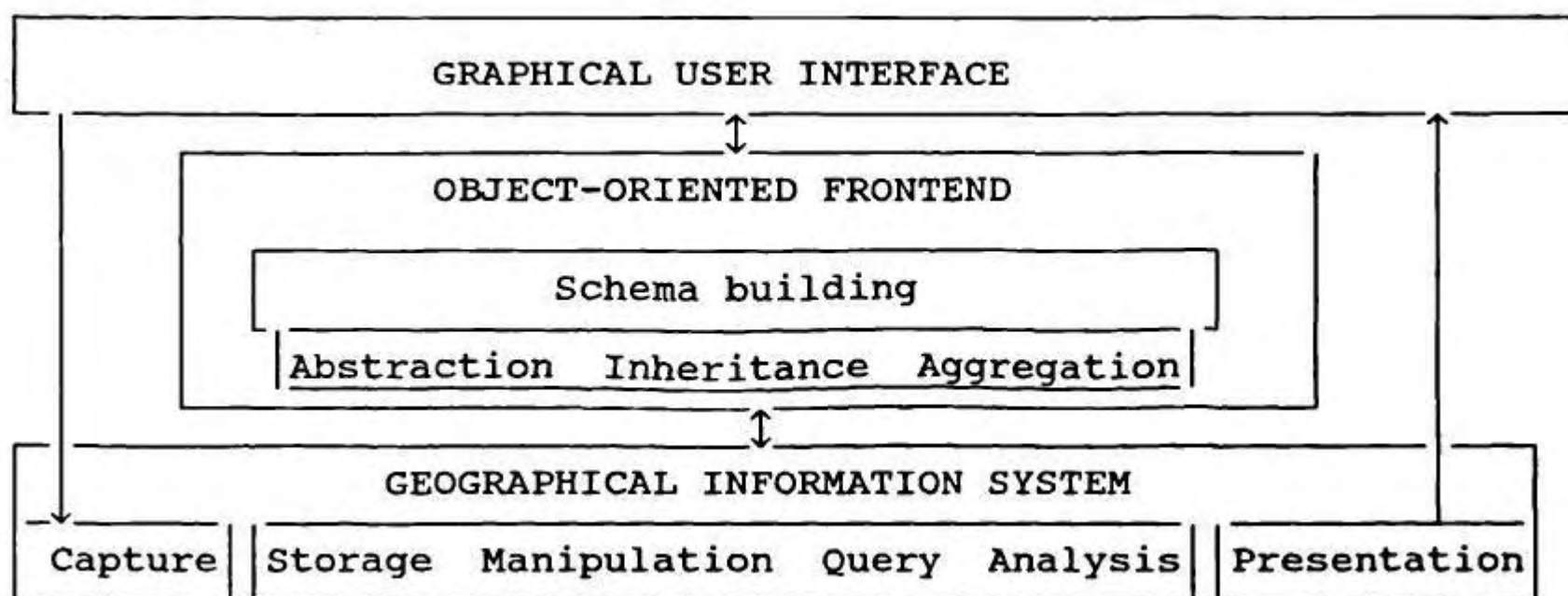


Fig 4.3 Block diagram of object-oriented gateway to GIS

For the functionality, such as spatial data management, data storage and spatial query processing, we have to rely on underlying GIS. Object-Oriented frontend implements the modelling capabilities, the underlying system lacks. The underlying GIS can be heterogeneous GISs developed in different approaches. Portability can be achieved, as the frontend is the upper layer

for different CIS. One has to rewrite the parser, which is responsible for communication between frontend part and underlying GIS. This frontend is not restricted to an interface layer between/ it allows users to model complex objects and for building schema. The frontend will augment the underlying GIS with features like, object identity, structural object orientation, and behavioral object orientation, which in turn may perform multiple transactions or multiple calls to underlying GIS. To attain extensibility, inheritance and reusability are helpful. In order to facilitate cooperative modelling, the frontend supports version control, multiple views of modelling and automated change propagation.

The system may be a tool, with which users abstract and model the data of their interest or the system is the outcome of modelling of tools which manipulate the data of user's interest. The information system developed in first approach gives overall modelling authority to the user, thus it is flexible and support varieties of applications, where as the one developed with, later approach suffers from certain limitations such as inflexibility, static and gives narrow view to the user, but it may be simple to use, because user merely follows the existing model for which tools are provided.

User has to be tuned to model or some tools it supports to use an information system, which is developed in latter approach. User finds greater difficulty when his need, application context, or data changes. User can not build any schema for application. The user has to build his own schema for each application, which

is a difficult task. The limitation can be overcome by providing an easy environment to define the schema and GUI. It reduces much of the key in work as user need not deliver many commands to built schema. Thus the former is superior to later.

Here, the main effect is to convert an already developed GIS without facilities for creating schema into a GIS which is flexible and allows the user to design his own model of the real world phenomena or geographical reality. The GUI makes use of graphics on a bit mapped video display. Graphics provides better utilisation real estate, a visually rich environment for conveying information, and the possibility of what you see is what you get video display of graphics and formatted text prepared for a printed document. In GUI, the video display, no longer confined to echo text, but becomes a source of input showing various graphical objects in the form of icons and input devices such as buttons and scroll bars. The user can directly manipulate these objects on the screen using a pointing device mouse or key board. Graphic objects can be dragged, buttons can be pushed, and scroll bars can be scrolled. The user can directly interact with objects on the display, rather than one way cycle of information from the key board to the program and to the video display.

4.11 Design:

4.11.1 Introduction:

The object-oriented approach views a system as a set of objects, each object having a well defined operations, and a

transformation function that transforms the objects by performing operations on the objects.

The object-oriented design methodology consists of three parts:

Define the problem

Develop an informal strategy

Formalise the strategy

The third step has, further, four phases

1. Identify the objects and their attributes
2. Identify the operations on the objects
3. Establish an interface and
4. Implement the operations

These three steps can be merged and then divided into four stages, which follow software engineering life cycle. The soul of the design will be found in the control concerns of object-oriented design. The stages are

Analysis

System design

Object design

Implementation

This object-oriented methodology is proposed by Rumbaugh et al. The methodology consists of building a model of an application domain and then adding implementation details to it during the design of the system. They called this approach as object modelling technique (OMT). Since the design follows the same approach as OMT, it is necessary to brief about this methodology.

1. Analysis:

It is concerned with understanding and modelling the application and the domain within which it operates. The conceptual view of the proposed system is arrived based on the initial input to this phase: the problem statement. In this stage model will be built of the real world situation showing its properties. The analysis model is a coarse abstraction of what the desired system must do, not how it will be done. The objects in the model should be application domain concepts and not computer implementation concepts such as data structures. The output from analysis is a formal model that describes the objects and their relationships, the dynamic flow of control, and the functional transformation of data subject to constraint.

2 System Design:

The overall structure is determined in this phase. During system design, the target system organised into sub systems based on both analysis structure and proposed architecture. The system designer must decide what performance characteristics to optimize, choose strategy of attacking the problem, and make tentative resource allocations.

3 Object design:

During this phase, design model is built based on analysis model but containing implementation details. The designer adds details in accordance with the strategy established during systems design. The focus of object design is data structures and algorithms needed to implement each class. Both application domain objects and the computer domain objects are described

using the same object-oriented concepts. During object design, practical designs are produced by elaborating, refining and then optimizing the analysis models. During the object design stage the shift in emphasis is from application concepts to computer concepts. The basic algorithm is chosen first to implement each major function of the system and object model is optimized for efficient implementation and this model structure is then optimized for efficient implementation.

4 Implementation:

The object classes and relationships developed during object design are finally translated into a particular programming language, database, or hardware implementation. Programming should be a relatively minor because all the hard decisions should be made during design. In this implementation phase, it is important to follow good software engineering practice so that the design is straight forward, and the implemented system remains flexible and extensible. Some initial implementation . was carried-out in [56],

Through the system development cycle i.e. analysis through design to implementation, object-oriented concepts can be applied- In all the above phases, the OMT methodology uses three kinds of models to describe the system:

a) the object model, describing static structure of the objects, in a system, and their relationships

b) the dynamic models describing the interactions among the objects of the system and

c) the functional model, describing the data transformation of the system.

The three models are orthogonal part of the description of a complete system and are cross linked and each model is amplified and acquires implementation details as development progress.

In functional decomposition technique which is most direct way of implementation, consisting of dividing the system into subprograms, transferring control between sub programs concentrate upon the algorithmic abstractions. The nature of the abstractions that may be conveniently achieved through the use of subroutines is limited.

The functional development methods suffer from the following limitations:

- do not effectively address data abstractions and information hiding;

- generally inadequate for problem domains with natural concurrency;

often not responsive to problem changes in space and hence the system can be fragile.

Massive restructuring by decomposing functionally is required if the requirement changes. In object-oriented approach focus is on identifying objects first from the application domain and then on filling procedures around them.

4.12 Analysis:

Analysis of the system, starts with the problem statement, which is to develop an object-oriented frontend to an existing GIS in order to enhance it as fully functional, flexible, application independent, user friendly GIS. Providing GUI is also part of the problem statement under consideration. These two can be combinedly grouped under the title "Object-Oriented gateway to GIS".

The analysis phase emphasizes building real word models, using object-oriented view of the world. The main task is to examine the needs from the perspective of the classes and objects found in the vocabulary of the problem domain. Apart from providing "object-oriented frontend to existing GIS" , by using object-oriented concepts, the system must take of the transparent line of separation between frontend and the underlying GIS. The two main objectives are to provide object-oriented concepts and GUI.

