A COMPUTATIONAL MODEL OF VISUAL INTERPRETATION

By

Kaustuv Kanti De Biswas

Bachelor of Architecture
Jadavpur University, India.
2003

SUBMITTED TO THE DEPARTMENT OF ARCHITECTURE IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE IN ARCHITECTURE STUDIES
AT THE
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

MAY 2006

Signature of Author: ____
Department of Architecture
May 25, 2006

Certified by: ____
Terry Knight
Professor of Architecture
Thesis Supervisor

Accepted by: ____
Julian Beinart
Professor of Architecture
Chairman, Departmental Committee of Graduate Students

A COMPUTATIONAL MODEL OF VISUAL INTERPRETATION

By

Kaustuv Kanti De Biswas

Submitted to the Department of Architecture, May 2006, in partial fulfillment of the requirements for the degree of Master of Science in Architecture Studies at the Massachusetts Institute of Technology

Abstract

From the very early phases of design conception, designers use sketches as a powerful design tool. Sketches are however ambiguous. Meanings are associated on fly as the designer 'comes up' with certain ideas while working with it. There is no hierarchy in a sketch. In fact 'structure' is established only after meanings are applied to the sketch. However even in such structurally and conceptually fluid territory, the designer solves most of his design problems and very often comes to quick resolutions. To understand this fascinating tool and how the designer interacts with it, we need to understand how we visually interpret sketches.

The process of design is also a reflective act. The designer keeps changing his perspectives and focus based on the unexpected opportunities that emerge from such reflection. Computational systems used today in design exploration are not capable of doing so. On the contrary these systems model the world in a very rigid structured way and cannot produce design ideas beyond what their preset description anticipates. From this perspective there is no novelty, or surprise, in such systems.

As a step forward, this thesis proposes the following:

1. Visual Schemas as procedural units of visual memory. They schematically store real world knowledge ('courtyard') and form the basis for interpretation.

2. Separation of Shape and Visual Concepts. This thesis suggests that shapes are flat and abstract collection of parts, while visual concepts are subjective and hierarchic ideas, which are formed from the shapes through interpretation.

A LISP machine is presented as a basic computational framework for implementing and establishing the model that is proposed. It observes a relatively simple architectural sketch, interprets it reflectively through the activation of potential, alternative contexts, and then gives a collection of concepts that it manages to 'see' in the sketch.

Thesis Supervisor: Terry Knight
Title: Professor of Architecture

Thesis Reader: George Stiny
Title: Professor of Architecture

Thesis Reader: Patrick Henry Winston
Title: Ford Professor of Artificial Intelligence and Computer Science

Acknowledgments

First of all, I would like to thank my thesis supervisor, Professor Terry Knight, for her valuable advice, support and constant encouragement. Her constant criticisms and reviews gave me the conceptual clarity. I owe a significant lot to my readers Professor George Stiny and Professor Patrick Winston. George introduced me to 'Shapes', 'Seeing' and 'Computation' in general. It was in his Shape Grammars class that I first envisioned this thesis. Patrick's optimism, encouragement and insights were crucial to this work.

I am thankful to Professor Marvin Minsky, whose 'Society of Mind' class opened up my mind and gave me the conceptual tools to start this thesis; Professor Takehiko Nagakura for his invaluable reviews and technical help.

I would like to thank my friends Jimmy Shen, Axel Kilian, Neri Oxman, Onur Gun, Sergio Araya and Daniel Cardoso for patiently enduring my explanations. Their reviews and comments were extremely helpful.

And of course, I owe my ability to write this thesis at all, to my family whose love and encouragement has been my cornerstone.

Contents

1. Introduction ……………………………………………………………... 5

2. Background – Revisiting Some Powerful Ideas …………………..8
2.1 George Stiny – No Structure Computation
2.2 Marvin Minsky – Distributed Cognition
2.3 Pegor Papazian – Principle of Opportunism in Design
2.4 Shimon Ullman – Visual Routines

3. Theoretical Basis for the Model …………………………………..15
3.1 Abstraction and Conception Spaces
3.2 Shapes
3.3 Concepts
3.4 Schemas
3.5 Passive Concepts
3.6 Cognitive Filter
3.8 Perceptual Filter
3.8 Memory and Persona

4. Visual Interpretation ……………………………………….…..25
4.1 The model
4.2 Finding emergent courtyards in a Site-Plan

5. LISP Machine (TRIGGER) ……………………………….…..32
5.1 Principles
5.2 Parts
5.3 Example Run

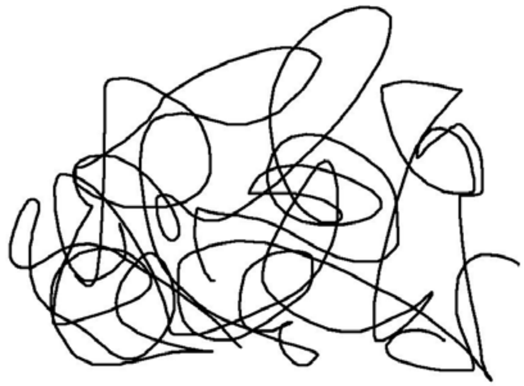
6. Discussion……………………………………………………………42
6.1 TRIGGER as a design exploration tool
6.2 Sketches with performance criteria
6.3 Shape Grammar Implementation
6.4 Schema Recognition versus Pattern Recognition

7. Contribution ……………………………………………………………46

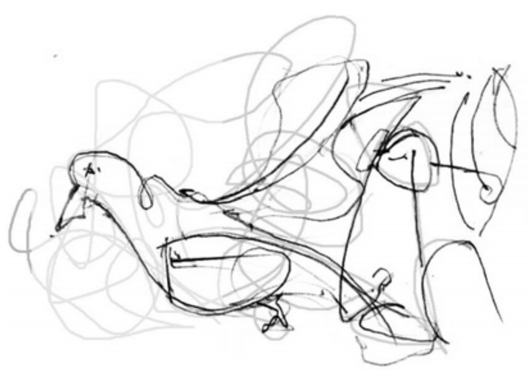
8. Conclusion……………………………………………………………46

9. Bibliography…………………………………………………………48

1. Introduction

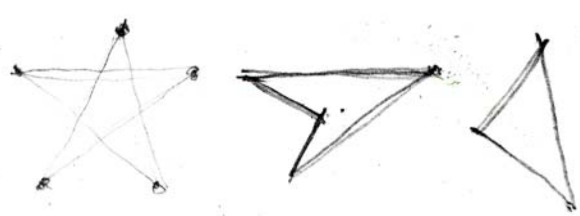


As a kid, I used to enjoy scribbling sketches, without really having any idea about what I was doing. After filling up a page, I would intently gaze at them and within the intense confusion of the scribble, meaningful forms would slowly appear and fascinate me.



Although distorted and incomplete, these forms - faces, animals, objects, would fill me with a sense of discovery and make me gaze further at them until these fanciful forms saturated my imagination.

I encountered such playful sketches, albeit in a much more sophisticated and formal way, in George Stiny's Shape Grammar class at the beginning of my masters program in MIT. He would draw a 'star' and show us how one could see a chevron or an arrow inside it.



The 'chevron' and the 'arrow' were like the 'faces' in the scribbles. All of them shared an interesting characteristic – they were emergent ideas, i.e. they were not present in the actual drawing description of either the scribble or the star, but appeared as we *reflected* on them and attached meanings to different parts of it. The sketches thus became more than just the structure that described them but were 'containers' of ideas. What was fascinating further was how these different forms would remain merged within each other. As I switched *focus*, the mouth of a face would become the tail of the bird.

Properties of Sketch

Therefore, along with the ability to contain multiple concepts concurrently, there was additionally a *multiplicity of semantics*. Sketches thus allow designers to switch from one 'way' of looking (interpretation of meaning and structure) to another with ease and fluidity. This is possible because a sketch has no innate structure. It is a *flat collection of visual assertions*. It is only through interpretation that the structure of generated conceptions is resolved in the mind of the designer. It is this non-hierarchic nature of a sketch that allows ambiguity and multiplicity of semantics. Once hierarchy is introduced, the world becomes monotonous and descriptions become rigid. The parts of the face-concept could never become parts of the bird-concept – topologically they are different altogether. If topological and semantic variations were disallowed, exploration would reduce to parametric variations only. This is the essentially the reason for the 'blindness' of current computational systems in use and limiting nature of design exploration in such framework, where act of designing is more akin to 'modeling' specific pre-configured semantics.

Critic of Current Models

In addition, the act of 'reflection' is fundamental to the mechanism of design [Schon 1987]. It is not always the final state but the interim states during the process of design that are intriguing. There often 'emerge' new possibilities that are not 'intended' or 'defined' in the system a priori. Without a layer of reflection, it becomes nearly impossible to figure out such emergence. Generative Systems such as Shape Grammars are concerned with the local rules, while the 'reflection' is more concerned with larger and more holistic goals and principles of the design. In the realm of design computation, contemporary systems do not facilitate this. The act of 'reflection' would actually ride on the act of 'interpretation' and a higher-order (meta-)reasoning structure that would interpret the 'interpretation' itself. Marvin Minsky in 'Society of Mind' says,

> *"There is one way for a mind to watch itself and still keep track of what's happening. Divide the brain into two parts, A and B. Connect the A-Brain's inputs and outputs to the real world – so it can sense what happens there. But don't connect the B-brain to the outer world at all; instead, connect it so that the A-brain is the B-brain's world!" [Minsky 1985]*

The idea of reflection and interpretation in design remains neglected in contemporary computational tools. Contemporary tools are essentially used as drafting-aid or as a post-rationalization tool rather than for true design exploration.

Proposed Framework

In order to build a design exploration tool, we must first understand how we visually interpret abstractions like sketches and model a computational framework that allows the following,

a) **Simplicity** (The sketch is easily *extensible.*)

b) **Fluidity** (There is no *a priori declaration of intent* in a sketch.)

c) **Ambiguity** (There is *multiplicity of semantics and structure* in a sketch.)

d) **Context** (Ambiguity is resolved using *'certain ways of thinking'* about the world)

e) **Interpretation** (Association of *real world concepts* to Abstractions)

f) **Reflection** *(Meta-Interpretation)*

This is the motivation of this thesis.