The OODB described previously is aimed at the description of object-oriented concepts that are incorporated in frontend. The high level abstraction, "Schema for an application" is the main objective, to achieve, which we decided to provide the user with the facility to create schema for each application.

The objective of providing GUI is to minimize the training overhead to users as many existing GISs force the users to undergo training in order to use the system, and make the system so user friendly, even for novice users. The primary function of GIS Viz., capture the data (spatially referenced attributes), analysis,

query and display are to be provided in an enhanced or object-oriented way.

In CIS can be made fully functional and flexible, if it allows the user to build the application with high level abstraction "schema".

While developing the system, the following tasks are to be considered and analyzed.

- 1). Incorporating object-oriented concepts discussed already (OODM/OODB)
- 2). Establishing GUI for user friendliness
- 3) . Frontend should act as the gateway to the underlying GIS (i.e./ capture, analysis, query and display in spatial context)
 - A) . Providing an effective communication between underlying GIS and frontend
- 5) . To have high level abstraction for an application by providing "building the schema" facility.

4.12.1. Schema:

It is the logical organization of entities and definition of their properties for a particular application. In object-oriented database systems, each entity in the real world is represented as an object.

The concept of 'class' is proposed to classify the objects, which have same characteristics or some abstraction level i.e., characteristics of objects are defined once per class but not for each object. Operations which can be applied to each object of a class are also defined in the class. In the schema classes that

are part of an application along with the hierarchical relationship among classes are defined. While defining the class, user has to give class description as follows.

NAME: Name of the class

CLASS RELATIONSHIP: The relationships with other classes are defined, to specify class hierarchy.

SUPER CLASSES: List of super classes

SUB CLASSES : List of sub classes

STRUCTURE : The aggregation hierarchy which represents the structure of objects in the class is specified.

REFERENCE NAME: The name which is to refer the component object

DOMAIN: The name of the class to which the component must belong

METHOD: The operations which are applicable for every object in the class are specified. Two kinds of methods are available here:

Daemon Method: This method is invoked when a specified object is created or modified. It can be specified in the form

Class name: Action

Operational method: This method is used to manipulate a class or object. It can be specified in the form

Message: Action

Fig 4.4 Class description

Classes are managed to form a hierarchy with reference to their abstraction level or their relations, and the descriptions are -inherited according to the hierarchy. In order to facilitate the abstraction constructs Viz., generalization, aggregation user is allowed to define two types of hierarchies in the schema. The first one is class hierarchy and the second one is aggregation hierarchy. Schema definitions consists of the definition of classes and class hierarchy.

Class hierarchy is defined as the super class-sub class relationship among the classes in the schema. It provides the user a way to generalize one or more classes as a single class. This aids user in specifying the relation between classes (abstraction at class level) i.e., all objects of a class are related to all objects of another class. The sub class inherits all the attributes and methods of its super class. Class may inherit properties from multiple super classes also. If there is any conflict among attributes of different super classes, the immediate super class attributes have higher precedence. Structural correctness of class hierarchy is to be maintained while user is defining class hierarchy.

We define a class hierarchy as correct if the graph representing the class hierarchy meets the following two conditions.

Condition - 1

Either the condition (a) or (b) holds for any pair of classes connected, (A,B), in the class hierarchy graph, but both of them are not satisfied.

- a) There exists one direct path from A to B
- b) If there are multiple paths from A to B, each passes through at least one node (class)

Condition - 2

For any class A, there is no cyclic path that starts from A and reaches A.

Aggregation hierarchy is defined in class description which is used to represent the complex objects (composite objects). Complex object classes are defined using references to other classes. The referencing instance variables (attributes or component objects) have non atomic values. Their domains are the sets of all instances of the referenced classes. The user may be given a set of primitive (int, real, char, boolean etc.) and non primitive (string, point, line, polygon etc.) predefined classes to use as references in the definition of complex object classes. However, user can also use his own defined classes as references in the description of class. It is abstraction at object level. The relation is defined between an object of a class and an object of another class. The relationship may be "part-of" or "contained-in" explained below is used to reflect the modification in CIS by allowing the user to modify the schema.

4.12.2 Schema modification [49]:

In this section, operations for modifying a schema and their effect are described. These operations are for class hierarchy, aggregation hierarchy, method, and as well as changing a class name.

There is no name conflict in the hierarchy:

No situation involving definition of names or methods which correspond to the same message in the multiple super classes. All of them are used commonly for schema definition and schema modification.

4.12.2.1 Operations for class hierarchy:

The following operations are applicable only for the class which classifies as the "root object" in the aggregation hierarchy.

Add a class:

This operation is used to add a class to existing schema. The newly added class will be, automatically, put in class hierarchy according to specification in class hierarchy in class description.

Delete a class:

Existing class can be deleted from the schema using this operation. Before deletion, the class is removed from the class hierarchy and all sub classes of class being deleted are redefined as sub classes to its super classes. However root classes in class hierarchy graph can not be removed.

Hierarchy Change:

This is used for changing super/sub class relationship between a class and its sub class. The conditions to be satisfied are given below.

- Adding super/sub relationship:

This is to add a super/sub relationship between two

existing classes. This operation is successful only if the resultant class hierarchy graph satisfies the conditions specified for structural correctness.

– Deleting super/sub relationship:

To delete a super/sub relationship between a class and its sub class this operation is used. In case a class with only one sub class, this operation can not be applied.

4.12.2.2. Operations for aggregation hierarchy: .

In order to change aggregation hierarchy, the following operations can be used. The user is allowed to modify the aggregation hierarchy defined in a class, but inherited structures of object defined in other classes cannot be modified. These will affect the structure of class in class description.

Add attribute:

This operation is used to add new attribute to an existing class. The attribute may have its domain either from primitive classes or user defined classes. In case an attribute is defined in with its domain as user defined class, then that class should not be super class/sub class to the class for which attribute is being added.

Delete attribute:

An existing attribute can be deleted simply by specifying the reference name of that attribute.

Change name:

The name of the schema or any class of schema can be changed. This can be done by specifying the current name and 'new name'. The new name should not be conflicted with schema name or other

class names.

Add/delete a method:

Operations are to be provided to add a method to a class or to delete any method from class. However modification of method is not permitted.

4.12.2.3 Propagation of changes:

As already explained user is allowed to modify schema. The system should take care of semantics and integrity of objects that have been created earlier to modification. This can be achieved by propagating changes to classes and/or objects.

Propagation of changes to class definition:

Whenever a class definition is changed it is to be propagated to other classes in the class hierarchy. These includes:

Addition of new instance variable (attribute); When an attribute is added to a class, then it should be propagated to all its sub classes, because the newly added attribute has to be inherited by the sub class. If the newly added attribute is defined in any of its super class then the corresponding inherited attribute is suppressed.

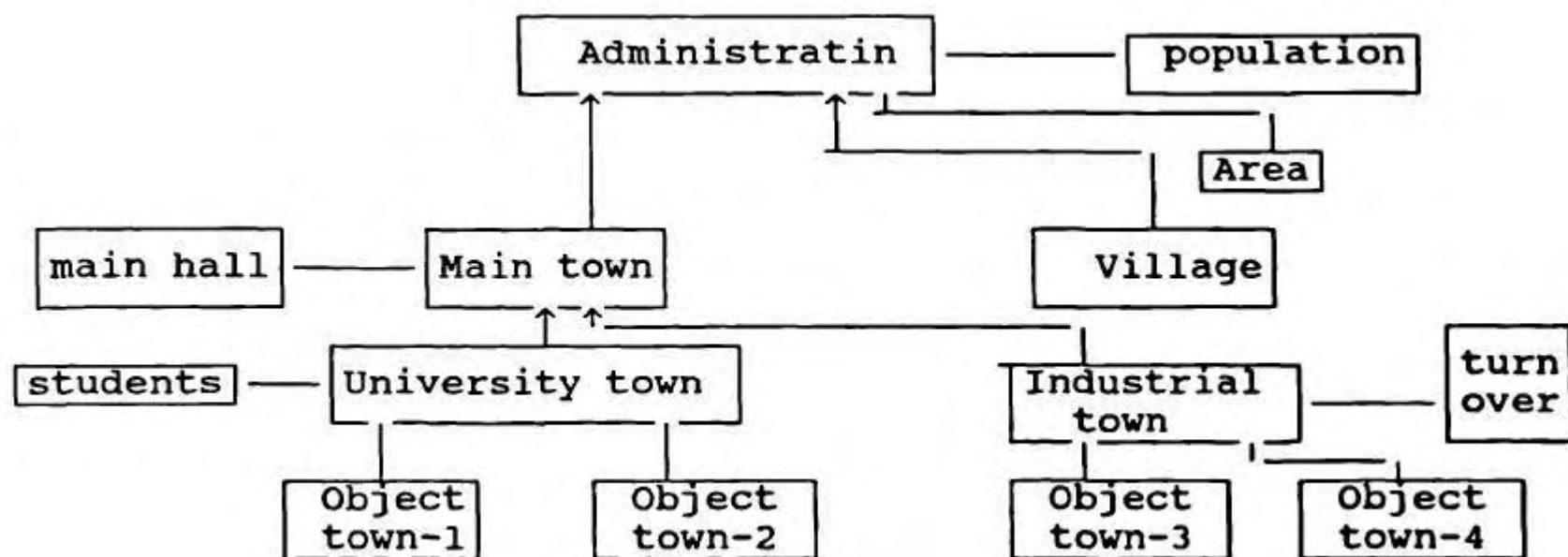
Deletion of an instance variable (attribute); If there is any attribute, with same name, defined in any of the super classes then it is to be inherited by the current class. Also messages have to be passed to all its sub classes so that they change their structure accordingly.