2. Background - Revisiting some powerful ideas.

2.1 George Stiny's No Structure Computation

"The moment you represent a shape with structure you lose opportunities"

Stiny raises a very pertinent concern about information processing models of computation. He criticizes the use of 'prefigured' structure and semantics that are used to describe the world. These prefigured units or definite 'parts' with specific features already known *limits* both observation and subsequent action.

> *"Descriptions fix things in computation, and nothing is ever more than its description anticipates explicitly."[Stiny 1994]*

The lure of creating computationally manageable representations, result in unambiguous, rigid and monotonous definitions of the world. Since all 'representations' are fundamentally a surrogate for the described entity, not embodying the real physical entity, the fragmentation and incompleteness inherent in such representation schemes makes them almost synonymous to constrained strategies of looking at the world. Such strategies are always closely linked with certain task domains, which are concerned with only certain aspects of the entity. In the realm of design, the designer in his explorative mode flows from one representation to the other, switching from one strategy to the other and the rigidity of a single logical structure of representation is shattered. Present computational models fail to capture the essence of 'ambiguity'. Computation thus is reduced to 'modeling' the information in a certain way rather than being a tool for 'exploring' novel ways to interpret the information.

> *"Whenever we treat a situation…[by] analyzing it in terms of objects and their properties, we thereby create a blindness. Our view is limited to what can be expressed in the terms we have adopted … A [computer program] by which formally defined tasks can be performed with carefully designed representations … does not touch the problem of blindness. We accuse people of lacking common sense precisely when some representation of the situation has blinded them to a space of potentially relevant actions" [Winograd and Flores 1987]*

To remove such 'blindness' derived from the monotonous descriptions, there has to be multiple conceptions of the world. This thesis suggests that world is exactly 'what it is' and remains an unstructured collection of parts. This collection is referred to as the 'abstraction space' in this thesis. It is through interpretation that we derive 'descriptions' about it. These are the concepts and they comprise the conception space. A single abstraction space can give rise to several conception spaces. The computational framework introduced in this thesis models a reasoning structure for deriving concepts from the abstraction space.

Separation of the abstraction and conception spaces

2.2 Marvin Minsky's Distributed Cognition

"*Unless you can represent an idea in many ways you don't have an idea*"

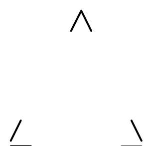
Minsky's theory of distributed cognition [Minsky 1985] suggests that we envision our mind (or brain) not as a unitary thing but as composed of many partially autonomous "agents" – a society of smaller minds. Then any 'mental state' can be interpreted as subsets of the states of the parts of the mind. Much like any human administrative organization, Minsky suggests that there are large divisions of our mind which specialize in areas like sensory processing, language, long-range planning etc; and within each subdivision there are *multitude*s of sub-specialists or agents which embody smaller elements of an individuals knowledge-base, skills and methods. These agents embody small units of knowledge; recognize certain configurations and respond by altering its (binary) state. Thus, the total mental state is described by the states of all the selected 'active' agents.

Visual perception seen in this light can explain the concept of ambiguity in shapes. After the first perceptive action of 'sensing' the world as a collection of parts, the agencies *react* to it, each using their limited rationality or reasoning structure, to change their states if required. To view this theory from the perspective of this thesis, this reasoning structure essentially encodes a schematic knowledge of some points of interest of the agents. If the points of interests are observed in the world, the agent triggers and generates a 'concept' in this world.

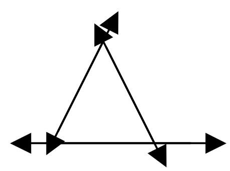
To elucidate this theory, let us imagine a set of agents who have been 'exposed' to a world containing just Shape M.

Shape M

The corner agent finds corner ideas in the world and creates corner-concepts.



The direction agent finds direction ideas in the world and creates direction-concepts.



The plane-agent finds a plane idea in the world and creates a planar-concept.

Introduction of Ambiguity

At this stage, the shape is neither resolved as a collection of three line concept nor as a single planar concept, but all the concepts are simultaneously present.

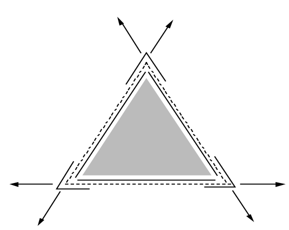


Figure: Directional, Planar, Corner, Edge ideas are simultaneously present

This captures the ambiguity of the world. Subsequent selection of sub-sets of active agents creates a partial-mental state and allows us to imagine the assertion as either three lines or a planar concept.

Real world concepts usually have multiple ways to *think* about them, and their representations should reflect that. For example, idea of a chair-object may have the following associations in our mind,

- 'Can sit on' (function)
- 'Has four legs' (structure)
- 'Stands of the floor' (relational)

In the framework modeled in this thesis, interpretation of a chair-object would happen through a reasoning structure based on these associations.

> *"To solve really hard problems, we'll have to use several different representations. This is because each particular kind of data structure has its own virtues and deficiencies, and none by itself would seem adequate for all the different functions involved with what we call common sense." [Minsky 1985]*

2.3 Pegor Papazian – Opportunism, Multiplicity of Semantics and Exclusivity of Seeing

"Our focus keeps changing with every design move"

Papazian highlights a few important elements of the act of 'designing', which provide pertinent background for the framework of this thesis. First, designing is based on a substratum of opportunistic activity. At any given time, the designer focuses on a limited number of components and evaluative criteria. If a 'stimulus' in the evolving design artifact is important enough for a given evaluative criterion, it will 'trigger' that criterion into focus [Papazian 1991]. In the framework of this thesis, the 'stimulus' is the collection of assertions from the world. Based on this collection a reasoning structure generate concepts and assign them a strength-metric as well i.e., there can be strong concepts as well as weak concepts. However, in the context of the design, a weak concept can however be an interesting one nonetheless. This translates to an 'opportunity' on the designer part to emphasize and strengthen this concept.

Papazian also highlights the idea of 'Multiplicity of Semantics' in a design document. The designer can interpret a sketch by attributing to its components and their relations one of many sets of meanings.

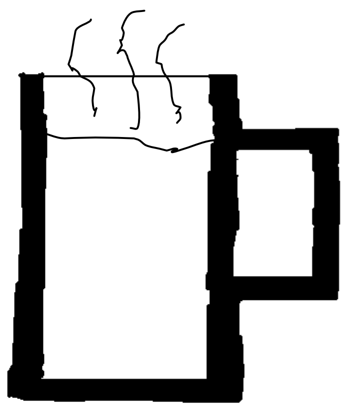


Figure: Section of a Jug

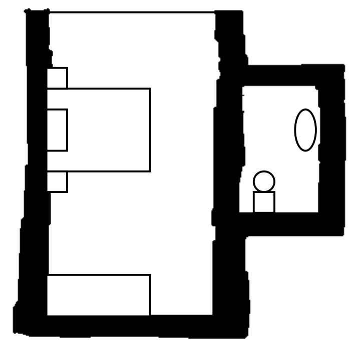


Figure: Plan of a Room

The figure above demonstrates an example of how the same abstract representation can represent multiple concepts. Without the visual clues (the thinner lines) there is no way a viewer can grasp the specific meaning. In his book, 'Emotion Machine' Marvin Minsky suggests that to appropriately interpret an abstract representation the author of the abstraction and the viewer needs to share a common 'realm of thought'. If no such agreement occurs between the author and the viewer, the viewer's mind would keep switching between points of views using some rapid machinery in his brain. According to Minsky, this is done by automatically linking analogous aspects of each view to the same role in one larger-scale structure. He calls it Panalogy (Parallel Analogy)

The figure below shows two possible 'realm of thought' for this graphic. The 'Room Layout' Realm and the 'Mug' Realm. For example, a physical wall might way be linked to the concept of a separator in a distant way (*'the Berlin wall'*: political realm). In essence, the schemas and concepts form of a massive knowledge network. What the 'realm of thought' or context enables us is to ignore 99.99% of our knowledge so that we can focus on the 'task at hand' [Lenat 1998]

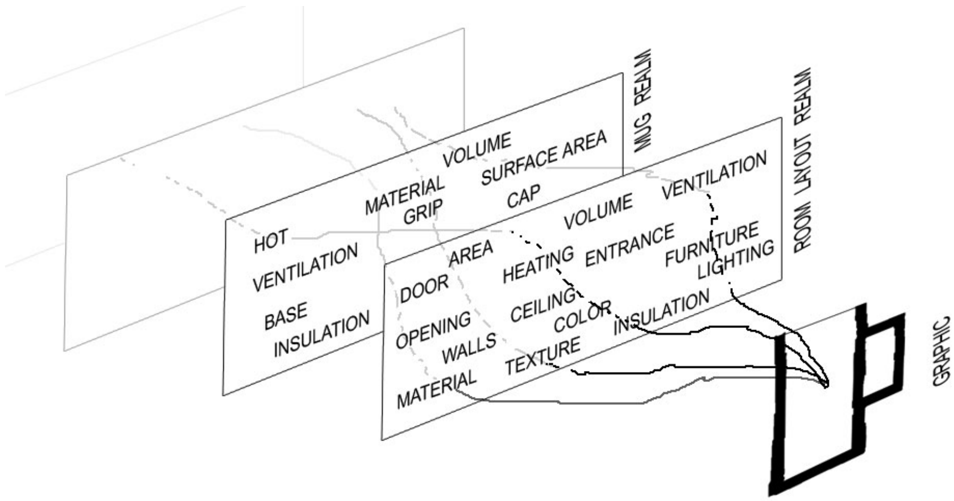


Figure: Realms of Thought

Exclusivity of Seeing

Alternating Illusion based on the classic vases-faces drawing, illustrates that a given interpretation of an image cannot simultaneously be perceived as one thing and another.

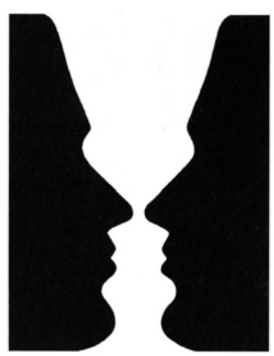


Figure: Vase-Face Ambiguity

Therefore during interpretation, there can be only one potential semantic active in the designers mind at any given time i.e. there can be no concurrent contexts in the interpretation cycle.

2.4 Shimon Ullman – Visual Routines and Spatial Reasoning

"There are elementary procedures that are assembled for vision"

Creation of the 'abstraction space', i.e. description of the world as a collection of assertions in elementary terms, is perhaps the first perceptual step necessary for cognition. The visual system does this with remarkable proficiency that cannot be mimicked at present by artificial computer vision systems. Shimon Ullman proposes that visual analysis of shape properties and spatial relationships is based on a set of elementary visual routines [Ullman 1996]. He suggests that visual processing is divided into two stages – First there is a viewer-centered computation of the 'base representation' and second, the extraction of abstract spatial properties by visual routines from such a base representation. It has been assumed that visual routines are based on a set underlying elemental operations like shift, bounded activation, boundary tracing, and marking.

However, an enquiry into visual routines for visual perception is beyond the scope of this thesis and for simplicity, the proposed framework is based on a simple and constrained user interface which allows only orthogonal lines. To create a framework, which can observe the world, create complex abstraction spaces before triggering interpretation cycles, further work is required in the lines of Ullman [Ullman 1996] and Rao [Rao 1998].

3. Theoretical Basis for the Model

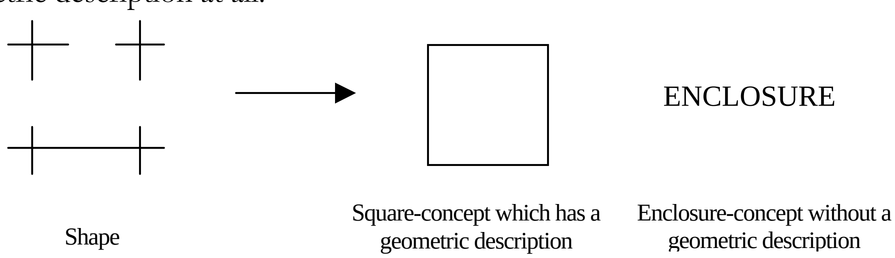
3.1 Abstraction and Conception Spaces

Sketches are abstract representations. However, designers use them to explore real world concepts. Thus, it is important to clarify between the abstract and concept early on in the discussion.

The actual shapes that form the sketch physically comprise the abstraction space, while the real world concepts that the abstractions trigger in the designers mind make up the conception space. While a designer works on a sketch, these two spaces interface and inform each other. Real world concepts are *learnt* from spatial, visual and tactile experiences. Since most people have different sets of experiences or memory, the conception space is subjective and is different from person to person. Hence, it follows that 'seeing' or 'visual interpretation' is subjective and two people with two different backgrounds might interpret completely different concepts in the same abstraction space.

3.2 Assertions and Concepts

An assertion can be defined as 'any pattern that can be observed' in the world. A visual assertion is a pattern or a shape that may be a collection of points, lines, and planes. Additionally it might have other qualities of patterns such as texture, gradient and even color. In this thesis, the line-sketches are the shape-assertions that make up the world and are used during the interpretation cycle to generate concepts. Concepts are imagined ideas inferred from the world. They may or may not have any physical existence in the actual assertions that generate it. For example the shape below might generate an 'enclosure' concept that might not have any geometric description at all.



Concepts have multiple representations and associations to rules and other concepts. The representations can be geometric, functional, language-based descriptions. In the process of design all these descriptions play an important role as the designer switches back and forth from different ways of conceptualizing in his explorative mood. For example, a designer might start off from a geometric description and switch to other kinds of descriptions and finally again return to a geometric description.

geometry -> *closed-shape* -> *'enclosure'* -> *'dark'* -> *'light'* -> *'window'* -> *geometry*

Each representation is like a mode of thinking about the concept and within each mode there are associations to other concepts and action-rules. A 'courtyard' concept may be represented as follows,

```
(Concept
(Tag courtyard)
(Geometry-mode
        (properties (boundary-information) (aspect) (closure) (edges) (corner) (ends))
        (associations (enclosure))
        (action-rules (define-center) (define-corners) (divide-into-parts)))
(Functional-mode
        (properties (gathering-place) (allows-light) (allows-wind))
        (associations (ampitheatres))
        (action-rules (add-fountain) (create-shades)))
(Relational-mode
        (properties (next-to-buildings) (has-entrance))
        (associations (pathways)(sky))
        (action-rules (create-transitions)))
(Strength 1))
```

Concepts also have a metric of strength associated with them. This metric is assigned by the reasoning structure, which 'makes' the concept, and it refers to how close the actual assertions map on to the logic of the reasoning structure.

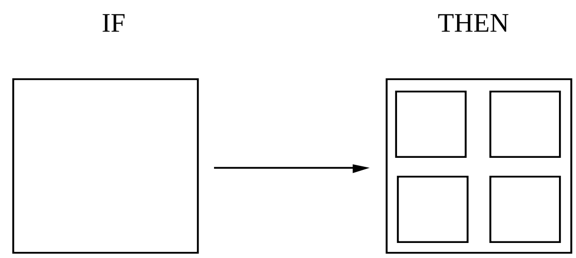
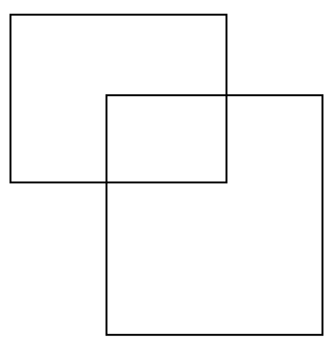


Figure: Action-Rules associated are associated with Visual Concepts

Visual concepts have visual rules attached with them in memory. Whenever a visual concept is triggered in the mind of a designer, the associated rules are the possible forward moves the designer can use in the next step of design. Therefore, in that sense the concepts drive the design moves. At every step when the abstractions trigger multiple concepts, the designer chooses the most opportunistic one (aligned with more holistic design goals) to drive the design generation. This is the Principle of Opportunism in Design and has been well elaborated in Pegor Papazian's PhD thesis [Papazian 1991].

3.3 Shapes



Shape A

Shapes are collections of 'visual assertions' that generate concepts in our mind. A shape has no inherent hierarchical geometric structure associated with it and can be conceived as a 'flat' collection of parts. The visual concepts, which are generated after interpretation however have hierarchical structures, associated with them. In essence, interpretation is what gives structure to the world. Philosophically the interpreter or the observer gives meaning to the world and essentially is the author of it.

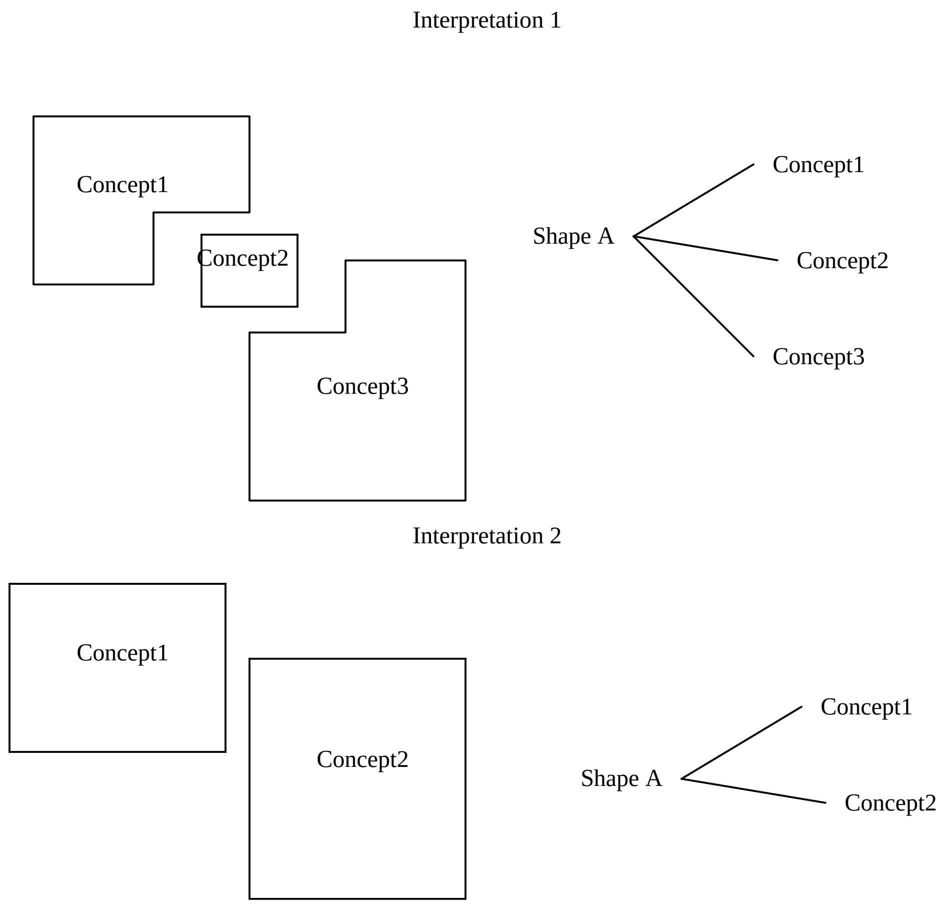


Figure: Two possible interpretations of Shape A. Notice their conceptual and topological difference.