Addition/deletion of class; Here all the super classes and sub classes of current classes have to change their class relationship accordingly.

Propagation of changes to object instances:

Whenever schema is modified, changes are propagated to other classes first so that they change their structure and/or class hierarchy. After that all the classes involved in the change, either directly by user or propagation of change by other classes, are to propagate these changes to existing object instances. For instance, if a class is deleted, this is propagated to all of its sub classes. Then all the objects of all sub classes are to be modified by removing attribute values that are due to inheritance property from the class being deleted. This is also valid in case of changes in class hierarchy (i.e. super class/sub class relationship).

Example: Class hierarchy:



Method area() is defined to find out area of an object of administration, will be applied to each object of each descendent of class Administration.

4.12.3 In between object-oriented frontend and CIS:

The entire system can be viewed as two layers one over the other. The upper layer is the object-oriented frontend and the

lower layer is an existing CIS with former used for building application schema. The user interact with the frontend only. The frontend's responsibilities are availing CIS functions to user and providing object-oriented data modeling capability to user. The underlying CIS and frontend must have effective communication. The two communications are frontend requesting the underlying GIS for some task (eg: digitize, store, overlay), and the other is the communication/feedback from GIS to frontend. Acknowledgements and information regarding tasks performed by GIS as a consequence of frontend's request are communicated. So frontend is entirely dependent upon GIS for its functionality. The required task from frontend has to be converted into a request or operation which the underlying GIS knows. This task is accomplished using parser or interpreter.

4.13 System design:

The basic approach to solving the problem is selected in system design stage. While in analysis, the focus is on what needs to be done, in system design the overall structure and style are decided. The object-oriented data modeling capabilities is distributed among the four sub systems viz., schema manager, class manager, object manager, and system manager.

The frontend is under the control of system manager. System manger has the responsibility of correlating the GIS and schema (upper layer). The sub systems are described below.

4.13.1 Schema manager:

'Schema creating facility is provided by schema manager. The structure and behavior of schema are explained in object-oriented

analysis phase.

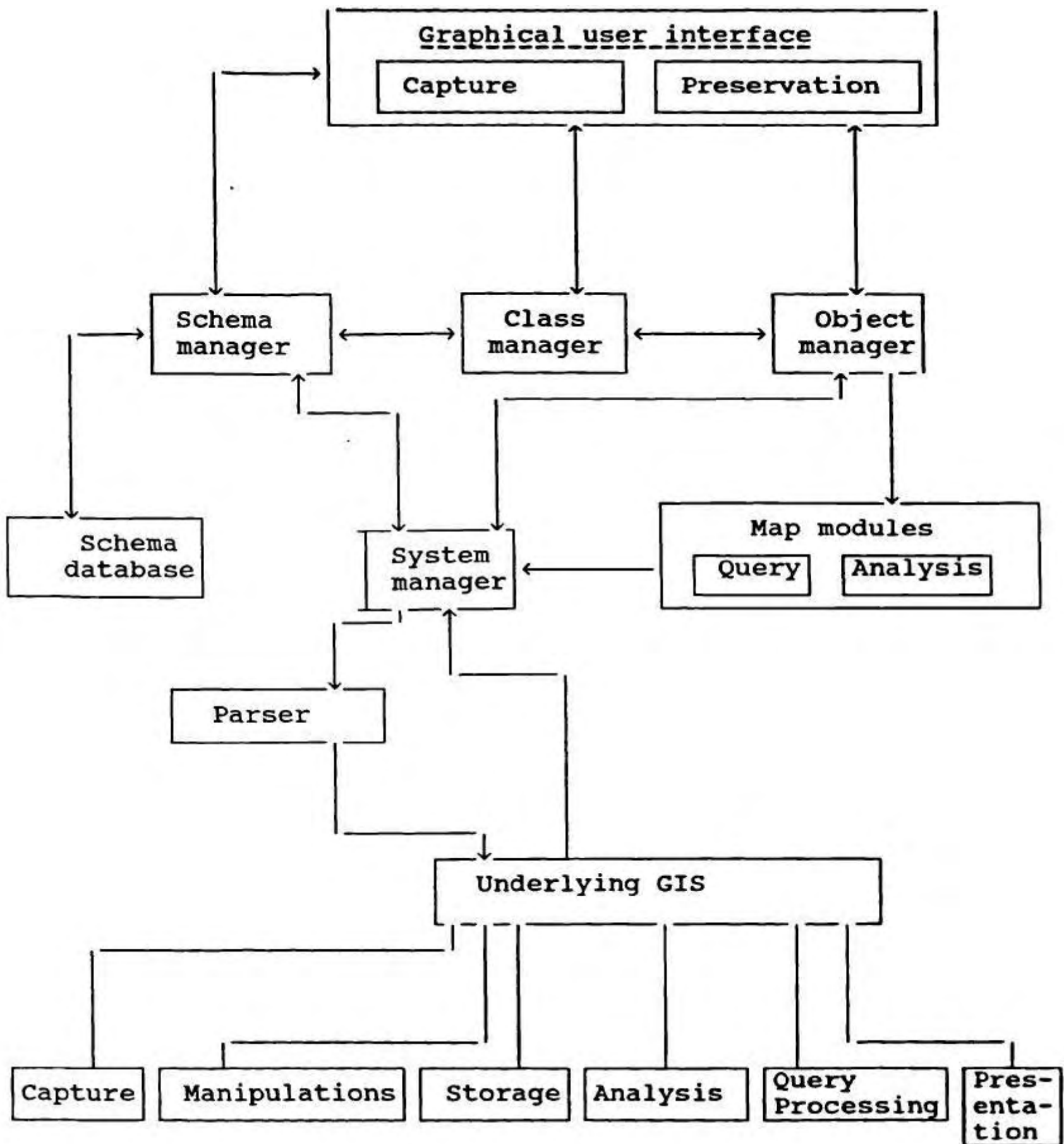


Fig 4.5 Object-Oriented gateway architecture

The major responsibilities of schema manager are given below.

Schema creation for each application:

The schema manager creates schema after checking for the correctness of all the information. The user is required to give schema name, structure of schema and class description of classes in order to create a schema.

Schema listing and view schema:

Existing schema is maintained by schema manager to know existing schema to enable users to create new schema. The schema structure (Partially or completely defined schema) and classes of that schema are shown to the user. Before modifying the existing schema, user may wish to view schema structure.

Resolving conflicts:

Conflicts like " duplication of names", 'referencing a class or object which is not pertaining to this schema are resolved by schema manager only.

Class hierarchy and aggregation hierarchy maintaining in a schema:

Schema manager maintains these two hierarchies in each schema. These two hierarchies can be defined or changed.

Modifying the schema, and propagation of schema changes to class manager:

Schema database has to be reflected with schema changes whenever the schema is modified. These changes are to be propagated to class manager and object manager.

Managing Schema database:

Schema database constitute schema structure, class descriptions and objects information of each class. Storing*

retrieving and modifying the schema information is taken care by schema manager.

4.13.2 Class manager:

Main function of class manager is to cooperate with schema manager in order to maintain the schema. The main tasks of class manager are listed below.

Class creation;

All classes defined in a schema are to be created. Using the class description, class is created by the class manager after checking its correctness.

View class or class list;

The class structure (its relationship on the class hierarchy), and attributes along with their domain class are shown to the user. Before any user modifies the class, user may wish to view the class structure. In a single schema, duplication of class names are prevented by the list of classes.

Modifying the class and propagation changes to object manager;

Adding an attribute, deleting an existing attribute, adding method and changing class hierarchy are included in class modification. After modification of the changes, they are propagated to object manager in order to change the objects of that class, already created.

Responding messages from schema manager:

Messages imply the changes that are propagated from schema manager. The schema manager requests class manager to change the occurrences of a class in all other classes that is being deleted from schema. Now the class definitions of all classes which are

having references to the deleted class are modified.

4.13.3 Object manager:

After the schema is created, the user then concentrates mainly on creating and manipulating the objects. The structure of the object i.e., class structure to which the object belongs, identifies, references to components and primitive attribute values are stored in a database maintained by frontend part. But the spatial data, pertaining to spatially referenced entities are stored by the underlying CIS. The references to such data are maintained by object manager. The important functions of object manager are as follows.

Creating objects:

Data from different input devices in different formats are captured for creation of an object. An object with different attributes is created by keying attribute values from key-board, or digitize maps, or specify data (remotely sensed, transformation of existing data) in some files. The underlying capture routines are to be invoked for this purpose by the object manager in coordination with systems manager which in turn sends requests to CIS.

Modification and retrieval of objects:

The changes incorporated in class definition or in class hierarchy are from class manger to object manger. The objects are suitably modified according to the changes made in the class. Objects can be explicitly modified by the user. The important task of object manger is to retrieve the objects. In

order to answer queries and to display objects, the required objects are retrieved by object manger and passed to either to display routines or system manager.

4.13.4 Transaction and map modules:

Transaction in CIS applications is totally different from that in ordinary database systems, because it includes operations over spatial objects. The analysis functions which the system provides are used to perform operations on spatially referenced entities. Map is a tool necessary to enable the user to perform transaction over the spatially referenced objects. A special kind of object class is a map which contains other geographical objects with respect to a reference to a system. Map objects can be derived by specifying map boundaries and objects that constitutes a map. Various analysis functions such as overlaying of maps, add maps, cut or zoom an existing map etc., which are provided by CIS, are used to manipulate maps to generate new maps.