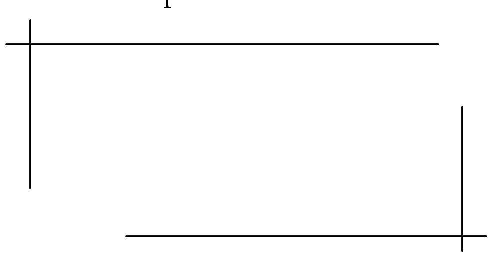
3.4 Schema

During Interpretation, generation of concepts from shapes, involves a deduction system; at the core of it lay reasoning structures called schemas. Shapes can trigger Schemas and Schemas in turn generate concepts. Schemas essentially contain 'schematic' knowledge about concepts. When the system[1] encounters the following visual assertion, the rectangle-schema is triggered,

```
(If 'assertion
      IS '(closed 1)
      HAS '(edge edge edge edge)
      HAS '(aspect (or 'wide 'tall))
 Then MAKE-RECTANGLE-CONCEPT)
```

Visual Assertion Rectangle Schema

It should be noticed the Schema is independent of exact geometrical structure of the assertion, i.e. is does not deal with exact metrics like vertex information, rather its more concerned with the schematics like whether an edge exists or not, whether it is tall or wide, whether it is closed or not etc. From the cognitive perspective, during the process of 'seeing', visual preprocessors [Ullman 1996] generate certain basic measures from the assertion. These measures must be elemental and contain *just-enough* information to trigger a large no of schemas, most of the time. This argument is aligned with the Goldilocks Principle highlighted in [Ullman, Vidal-Naquet and Sali 2002]. In the LISP machine, that I discuss later in Chapter 5, 'directional ideas' and 'edge-ideas' are used elementary measures. This independence from actual geometric structure allows Schemas to generate concepts from even from weaker assertions, where their actual geometric structure of the concept does not exist. For example a rectangle concept can be generated from the following by a schema with the 'closure' predicate is relaxed.



```
(If 'assertion
      IS '(closed 0.5)
      HAS '(edge edge edge edge)
      HAS '(aspect (or 'wide 'tall))
 Then MAKE-RECTANGLE-CONCEPT)
```

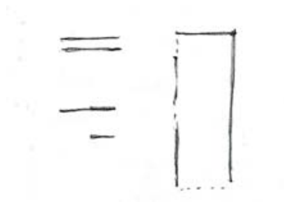
Visual Assertion Rectangle Schema

1 'System' refers to the LISP machine described in Chapter 5

The Schemas can be of different types,

a) **Geometric**

A courtyard has a boundary with three or more edges



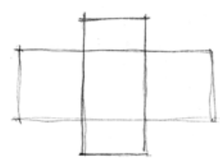
The representation in the framework,

(**courtyard**) -> (enclosure) -> ‘(unsorted (edge,edge,edge,*))

The symbol ‘*’ refers to a ‘don’t-care’ condition i.e. an edge or no-edge both satisfy.

b) **Relational**

A courtyard cannot be next to another courtyard in the abstraction i.e. all the rectangles in the figure below cannot be courtyards simultaneously,

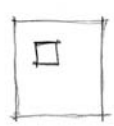


The representation in the framework,

(**courtyard**) -> (not (next-to (courtyard)))

c) **Logical**

A courtyard cannot be inside a room. In the abstraction below if the bigger rectangle is interpreted as a room, then the smaller one is not a courtyard.



The representation in the framework,

(**courtyard**) -> (not (inside (room)))

d) **Functional**

(**courtyard**) -> (gathering-place) -> (adjacent-to (many (building)))

(**courtyard**) -> (has-a fountain)

Schemas are learnt from real world interaction and experience. They are used to schematically store our knowledge about the world. As shown previously the schematic nature is independent of the exact geometric structure of the world, since conceptually similar things might not share exact geometry. Schemas however contain the logic of associating geometric structure to generated concepts based on the abstractions in the world.

Schemas can trigger other schemas within the context as well. For example, a courtyard-schema might trigger entrances-schema or fountain-schema. Therefore, Schemas also gives a structure to the entire conception space as well.

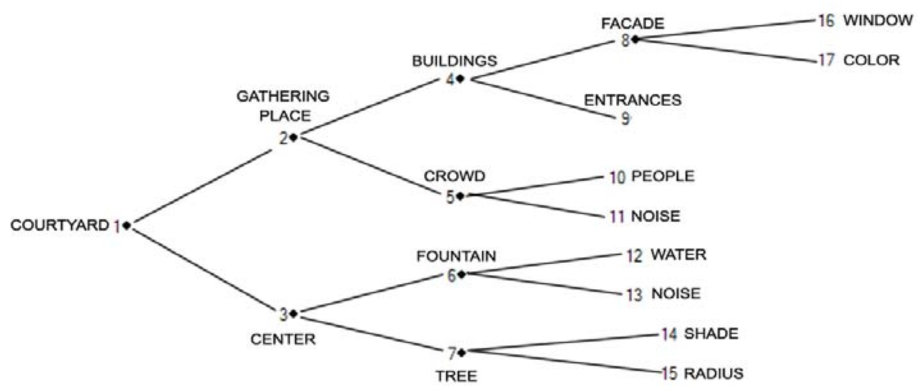


Figure: Hierarchic Conception Space

3.5 Active, Passive Assertions and the Imagined World

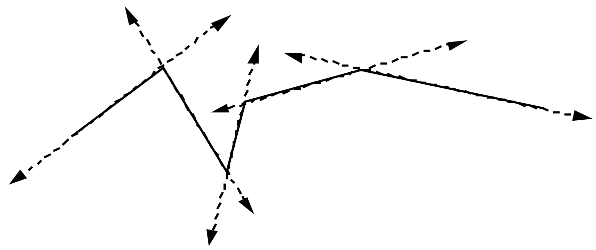


Figure: The active edge assertions ‘activate’ passive directional assertions

Active assertions are those assertions, which are physically present in the shape. These assertions however ‘activate’ secondary ‘passive’ assertions. For example, ‘Edges’ activate ‘Direction-concepts’. In the framework of the thesis, this collection of active and passive assertions in focus makes up the ‘Imagined World’. This imagined world is the basis for interpretation. This is analogous to Paul Klee’s Active, Medial and Passive elements in his ‘Pedagogical Sketchbook’ [Klee 1953].

3.6 Cognitive filter: Context

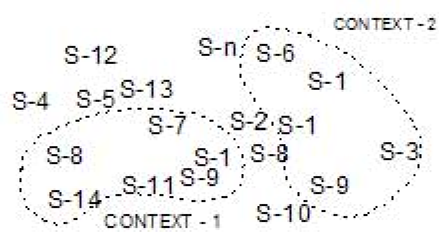


Figure: Contexts are collection of schemas

Context is a cognitive filter, which selects a subset of schemas from the entire pool of schemas in memory. Without context it becomes extremely likely that large number of schemas will be triggered, diminishing the chances of resolution[2]. To allow for generality[3] as well as coherence,

[2] Having a single resolved interpretation of the abstraction
[3] Different schemas refer to the same general set of elemental concepts for reasoning

Context filters are extremely important in this framework. In the LISP machine, contexts are represented as a list of schemas

(Context
('Building 'Courtyard 'Alley 'Street 'Entrance 'trees…'cul-de-sac)
(Strength 1))

For example, the 'site plan' context would select all the site-plan related schemas (building, green, streets, trees, crossings etc). Therefore, when a sketch is viewed in the context of 'site-plan', only schemas within that context may be triggered. The same sketch when viewed in the context of a 'plumbing-fixture' would trigger different schemas. It is possible that two different contexts have links to the same schema. This is especially true for general or basic schemas. For example, a cross-schema might be triggered by a road crossing in the site-plan context or by a pipe-junction in the plumbing-fixture context. The contextual focus manager thus constrains us to 'think in a certain way' and allows for quicker resolution of the meaning of the sketch.

3.7 Perceptual filter: Focus

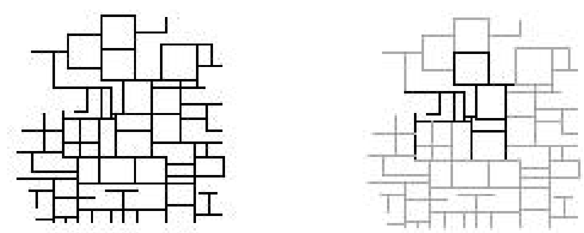


Figure: Focus

Focus is perceptual filter, which allows the selection of only a subset of spatially related 'assertions' for the deduction system to work on during interpretation. By reducing the number of assertions it reduces the processing time for the deduction system, maintaining the smooth flow of operation during a sketch exploration.

3.8 Memory and Persona

It is important to make explicit the role of memory in Visual Interpretation. One of the core arguments of the thesis is the claim that '*To See one must almost Know*'. This is aligned with Semour Papert's argument that to learn one must almost know already. From this argument it can be further derived that our learning is incremental and every forward step is reliant upon previous states. From the perspective of the computational model, to see, i.e. forming a visual concept, one must have either a corresponding schema or *nearly* corresponding schema in the memory. The word 'nearly' is emphasized, because the idea of 'near-miss' is a learning opportunity [Winston 1977] and it is important for otherwise nothing new would ever be discovered.

The involvement of memory makes seeing a subjective act. Since schemas in memory are learnt from experience, the same set of assertions in a sketch might trigger different sets of schemas in different people, giving rise to different interpretations. It is not difficult to understand in these terms why an engineer would often see different things in a sketch than a poet or an artist.

4. Visual Interpretation

4.1 The Model

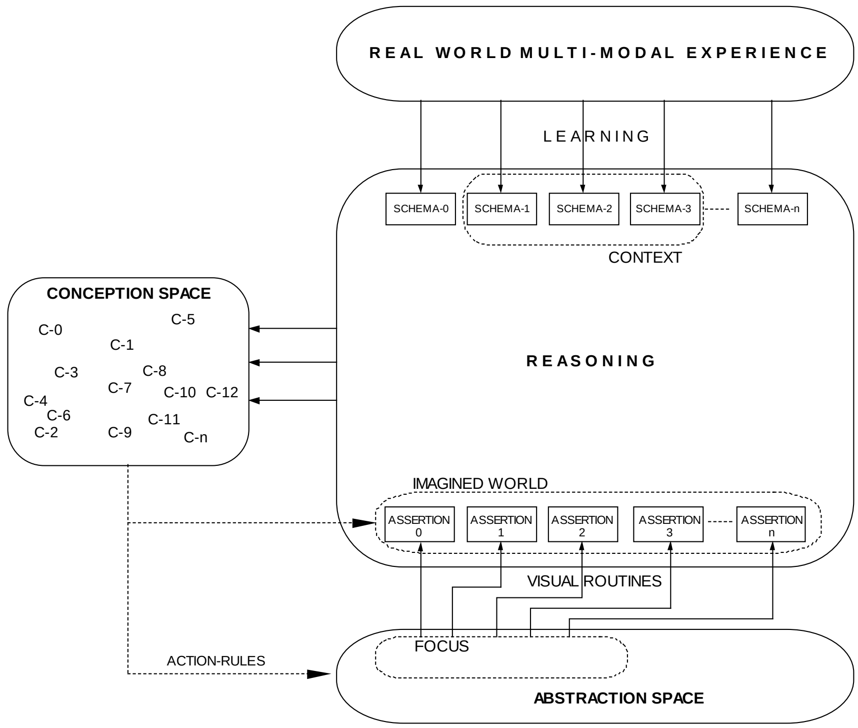


Figure: Schematic Diagram for the Visual Interpretation Model

This thesis models Interpretation as successful *triggering* of schemas from assertions within a context, to generate concepts. The reasoning structure does the actual interpretation using the schemas in context and the imagined world. Each schema contains a reasoning structure, which is used to scan the imagined world in focus and ascertain whether a 'matching' criterion is found. If there is a match between these assertions and the reasoning structure of the schema, the schema is "triggered" and concepts are generated. The knowledge required to create the concept from the assertion is also embedded within the schema.

The generated concepts have multiple descriptions (the most prominent of which in the context of this thesis is the geometric descriptions), action-rules and associations embedded in them. The action-rules *(draw-geometry-of-concept)* might be used to add active assertions (shapes) to the abstraction space (sketch), which is similar to a person drawing further on the sketchpad or might be used to just add passive assertions to the imagined world, which is similar to a person imagining ideas without actually sketching it. This is how the collection of active and passive assertions in the imagined world keeps incrementing with every cycle of interpretation in this framework.

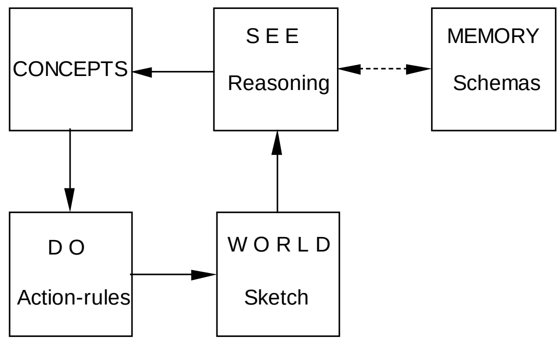


Figure: Extended See-Do-See cycle for Interpretation

When the assertions partially match the predicates of a schema, a 'weak' concept is generated. The strength of any concept is ascertained by the logic of the schema. Therefore, over time the reasoning structure produces a pool of 'weak' and 'strong' concepts in the conception space, which are translated by the actions to become part of the imagined world for the next interpretation cycle.

4.2 Finding Emergent Courtyards in a Site Plan

A) The Abstraction Space

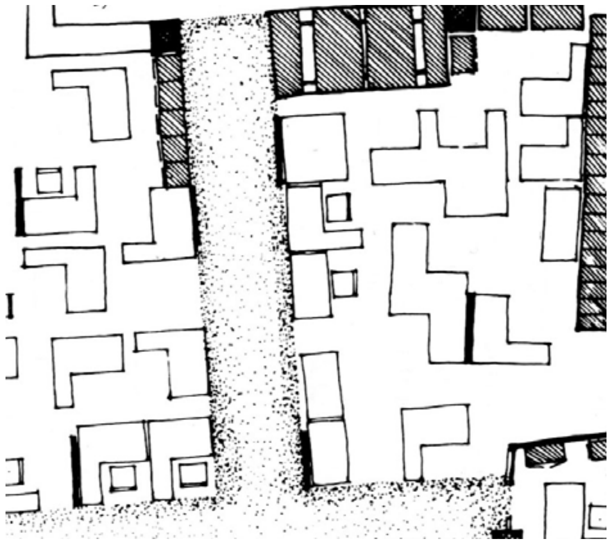


Figure: The sketch is a collection of visual assertions. These assertions make the abstraction space

B) Focus

Figure: Assertions in 'focus'

C) Passive Assertions

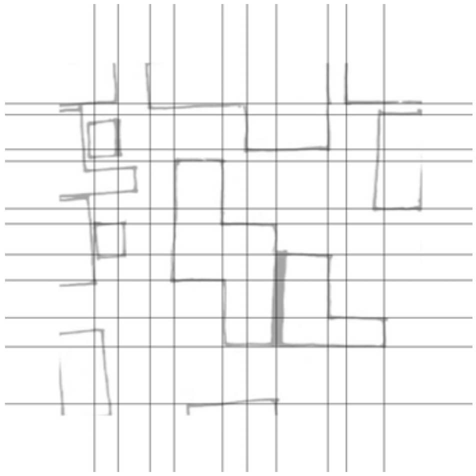


Figure: Directional Concepts

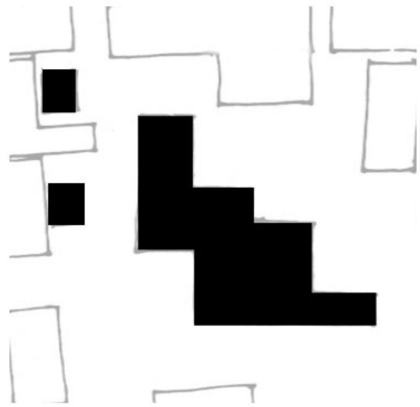


Figure: Figure Concepts

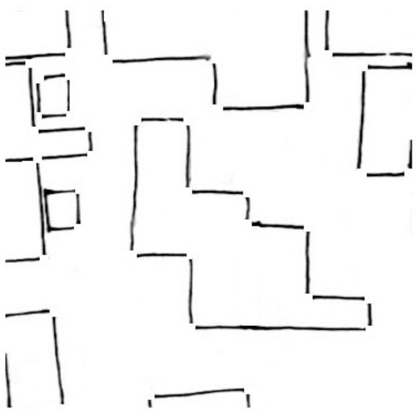


Figure: Edge Concepts

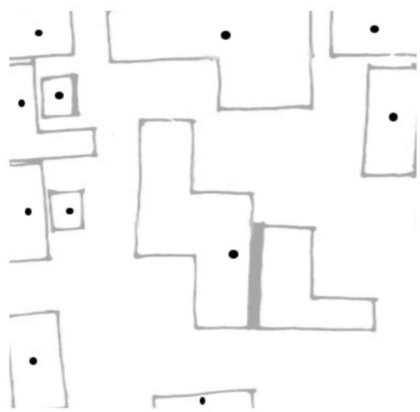


Figure: Position Concepts

D) Generation of the Imagined World (Overlap of Active and Passive Assertions)

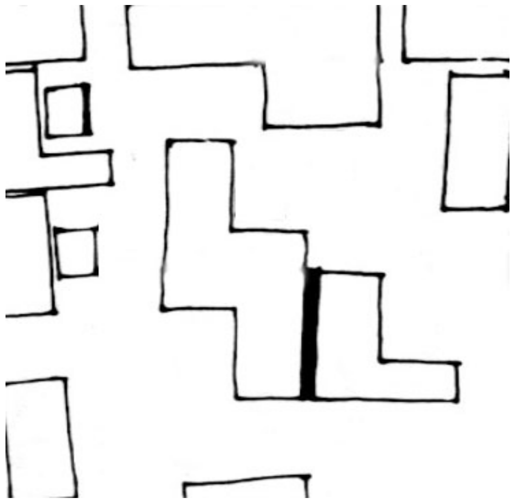


Figure: Active Assertions

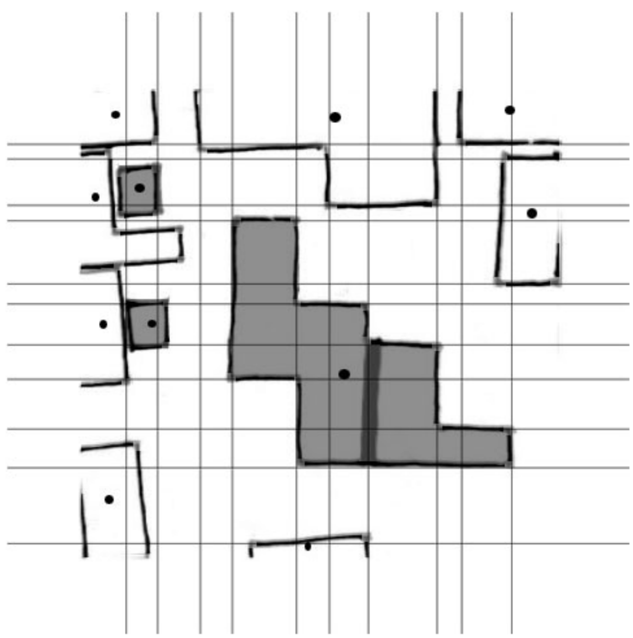


Figure: Active and Passive Assertions

E) Courtyard Schemas and the Reasoning Structure

The 'Site Plan' context is a collection of the schemas like,

(Buildings)

(Courtyard)

(Street)

(Footpath)

(Entrances)

(Trees)

.

.

.

(Parking)

(Cul-de-sacs)

Hence, visual interpretation of the sketch in the context of 'site plan' will use the reasoning structures or heuristics embedded in this set of schemas. These reasoning structures can be geometric, functional, relational or logical. The following is an example of a courtyard schema,

```
(courtyard-schema) ->
(concept which-is-in 'world
  which-is-not-inside 'figures
  which-is-not 'too-wide 'too-long
  which-has '(edge edge edge edge)
  which-has '(edge edge edge)
  which-is-not 'too-small )
```

F) Formation of the Courtyard Concepts

The schemas generate the following courtyard concepts.

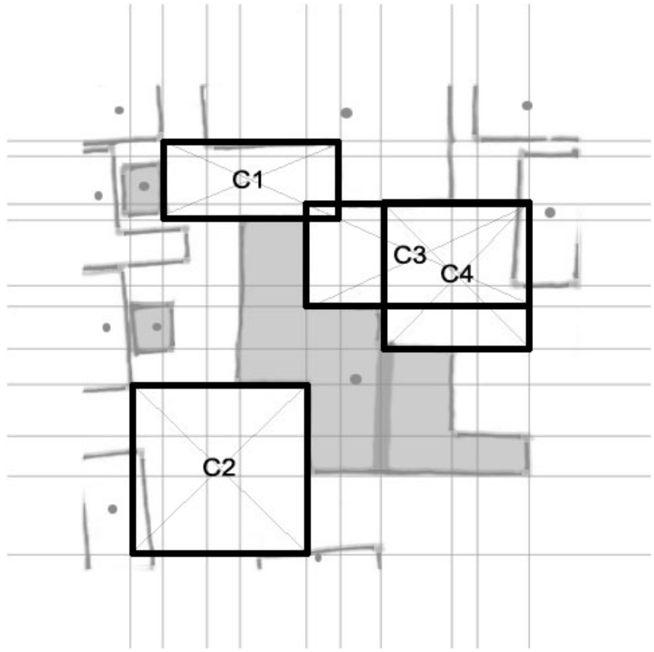


Figure: Generated Courtyard Concepts

These concepts are added to the 'Imagined World', for the next cycle of interpretation. By default, the concepts remain as 'passive' assertions.