So, transaction which is the main part of frontend relies on underlying CIS. Transaction consists of analysis and query processing. The underlying CIS is invoked to accomplish these two through system manger. The transaction is suitably fragmented into tasks and a request is sent to CIS for each task.

4.13.5 System manager:

System manager's responsibility is to coordinate all the managers and it has overall control on frontend. The requests from object manager, schema manager, and transaction part are

transferred to the CIS by Invoking parser. System manger process the reply or feedback given by the underlying CIS and the result is communicated to schema manager and object manager. The result may be a data file name or reference to some component object or output of some analysis function.

4.13.6 Parser :

The request delivered by system manager is converted by parser into requests or operations or commands which the underlying GIS can understand. So, design is independent of GIS.

4.14 Object design:

Object design is a mapping process, and keeps everything ready for implementation. This is called mapping process in the sense that, the ideas, strategies, and models developed earlier are mapped onto objects, data structures and algorithms. The analysis model is extended with specific implementation decisions and additional internal classes, attributes and operations by object design. During analysis, the discovered objects serve as the skeleton of the design, but object designer must choose among different ways to implement them, depending on the language selected (C can be used as it supports directly object-oriented programming) for implementation. The details of object design are given in the general object-oriented notion of class diagrams and object diagrams proposed by Booch [5],

The existence of classes and their relationships in the logical view of the system are shown in a class diagram. The class

structure of the system can be viewed in a single class diagram. The class diagram captures the structure of classes that form system architecture. The objects existence and their relationships in the logical design of the system are shown in an object diagram. Object diagrams are prototypical, in the sense that, each one represents the interactions or structural relationships that may occur among a given set of class instance, no matter what specially named objects participate in the actual implementation.

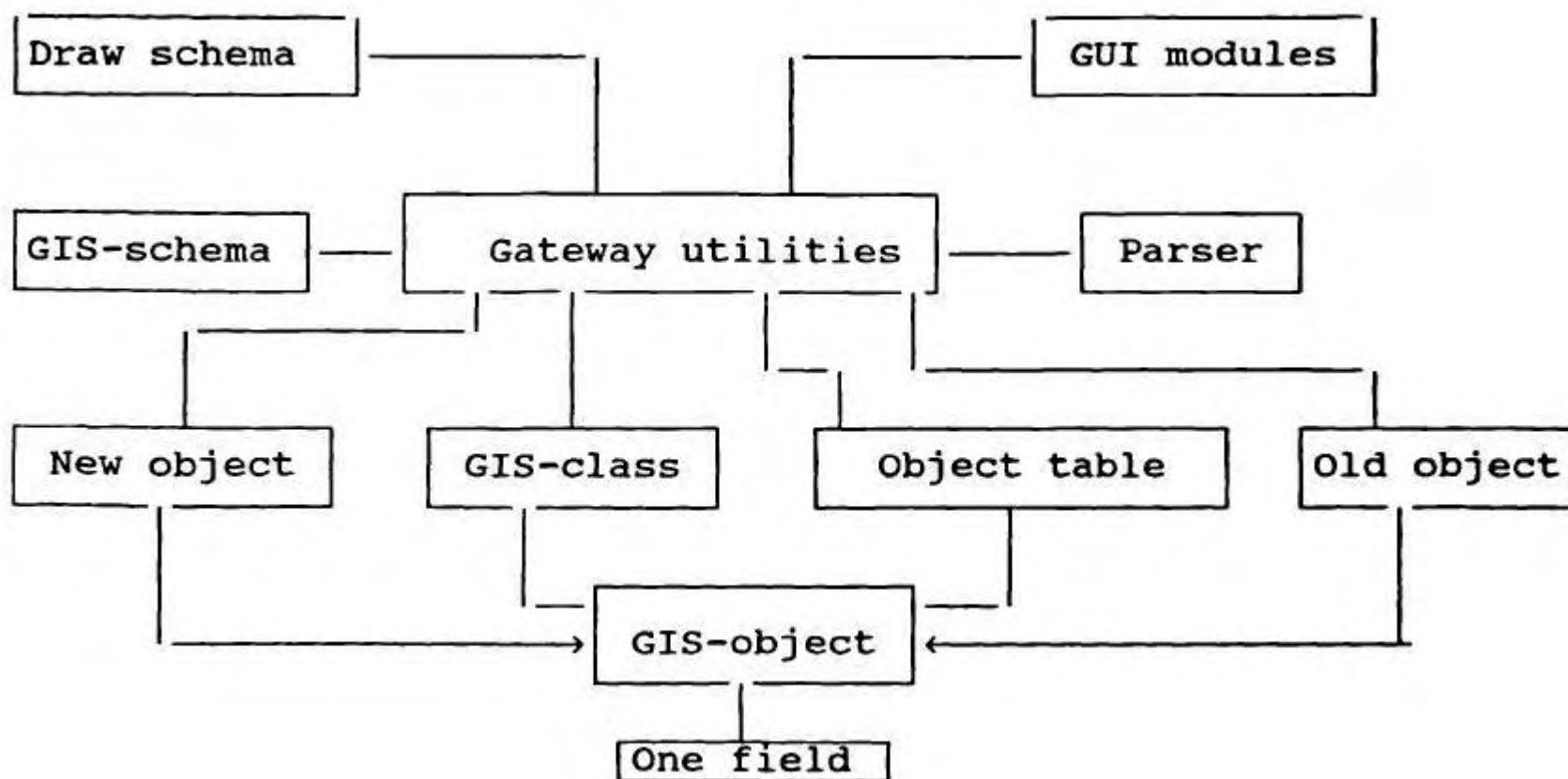


Fig 4.6 Class diagram

Class diagram is shown above and it shows the major classes and their interaction with other classes. The classes have vital contribution in the system development. Each class is intended to do specific task identified during system design.

CIS-schema, CIS-class, CIS-object are classes corresponding to schema manager, class manager, object manager respectively. The gateway control utilities which is responsible for integration of

classes in the system is the central component of class diagram. The whole system is monitored by this and it acts as a bridge between frontend and underlying CIS. The parser is invoiced whenever it has to communicate with the underlying CIS. This keeps track of user actions with GUI environment. The GUI part displays the graphical windows (widgets) requested by various classes on the screen and communicates the user actions to system manager or other classes. The object table is to maintain the objects and their offsets of each class (user defined class and not to be confused with classes designed) in each schema. The various object diagrams which correspond to the classes in the class diagram are given below.

Object diagrams:

The class design i.e its fields, member functions, functions used that are not members is clearly shown in object diagram. Just by looking at the object diagram which contains the sub class also, it becomes easy to understand the complete set of member functions of the sub classes. The object diagrams for the classes viz., GIS-schema, GIS-class, GIS-object and other supporting classes are described below.

1. GIS-schema:

The instances of GIS-class are present in GIS-schema. Each instance of GIS-schema corresponds to a schema defined for an application. This contains all the classes of that schema and schema database. To perform tasks like creating schema, modifying schema, deleting schema, various methods are shown in object diagram. It uses the routines 'hierarchy validation' for checking and maintaining the hierarchy class in a schema.