G) Action-Rules

Further, action-rules embedded in the courtyard-concepts can be used to add 'active assertions' to the abstraction space.

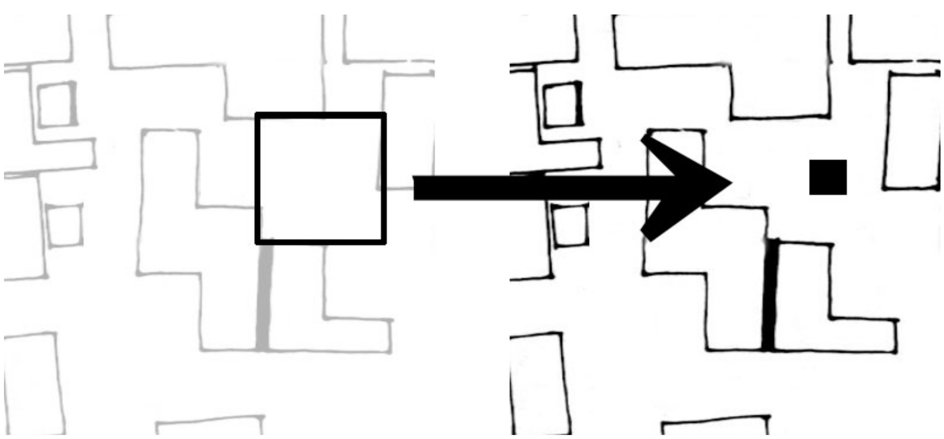


Figure: Application of Action-Rules

5. LISP Machine

5.1 Principles

In this chapter I will introduce a computational framework called TRIGGER. The framework allows the user to draw sketches using a simple drawing interface. The sketch triggers interpretation cycles, based on the model proposed in previous chapters. The salient principles of this framework are the following:

A) The world in itself is flat, the conceptions about it are however hierarchic.

The sketch is treated as a flat collection of visual assertions. Conceptually this is one of the core differences between the framework proposed and the contemporary computational systems in use. The flat-ness of the world allows the framework to accommodate notions of ‘multiplicity of semantics’ and ‘emergence’.

B) Knowledge in seeing.

In this framework any interpretation requires schematic knowledge in memory. ‘Memory’ comprise of a collection of schemas, which encode real world knowledge. These form the basis for interpretation.

C) Conceptions space increments with interpretation.

In this framework Visual Interpretation has been modeled as a see-do-see cycle. A recursive structure has been proposed which triggers schemas every cycle and generating collection of concepts, which accumulate over time.

D) Principle of exclusivity

In this framework there is only one context active in the working memory of the machine at any point. This setup is based on the assumption that we cannot have two separate active semantic ideas concurrently.

5.2 Parts

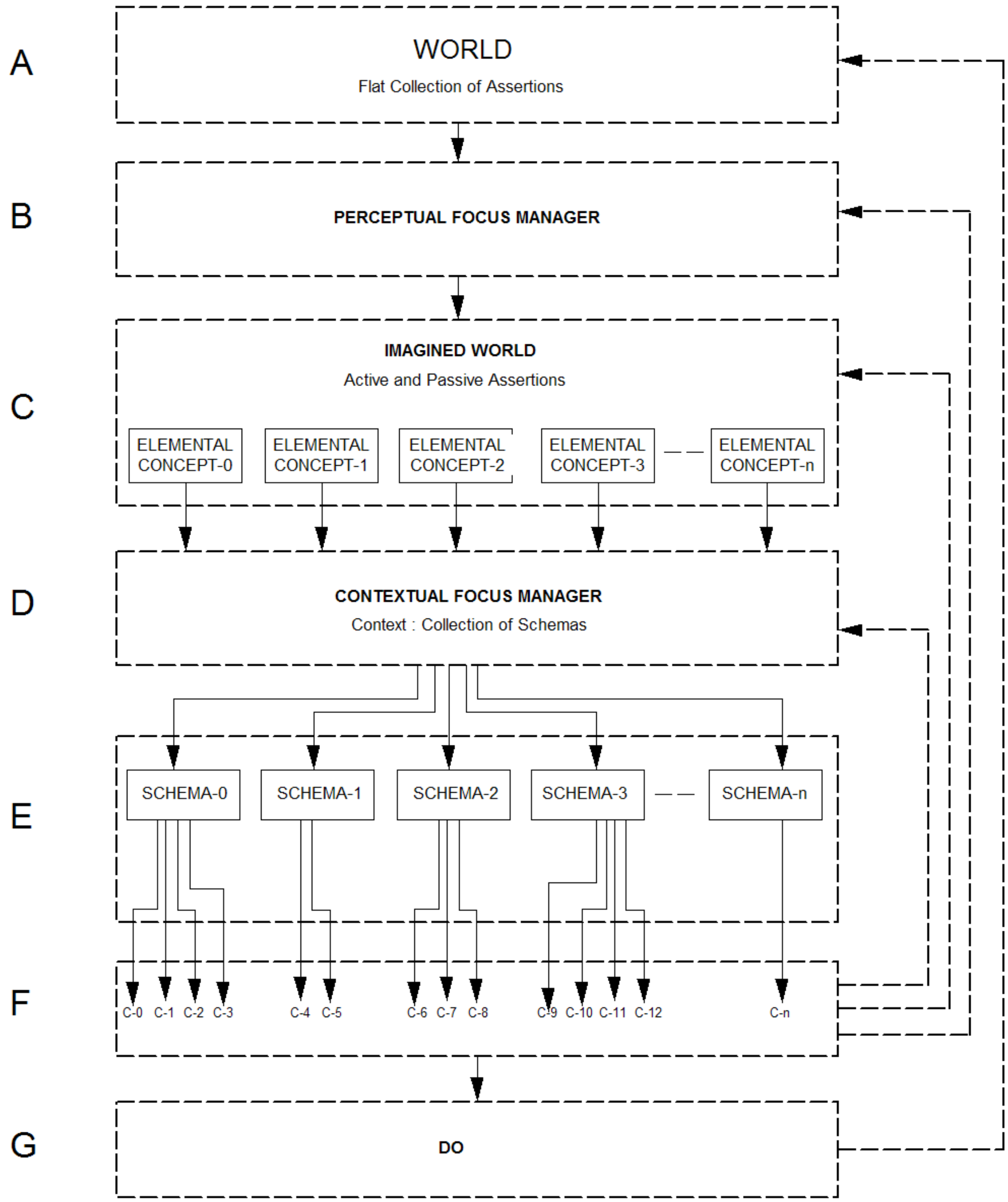


Figure: The flowchart showing the main logical parts and the flow of the LISP machine

A) World

In order to simplify and work with a small set of visual-routines, TRIGGER allows the user to sketch using orthogonal lines only. The initial sketch (collection of lines) is stored in a list. This list is termed as the *world*.

B) Perceptual Focus Manager

The perceptual focus manager selects a subset of the shape assertions in the 'world'. To simulate gaze in a simple manner TRIGGER creates a boundary of preset dimensions, randomly places it in the sketchpad and selects the parts of the sketch, which are completely or partially inside the imposed boundary.

C) Imagined World

TRIGGER applies a set of visual routines to extract passive assertions from the initial sketch. These passive assertions are in fact the first elemental concepts. These are *elemental* as no reasoning or interpretation is required to generate them. The sketch gives rise to Edge-concepts, Direction-concepts, Figure-Concepts, Position-Concepts. All of these together make up the *Imagined world.*

```
(WORLD
(CONCEPT (ST (((87 172) (87 76) (377 76) (377 172)) (232 124) (290 96))) (PR (145/48 0 NIL 4 TRUE)) (TG FIGURE) (RL NIL))
(CONCEPT (ST (((88 296) (88 212) (165 212) (165 296)) (253/2 254) (77 84))) (PR (11/12 0 NIL 4 TRUE)) (TG FIGURE) (RL NIL))
(CONCEPT (ST (((26 498) (169 498)) (195/2 498) ((26 498) (169 498)))) (PR (WIDE 0 (2 4) 0 FALSE)) (TG EDGE) (RL NIL))
(CONCEPT (ST (((26 369) (26 498)) (26 867/2) ((26 369) (26 498)))) (PR (TALL 0 (1 3) 0 FALSE)) (TG EDGE) (RL NIL))
(CONCEPT (ST (((169 369) (26 369)) (195/2 369) ((169 369) (26 369)))) (PR (WIDE 0 (2 4) 0 FALSE)) (TG EDGE) (RL NIL))
(CONCEPT (ST (((169 498) (169 369)) (169 867/2) ((169 498) (169 369)))) (PR (TALL 0 (1 3) 0 FALSE)) (TG EDGE) (RL NIL))
(CONCEPT (ST (((165 296) (88 296)) (253/2 296) ((165 296) (88 296)))) (PR (WIDE 0 (2 4) 0 FALSE)) (TG EDGE) (RL NIL))
(CONCEPT (ST (((380 202) (380 407)) (380 609/2) ((380 202) (380 407)))) (PR (TALL 0 (1 3) 0 FALSE)) (TG EDGE) (RL NIL))
(CONCEPT (ST (((87 172) (87 76)) (87 124) ((87 172) (87 76)))) (PR (TALL 0 (1 3) 0 FALSE)) (TG EDGE) (RL NIL))
(CONCEPT (ST (((87 0) (87 1200)) (87 600) ((87 172) (87 76)))) (PR (TALL 2 (1 3) 0 NIL)) (TG DIRECTION) (RL NIL))
(CONCEPT (ST (((377 0) (377 1200)) (377 600) ((377 76) (377 172)))) (PR (TALL 2 (1 3) 0 NIL)) (TG DIRECTION) (RL NIL))
(CONCEPT (ST (((299 0) (299 1200)) (299 600) ((299 407) (299 202)))) (PR (TALL 2 (1 3) 0 NIL)) (TG DIRECTION) (RL NIL))
(CONCEPT (ST (((380 0) (380 1200)) (380 600) ((380 202) (380 407)))) (PR (TALL 2 (1 3) 0 NIL)) (TG DIRECTION) (RL NIL))
(CONCEPT (ST (((88 0) (88 1200)) (88 600) ((88 296) (88 212)))) (PR (TALL 2 (1 3) 0 NIL)) (TG DIRECTION) (RL NIL))
(CONCEPT (ST (((0 212) (1200 212)) (600 212) ((88 212) (165 212)))) (PR (WIDE 2 (2 4) 0 NIL)) (TG DIRECTION) (RL NIL))
(CONCEPT (ST (((0 296) (1200 296)) (600 296) ((165 296) (88 296)))) (PR (WIDE 2 (2 4) 0 NIL)) (TG DIRECTION) (RL NIL))
(CONCEPT (ST (((0 369) (1200 369)) (600 369) ((169 369) (26 369)))) (PR (WIDE 2 (2 4) 0 NIL)) (TG DIRECTION) (RL NIL))
(CONCEPT (ST (((0 498) (1200 498)) (600 498) ((26 498) (169 498)))) (PR (WIDE 2 (2 4) 0 NIL)) (TG DIRECTION) (RL NIL))
)
```

D) Contextual Focus Manager

The contextual focus manager selects a subset of schemas available in memory. It limits the ways TRIGGER can interpret the sketch. Contexts are modeled as lists of related schemas. For example a site plan schema in TRIGGER would be represented as

```
(context ‘site-plan
          (‘building ‘courtyard ‘alley ‘street ‘entrance ‘trees…‘cul-de-sac)
          (strength 1))
```

E) Schemas

TRIGGER collects information from the sketch to find possible concepts using combinatorial procedures and then it filters out the ones, which do not match the predicates of the schema. The following is a definition of a schema which takes the entire ‘world’ (the sketch) as its input and then generates ‘L – concepts’ if it finds any.

```
(defun trigger-L-schema (world)
 (let* ((direction-concepts (extract-concepts world 'direction))
       (el-as-direction-sets (match-pattern direction-concepts '(tall tall tall wide wide wide)))
       (possible-el-structures (make-l-structure el-as-direction-sets direction-ideas))
       (filtered-el-structures (aspect-el-check possible-el-structures))
       (filtered-el-structures (filter-edge-value 3 filtered-el-structures world))
       (filtered-el-structures (filter-not-contain 'figure filtered-el-structures world))
       (el-list-set (make-el-concept filtered-el-structures 1))
       )
  el-list-set
  ))
```

Once the schemas generate concepts from the sketch they contain methods to associate descriptions to the generated concepts as well. The *make-el-concept* in the previous function is defined as follows,

```
(defun make-el-concept (el-st-list strength)
  (mapcar #'(lambda(x) (list 'concept
                        (list 'st (list x (find-center (list (nth 2 x) (nth 5 x))) 'aspect-lns))
                        (list 'pr '(nil nil nil 4 true))
                        (list 'tg 'el)
                        (list 'rl ‘draw-to-sketchpad ‘make-entrance)
                        (list 'sr (find-strength x))
                        (list ‘assoc nil)))
            el-st-list
            )
 )
```

F) Concepts

The concepts generated by the schemas are added to the *imagined world* by default. At this point they exist for the next cycle of interpretation but are not drawn to the sketchpad. They can potentially affect the cognitive filter (contextual focus manager) – in terms of the framework, concepts can have association with them and when they are generated, these associations can trigger further contexts. These triggered contexts become active and drive forward cycles of interpretation. Therefore generated concepts can trigger a certain 'frame of mind' or 'way of thinking' during interpretation. The interpretation cycle is modeled as a recursive function that triggers schemas, generate concepts and extracts associations every cycle. The following is the LISP function that is used,

```
(defun look-at (context world)
 (cond ((eq context 'nil) nil)
     (t
      (cons (trigger-schema (first context) world)
           (look-at (append (extract-associations (first context))
                            (rest context))
                   world)))
     ))
```

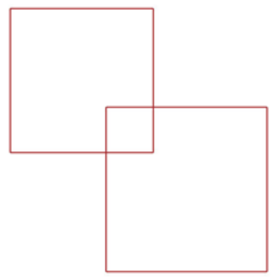
G) Action

The generated concepts have action-rules associated with them, which can be triggered at this point. For instance a common action rule *draw-concept-to-sketchpad* action-rule uses the geometric descriptions of the concepts to draw them on to the sketchpad. This adds assertions to the world and such assertions are considered in the next forward cycle of interpretation. This is the 'do' part of the entire design cycle.

5.3 Example runs

In the current version of TRIGGER, only a limited set of schemas is implemented. The idea was to develop a simplistic model as a proof of concept rather than building a tool with any real application in mind.

Sketch – 1



In the interface TRIGGER lists out the concepts it generates and also gives visual hints (dots) in the sketchpad to where it found them. The elemental concepts are shown in the bottom left window (edges, directions and positions) of the interface.

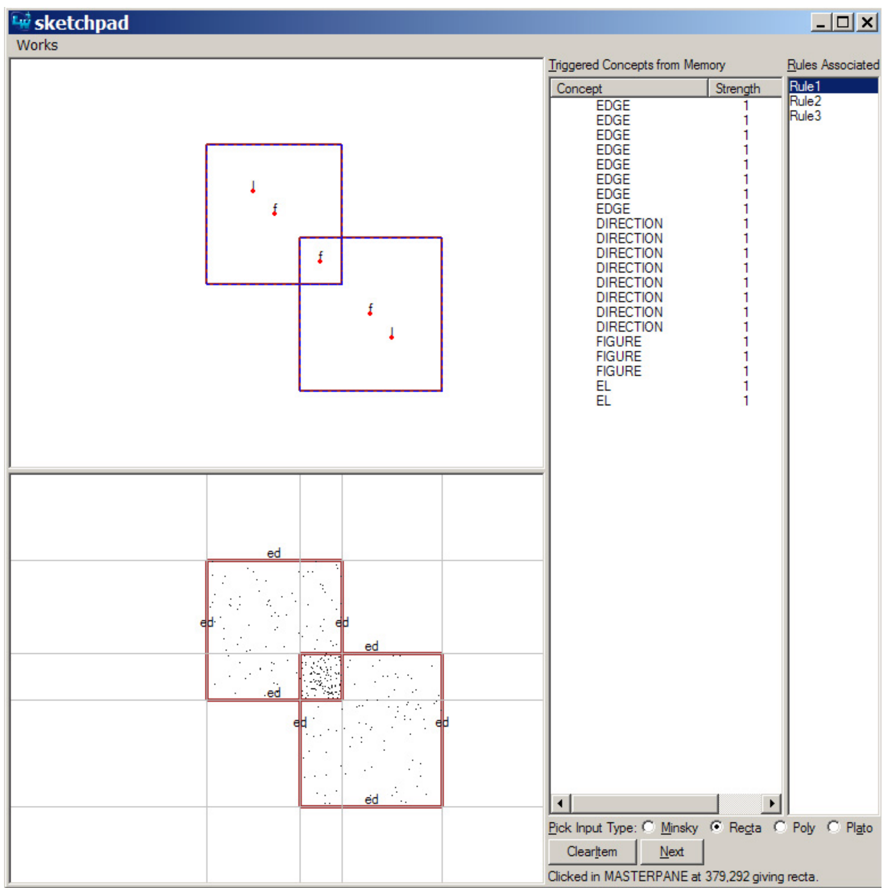


Figure: The interface of the TRIGGER program

Other than the elemental concepts of direction and position, TRIGGER generates the following concepts,

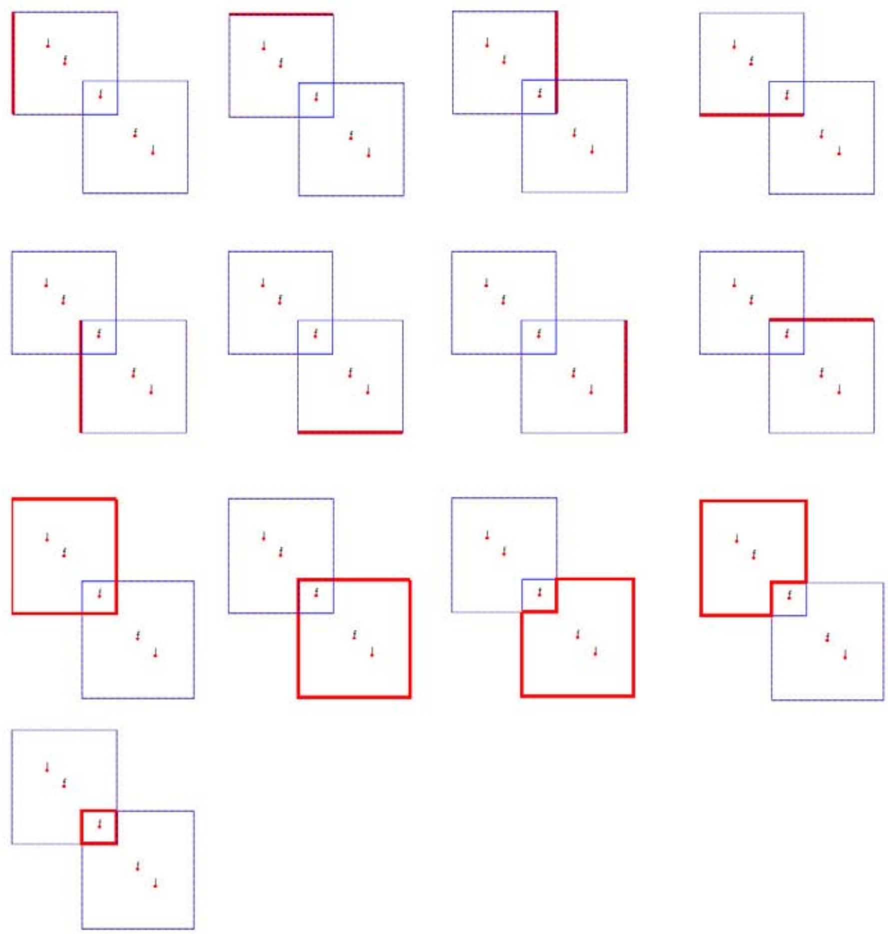
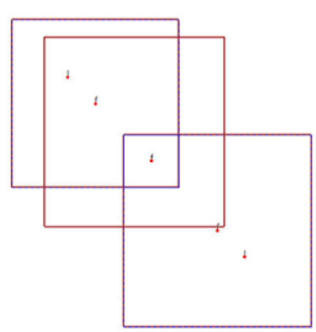


Figure: Concepts generated

If the sketch is further modified to the following,



TRIGGER uses the schemas in memory to find the following concepts in the sketch. Notice the multiplicity of semantics where part of an edge becomes a part of the rectangle or a part of an L interchangeably.