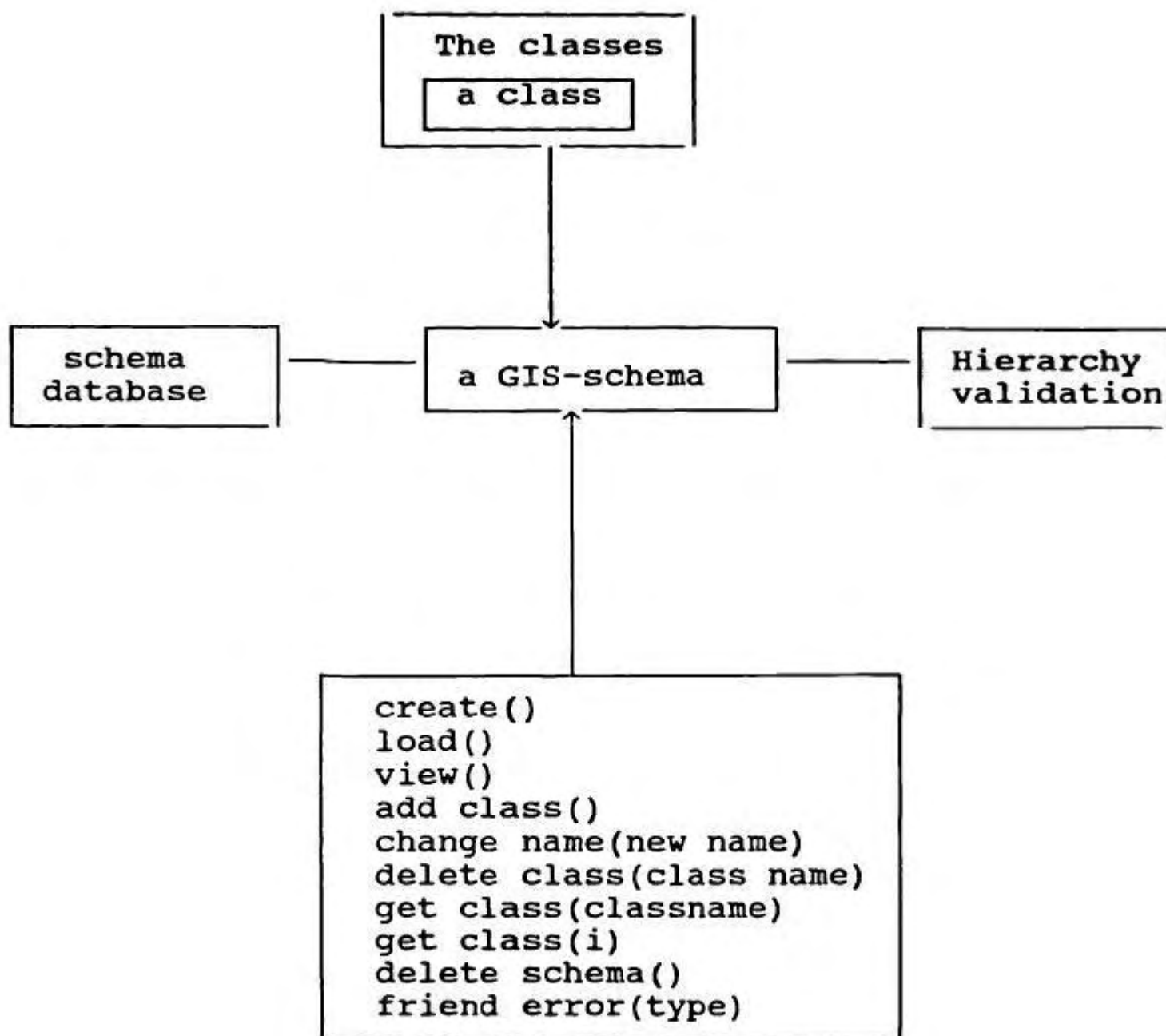


Fig 4.7: CIS-schema

2. CIS-class: The CIS-class contains the class structure in terms of its super classes, sub classes and attributes. The methods defined in CIS-class create class, view class, and modify class hierarchy, relationship.

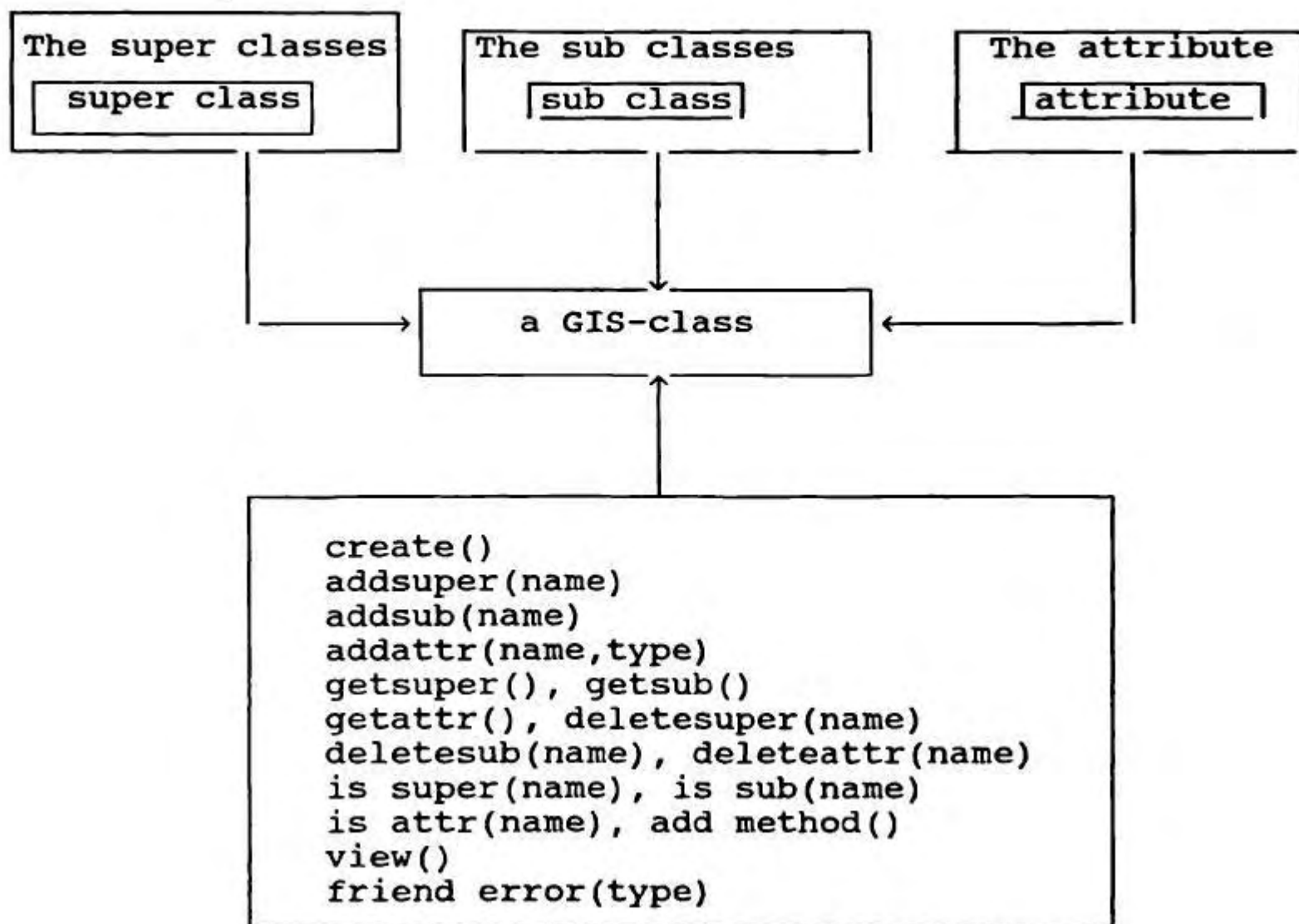


Fig 4.8 Object diagram: CIS-class

3. CIS-object:

CIS-Object maintains objects of each class. It responds to the propagation of changes in schema (i.e., whenever class definition is changed it modifies the objects) and it is part of object manager. New object, old object, the two classes inherit the structure of CIS-object and are responsible for creation of

the objects. CIS-object points (with next pointer) to another object in the schema. Display of objects on the screen in GUI environment is done by the member function view().

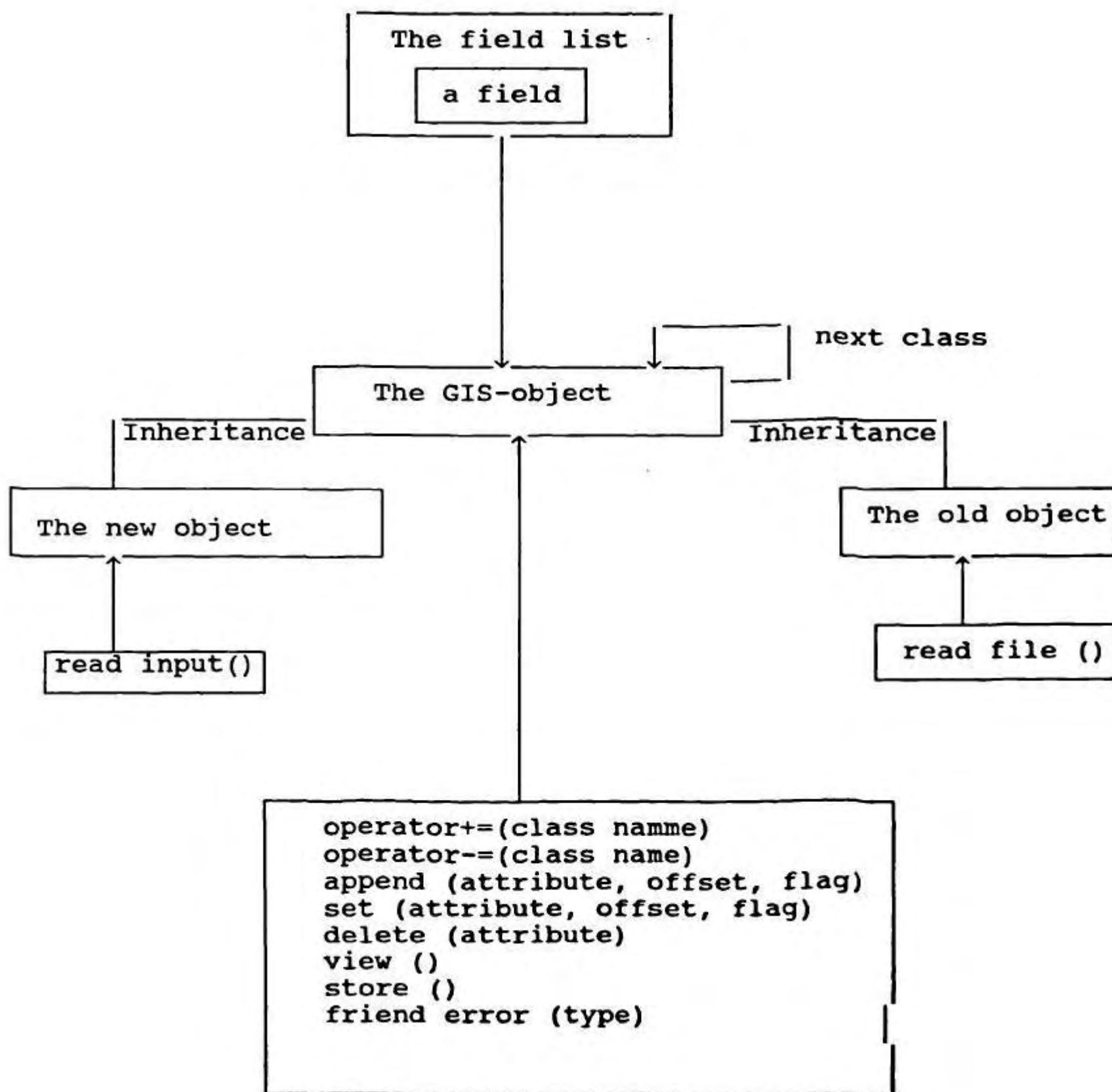


Fig 4.9 Object diagram: CIS-object

4. Object table:

It is the hash table maintained by the system manager and used by CIS-object. The list of objects and corresponding offsets pertaining to a class are available in object table. It also points to another object of object table so that hashing list of all classes of schema are maintained in linked list of object table. Append(), setoffset(), delete are methods defined to create, and modify the hash table. For knowing the existence of a particular object, the method exists() is used.

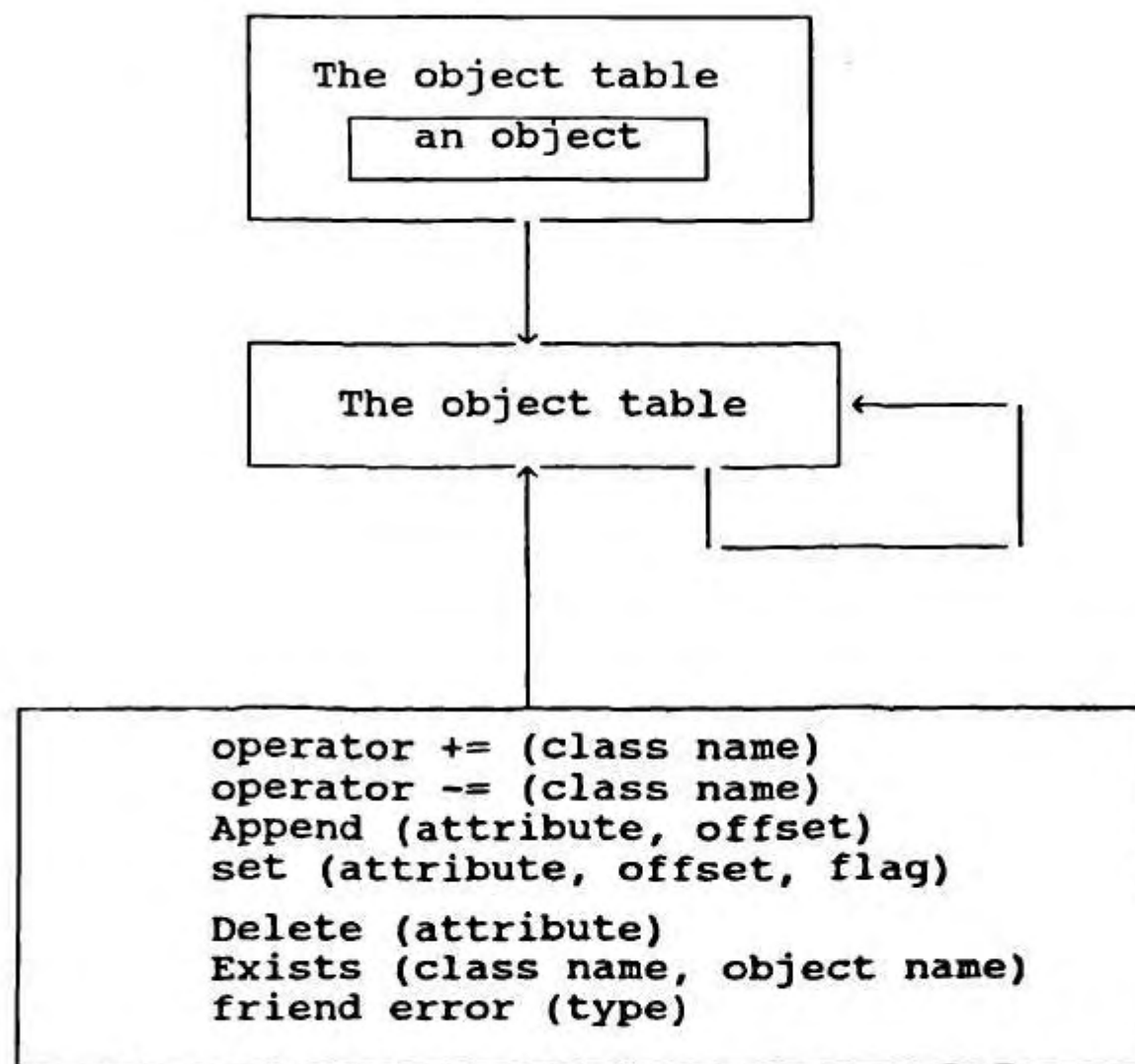


Fig 4.10 Object diagram - object table

5. Draw schema structure:

It provides graphical user view of a schema. When the user wants to view the loaded schema, the schema is shown as a directed acyclic graph (DAG). This structure is generated and maintained by this class. It also takes care of the subsequent modifications of the schema.

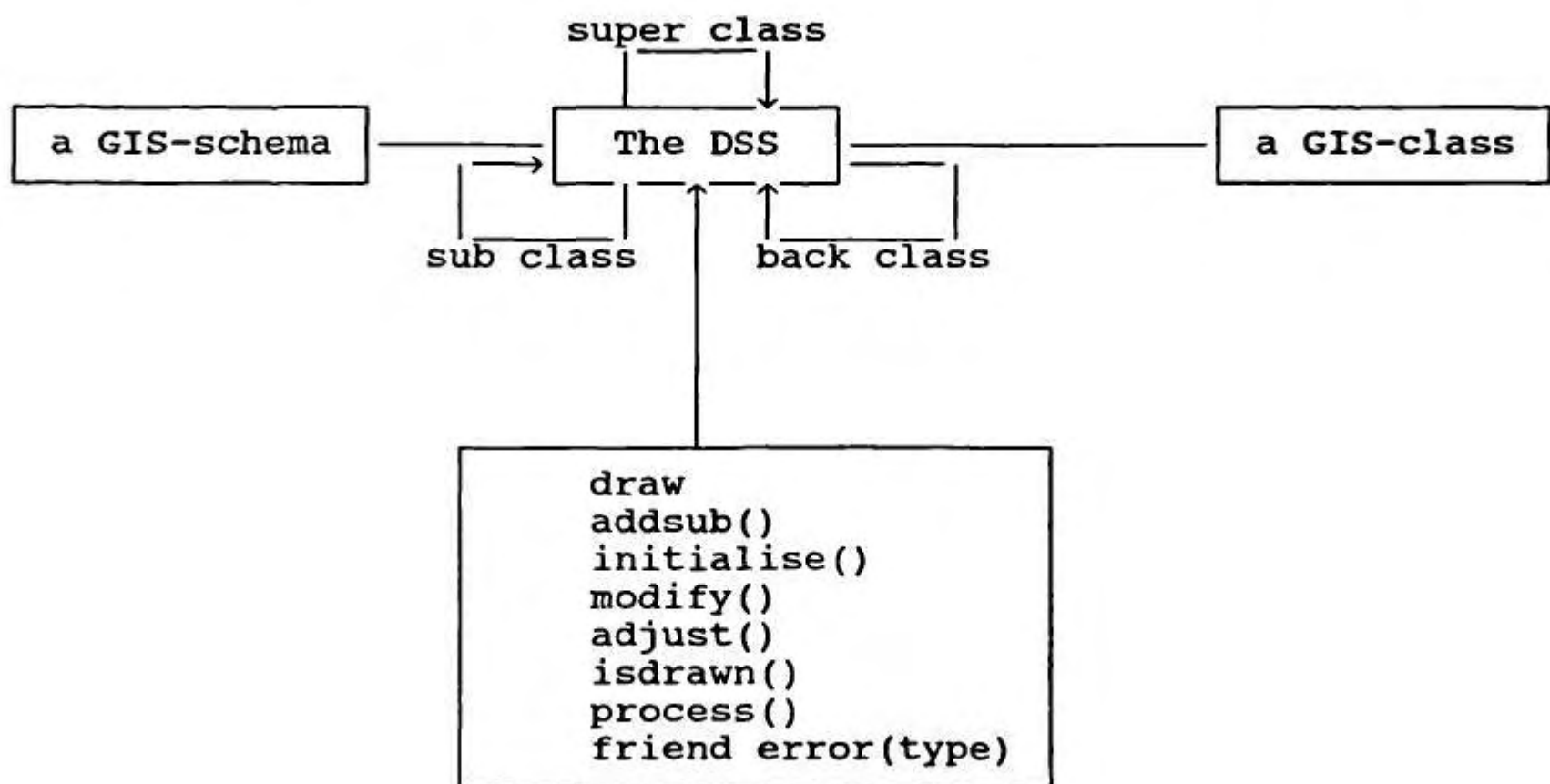


Fig 4.11 Object diagram:
Draw schema structure

Complete design is explained so far. We started with brief description of object-oriented design technique. Various phases of general object-oriented design are explained. Major tasks are identified during problem analysis phase. The proposed architecture is clearly described. At the end major classes, class diagram and object diagrams are identified and role of each class in the system are explained.

CHAPTER-V

5.0 Conclusions and future directions:

In chapter-I, we have discussed the three computer applications under spatial information section, viz., first generation applications involving numerical computing, second generation applications involving commercial data processing and third generation dominated by computing with geometric and pictorial objects resulted in development of various data structures such as arrays, list structures and multi-key access structures. Then characteristics of spatial data were explained. They are objects being accessed by unique number and representation of an object is an integral part of data structure. Geographical Information System (GIS) and a spatial data base system which is a special kind of GIS are described along with components of GIS viz., data collection, data management, retrieval and transformation, display. The GIS features are dealt which include use of primary transformation tools which include simple spatial analysis tools and use of compound information tools which make use of mathematical models or expert system. Need for building expert system and expertdatabase system along with its categories are explained. The necessity to use reasoning which includes reasoning using logic, reasoning using rules, reasoning using frames and semantic nets, nonmonotonic reasoning, reasoning under uncertainty, evidential reasoning are discussed. The advantages of use of Artificial Intelligence (AI) in building knowledgebase system for GIS, with AI playing predominant role in deriving a decision support system with GIS acting as a spatial

database i.e., a special kind of CIS to store spatial and non-spatial data has motivated in solving problems viz., reasoning using multiple sources of information to identify features in a target scene, image to map registration problem and building object-oriented gateway to CIS in this dissertation is explained. The aim of the thesis and present work and organisation are described in later sections of chapter-I.

The first objective of this dissertation i.e., developing an evidential reasoning technique to reveal the information content of a target scene using multiple sources of information is described in chapter-II. Introduction to spatial reasoning, spatial reasoning as described by different authors, and earlier works on spatial reasoning viz., Algebraic approach to spatial reasoning, fuzzy logic based expert system to CIS, a logical framework are explained. Various reasoning techniques uses single set of information which is clear and unambiguous and single line of reasoning is adopted in order to come to conclusions. In order to identify features like mountains, plannar surfaces etc., in a target scene, multiple sources of evidence constituting evidence domain, involving basic elements like tone/colur, texture, shadow etc., incase of satellite image have to be made use of to develop knowledgebase or gallery which contains frames along with compatibility relationships among them. The various evidential operators gisting, summarisation etc., are to be applied depending on the way the gallery is made use of, to move through these frames where target questions can be answered to reveal the

information content to know about various features. So, need to use evidential reasoning is described.

The various terms used viz., frame of discernment, basic probability assignment, belief function, focal element are explained. Implementation details are also described. The framework for implementing includes: specifying frame of discernment, arriving at compatibility relations among them, assigning quantitative belief, reasoning by moving through the gallery by making use of various evidential operators to reveal the information content of a target scene is explained next. The major steps in evidential reasoning are: building knowledgebase or gallery and reasoning to reveal the information content of a target scene and these are described in subsequent sections of chapter-II.

The basic elements tone/color, texture, shadow, context, pattern, shape, size, and association are used in identifying different features on satellite images. The various evidential operators viz., fusion, discounting, translation, summarisation, gisting and interpretation are used depending on the way the gallery is made use of, and are explained in detail. While making use of fusion operator various algorithms viz., Dempster's rule, optimistic rule of combination, Bayesian approximation, Harriett's technique for singleton hypotheses, Harriett's technique applied to partition, Gordon and Shortliffe algorithm, Shafer and Logan's algorithm for hierarchical evidence are to be used depending on

the way the evidence domain and evidences are used are explained. A case study which uses geologists knowledge, to identify geomorphological features using evidential reasoning in topomaps, satellite images is discussed in length in later sections of chapter-II. Various geomorphological features to identify are mountains, inselbergs/born hardts, irregular hills, domal hills etc. . The features in topomaps such as closely spaced contour lines, widely spaced contour lines etc., along with corresponding hypotheses are described. In similar lines features with corresponding hypotheses of geomorphological features on satellite images (FCC and Band-4) are explained. The information gathered from experts is structured properly to build gallery or knowledgebase.

The frames include the list of features to identify drainage, area etc., with compatibility relations like drainage-features, area-features etc.. Alternate way is described in which frames are described with each frame for one feature such as sand, sand dunes, desert etc., and are used to identify geomorphological features in the target scene. Future directions to evidential reasoning are inclusion of additional features to identify like sub ways, canals etc.. So, evidential reasoning can be very effectively and efficiently applied to identify various features in topomaps and satellite images, using multiple sources of information which are imprecise, and uncertain in nature.

Chapter-III explains the need for image to map registration which includes map updation, image interpretation, and data acquisition. Advantage of reasoning using logic is explained. The portion of a map is to be registered with the same portion of a relevant satellite image so that the changes in environment are suitably carried out to various spatial objects in the map. In this process image interpretation is also achieved. Spatial objects viz., chains and regions are considered. The spatial objects and spatial relationship details are obtained by making use of coordinates describing each spatial object or by the digitization process, and using various logics respectively. Tee, chi, bounds etc., relationships are obtained in this process. Axiom set is built using 'PROLOG' which contains spatial objects, spatial relationships, and rules which govern real world scene such as rivers do not cross each other, rivers flow into other rivers or shores etc.. Implementation steps are described. A 'C' program is used to arrive at spatial objects, and spatial relationship details. All feasible solutions proved 'true' in the knowledgebase i.e., 'PROLOG' programme are updated into a dynamic database. As selective back tracking was not possible in 'TURBO-PROLOG', a 'C' programme is written to generate possible interpretations one by one to be checked in 'PROLOG' code. Due to deficiencies in 'TURBO-PROLOG', certain modules are written using 'C' language. The interpreted or spatial reasoned details (those feasible solutions proved 'TRUE' in 'PROLOG' code) are saved in a dynamic database to be available to CIS user for solving image to map registration problem. Future directions are. including more

image primitives such as railway lines, culverts, national high ways, points etc., inclusion of more attributes such as length, adjacent to which image primitive etc.. More image primitives and attributes results in arriving more spatial relationships like points inside a region, region adjacent to another region etc., will further enhance the capability of the developed system to interpret more features in a better way. Object-Oriented modelling advantages (explained in chapter-IV) can be extended to build the knowledgebase.

Chapter-IV discusses about object-oriented gateway to GIS. Necessity to have object-oriented frontend and graphical user interface (GUI), are described. The various terms used are, object, object identity, class, complex object, encapsulation, hierarchy and inheritance, binding, polymorphism, extensibility, and object-oriented DBMS are explained. In order to support complex objects, modelling structured objects, customised data types and interfaces, object-oriented GIS (OOGIS) has to be developed. The steps used for this purpose are described. The comparisons of OOGIS vs GIS viz., generic concepts, adhoc query facility, concurrency, distribution, cooperative working environment, graphical user interface are explained. The aim of object-oriented gateway includes providing object-oriented frontend to underlying GIS and GUI are described.

The features of traditional database system viz., persistence, sharing, query language, concurrency, transaction.

features of semantic data models such as aggregation and generalisation, concepts of object-oriented programming viz., complex object, object identity, class and methods, encapsulation, inheritance, extensibility are conveniently merged and an object-oriented data model (OODM) is developed. These are dealt in detail in subsequent sections of chapter-IV. Version control, equivalent object etc., which are additional features of OODM are explained. Object-Oriented database can be obtained by extending the relational system to support the concepts of objects, and extending object-oriented programming language to include persistence, sharing etc., of database features. These aspects are covered in later section. A data model is structurally object-orientation if it supports construction of composite objects and is behaviorally object-oriented if it supports user defined types and definition of operators (methods) that are applied to these types. Modelling real world phenomena, different approaches to OOGIS viz., extension of OODB functionality to existing CIS, extension of an existing DBMS by geometric and geographical functionality are described.

The approach followed for developing an object-oriented gateway is described in next section of Chapter-IV. It includes providing an object-oriented frontend to any CIS. For spatial data analysis we will rely on underlying CIS. Object-Oriented frontend uses object-oriented data modelling concepts which the underlying CIS lacks. The communication between existing CIS and object-oriented frontend is achieved by a parser. All the

object-oriented features (explained above) are augmented to CIS by the frontend.

Object-Oriented gateway design aspects viz., analysis, system design, object design, implementation are explained in latter sections of chapter-IV. The tasks of analysis phase are object-oriented concepts inclusion, establishing GUI, providing communication between CIS and frontend via parser, high level abstraction for an application by providing 'schema building' facility. Schema building, modifications to schema (add a class, delete a class, affecting hierarchy changes, add attribute, change name, add/delete method, propagation of changes to class and objects) are explained. System design describes the approach to solve the problem. The over all structure is described. The four sub systems are schema manager, class manager, object manager, and system manager.

The object-oriented frontend is under the control of system manager. Schema creation, schema listing and schema viewing are under the control of schema manager. Class manager is used for class creation, view and list class, modifying and propagation of changes to object manager and responding to messages from schema manager. Object manager is used for object creation, modification, and retrieval of objects. Transaction and map modules uses underlying CIS capabilities viz., manipulation, analysis, storage, query processing. System manager process requests from object manager, schema manager, and transaction part and are

transferred through parser to CIS. The replies from CIS are communicated to object manager, and schema manager. The object diagrams of CIS-schema, CIS-class, CIS-object are also explained in chapter-IV. The future directions to object-oriented gateway to CIS are incorporating equivalent object concepts, i.e., to keep track of different representations of same object and building client-server model. In a client-server model a single server work as kernel which is rich in functionality and will be used by different clients on client GISs placed wide apart.

R E F E R E N C E S

[1] A. Alhzobaidie, and J.B. Crimson, Expert, systems and database systems, how can they serve each other?. Proceedings of Expert Database Systems, 1988.

[2] Gustaro Alonso, Amr Ei Abbadi, Cooperative modelling in applied geographic research, International Journal of Intelligent and Cooperative information systems, Vol:3, No:1, 83-102, 1994.

[3] J.A. Barnett, Computational methods for a mathematical theory of evidence, In Proceedings Seventh International Joint Conference on Artificial Intelligence, Vancouver, BC, 868-875, 1981.

[4] Elisa Bertino, Lorenzo Martino, Object-Oriented database management systems: concepts and issues, IEEE Computer, Vol:24, No:4, 33-48, 1991.

[5] Grady Booch, Object-Oriented development, IEEE Transactions on Software Engineering, Vol: SE-12, No: 2, 1986.

[6] G. Bossu, and P. Siegal, Saturation, nonmonotonic reasoning and the closed world assumption, Artificial Intelligence, 25, 13-63, 1985.

[7] B.G. Buchanan, and E. Shortliffe, Rule base expert systems. The MYCIN experiments of the Stanford heuristic programming project, reading, Mass: Addison-Wesley.

[8] P.A. Burrough, Development of intelligent geographical information systems. International Journal of Geographical Information Systems, vol:6, No:1, 1-11, 1992.

[9] P.A. Burrough, Are GIS data structures too simple minded?, Computers and Geosciences, Vol: 18, No:4, 395-400, 1992.

- [10] Dempsteri P. Arthur, A generalisation of Bayesian inference. Journal of the Royal Statistical Society 30 (series B), 205-247, 1968.
- [11] Andrew U. Frank, Spatial concepts geometric data models and geometric data structures, Computers and Geosciences, vol:18, No:4, 409-417, 1992.
- [12] Herve Gallire, and Jean Nicolas, Foundations of knowledgebase management, Contributions from Logic Databases and Artificial Intelligence Applications, 119-130, 1989.
- [13] T.D. Garvey, J.D. Lowrance, and M.A. Fischler, An inference technique for integrating knowledge from disperate Sources, In: Proceedings 1981 International Joint Conference on Artificial intelligence, 319-325, 1981.
- [14] Michael F. Good child, Geographical data modelling, Computers and Geosciences, Vol: 18, No: 4, 401-408, 1992.
- [15] J. Gordon, and E.H. Shortliffe, A method for managing evidential reasoning in a hierarchical hypothesis space, Artificial Intelligence, Vol:26, 323-357, 1985.
- [16] Oliver Gunther, Johannes Lamberts, Object-Oriented techniques for the management of geographic and environment data, The Computer Journal, Vol:37, No:1, 1994.
- [17] Steven J. Henkind, and Molcolm C. Harrison, An analysis of four uncertainty calculi, IEEE Transactions on Systems, Man, and Cybernatics, vol: 18, No: 5, 700-713, 1988.
- [18] John R. Herring, TIGRIS: A data model for an object-oriented Geographic information system. Computers and Geosciences, vol:18, No:4, 443-452, 1992.
- [19] Fred Holroyd, Sarah B.M. Bell, Raster GIS: models for raster encoding, Computers and Geosciences, Vol: 18, No:4, 419-426, 1992.
- [20] A.R. Hurson, Simon H. Pakzad, Jia-bing cheng, Object-Oriented database management systems: Evolution and performance issues.

IEEE Computer, Vol:26, No:2, 48-61, 1993.

[21] P. Jalote, Functional refinement and nested objects for object-oriented design, IEEE Transactions on Software Engineering, Vol: 15, No: 3, 264-270, 1989.

[22] C.P. Kofran, AI and CIS connections: Views from industry and field, PGIS/LIS-89 Proc, American Congress in Surveying and Mapping, Bethesda, Md, 1989.

[23] K. Konolige, On the relation between default theories and autoepistemic logic, SRI International Artificial Intelligence Center Technical Report, Palo Alto.

[24] R. Kowalski, Algorithm - Logic + control , Comm ACM, Vol:22, No:7, 424-436, 1979.

[25] Henry E. Kyburg Jr, Bayesian and non bayesian evidential updating, Artificial Intelligence, Vol:31, 271-293, 1987.

[26] Y. Leung, and K.S. Leung, An intelligent expert system shell for knowledge-based Geographic information systems: The Tools, Geographical Information systems, Vol:7, No:3, 201-213, 1993.

[27] Pawn Lingras, and S.K.Wang, An optimistic rule for accumulation of evidence, Methodologies for Intelligent Systems.

[28] J.D. Lowrance, and T.D. Garvey, Evidential reasoning: A developing concept, In: Proceedings of the IEEE International Conference on Cybernetics and Society, 6-9, 1982.

[29] John D. Lowrance, Thomas D. Garvey, Thomas K. Strat, A framework for evidential reasoning systems, In: Proceedings of the Fifth National Conference on AI. AAAI-86, Philadelphia , Pennsylvania, 896-901, 1986.

[30] David M. Mekeown jr, The role of Artificial intelligence in the integration of remotely sensed data with Geographic information systems, IEEE Transactions on Geoscience and Remote Sensing, vol: GE 25, No:3, 329-345, 1987.

- [31] Peter Milone, Scott Mitton, John L. Smith, Geographical object-oriented databases - a case study, *International Journal of GIS*, Vol: 7, No:1, 39-55, 1993.
- [32] Judea Pearl, Fusion, propagation, and structuring in belief networks, *Artificial Intelligence*, Vol:29, No:3, 241-288, 1986.
- [33] Judea Pearl, Evidential reasoning under uncertainty. In: Howard E. Shrobe, Editor, *Exploring Artificial Intelligence*, 381-418, Morgan Kaufmann, San Mateo, CA, 1988.
- [34] J. Pearl, On evidential reasoning in a hierarchy of hypotheses, *Artificial Intelligence*, Vol:28, 9-15, 1986.
- [35] David Poole, A methodology for using a default and abductive reasoning system, *International Journal of Intelligence Systems*, Vol:5, No:5, 521-548, 1990.
- [36] A.K. Pradhan, K. Tripathi, A conceptual framework of spatial and non-spatial database development of an efficient GIS, *International Journal of Intelligent cooperative information systems*.
- [37] Davis Randall, and Lenat, B. Douglas, *Knowledge-based systems in Artificial intelligence*, McGraw-hill, Newyork, 1982.
- [38] Jonathan F. Rapp, David J. Magnix, Design models and functionality in GIS, *Computers and Geosciences*, Vol:18, No:4, 387-394, 1992.
- [39] R. Reiter, and A.K. Mackworth, A logical framework for depiction and image interpretation. *Artificial Intelligence*, Vol: 41, 125-155, 1990.
- [40] David Rind, A GIS research agenda. *International Journal on GIS*, Vol:2, No:1, 23-28, 1988.
- [41] Stuart A. Roberts, Mark N. Gahegm, An intelligent object-oriented Geographical information system. *International Journal on GIS*, Vol:2, No:2, 101-110.

- [42] V.D. Robinson, A.V. Franik, and M.A. Blaze, Expert system and CIS: review and prospects, Expert system in engineering, 201-213, 1988.
- [43] Glenn Shafer, and Roger Logan, Implementing Dempster's rule for hierarchical evidence, Artificial Intelligence, Vol:33,271-298, 1989.
- [44] G. Shafer, Hierarchical evidence, In : Proceedings of the Second Conference on Artificial Intelligence Applications, Miami beach, FL, IEEE Computer Society press, 16-25, 1985.
- [45] G. Shafer, The combination of evidence, International Journal of Intelligent systems, Vol:1, 155-179, 1986.
- [46] G.A. Shafer, A mathematical theory of evidence, Princeton University Press, New Jersey, 1976.
- [47] T.R. Smith, and K.R. Park, Algebraic approach to spatial reasoning, Geographical Information Systems, vol:6, No:3, 177-192, 1992.
- [48] Thomas M. Strat, Continuous belief functions for evidential reasoning, Proceedings, AAAI-84, Auston, Texas.
- [49] Atsno Toshiba, M. Hira kawa, I. Tadao, Schema management in object-oriented database systems, Visual Database Systems.
- [50] Frans Voorbrack, A computationally efficient approximation of Dempster Shafer theory, International Journal of Man Machine studies, Vol:30, 525-536, 1989.
- [51] L.p. Wesley, and A.R. Hanson, The use of an evidential based model for representing knowledge and reasoning about images in the vision system, IEEE 1982 Proceedings International Conference on Cybernatics and Society, Seattle, WA, 14-25, 1982.
- [52] Michael F. Worboys et al, Object-Oriented data modelling for spatial databases, International Journal of CIS, Vol:4, No:4, 369-383, 1990.

[53] Liping Zhao, S.A. Roberts, An object-oriented data model for database modelling, implementation and access. The Computer Journal, Vol: 31, No: 2, 116-124, 1988.

[54] Ch. Nageshwar Rao, Expert system shell for knowledge-based CIS, M.Tech project report, University of Hyderabad, Hyderabad, 1995.

[55] Paurushasp N. Mistry, and G.A.B. Ragalakshmi, Knowledge-based CIS - Image interpretation using logic, MCA project report. University of Hyderabad, Hyderabad, 1991.

[56] G. Chandra shekhar, and R. Sridhar, Object-Oriented gateway to CIS, M.Tech project report, University of Hyderabad, Hyderabad, 1995.

papers published by authors:

[1] Parushasp N. Mistry, G.A.B. Raga lakshmi, Arun K. Pujari, c. Ravindra kumar, Logical interpretation of a map, INCA Conference, INDIA, 1992.

[2] C. Ravindra Kumar, Arun K. Pujari, P.G. Reddy, Contingent knowledge incorporation to interpret an image using logic. Conference on Remote Sensing Applications and CIS, Recent Trends, INDIA, 465-469, 1992.

[3] Arun K. Pujari, C.Ravindra kumar, AI techniques for advanced CIS, CIS Today, 3-6, 1993.