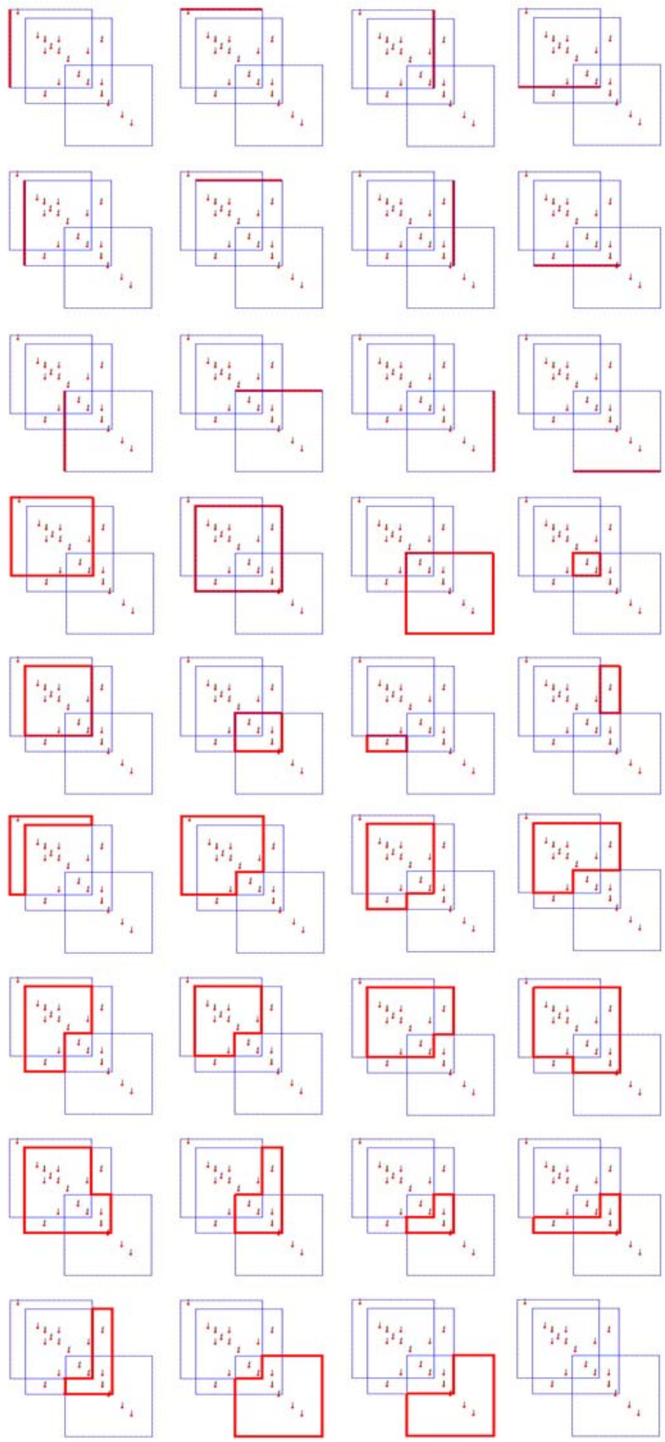
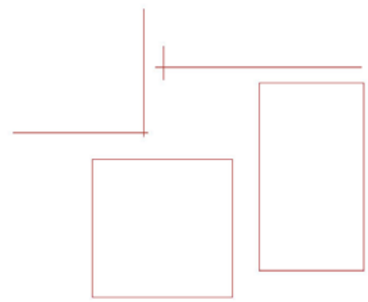
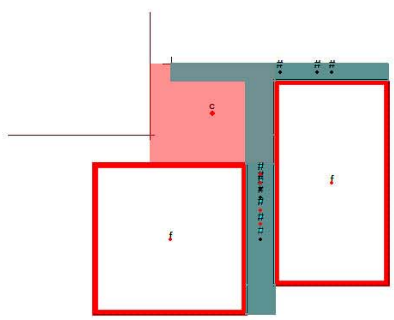


Figure: Concepts generated

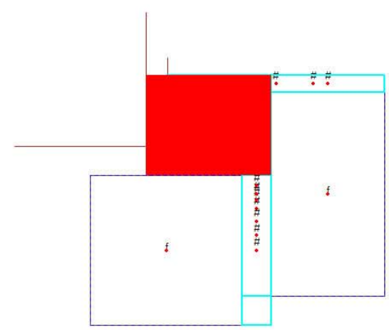
Sketch – 2



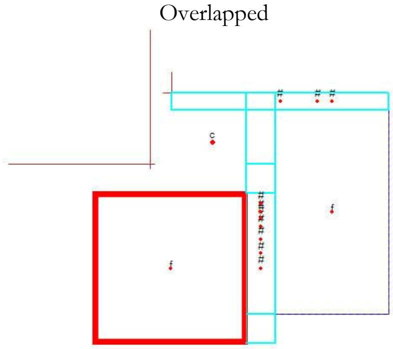
TRIGGER uses a simple Site-Plan context (Building-Schema, Courtyard-Schema and Street-Schema) to interpret this sketch. The following concepts are generated. (Notice that the courtyard and the streets are emergent ideas).



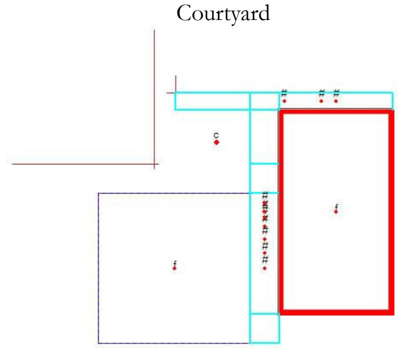
Overlapped



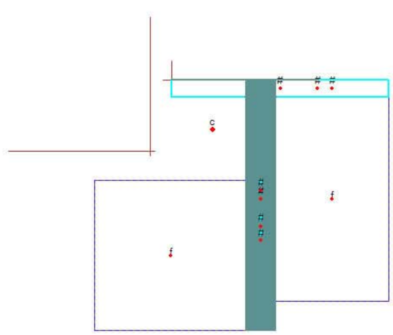
Courtyard



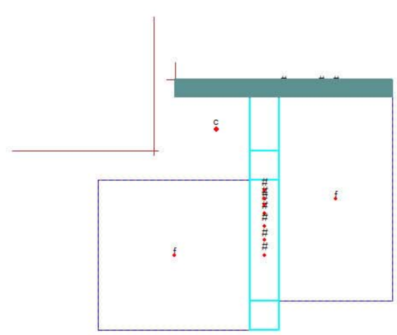
Building



Building

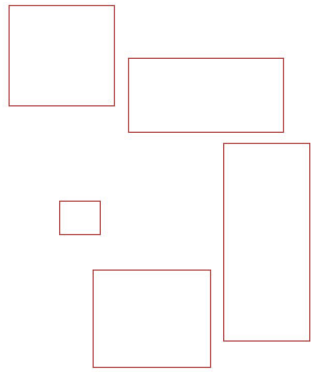


Street

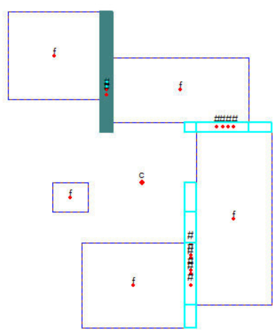


Street

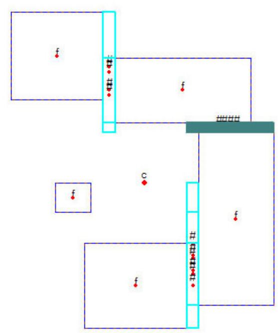
Sketch – 2 (extended)



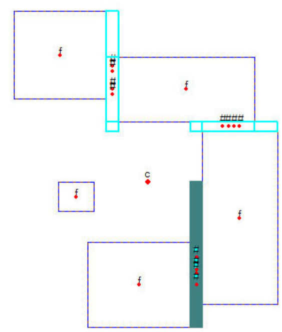
The following concepts are generated (in the Site-Plan context),



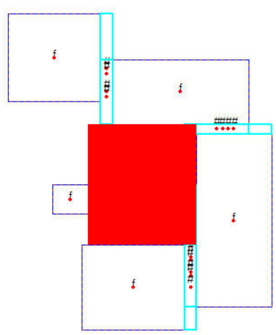
Street



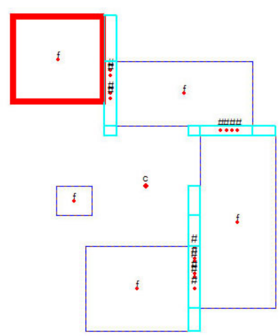
Street



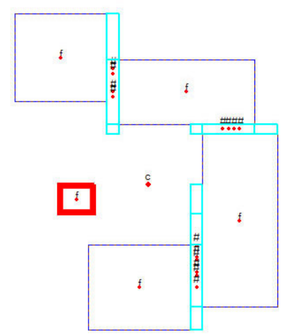
Street



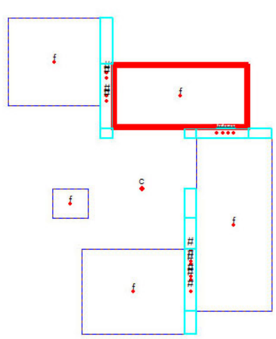
Courtyard



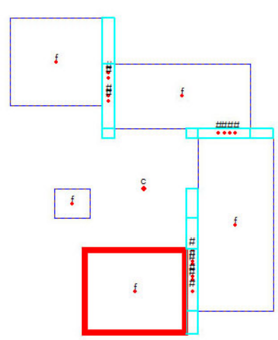
Building



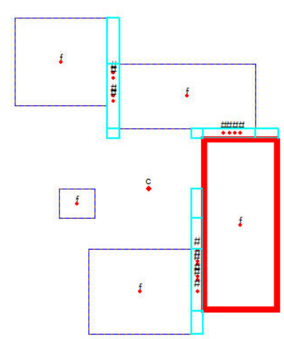
Building



Building



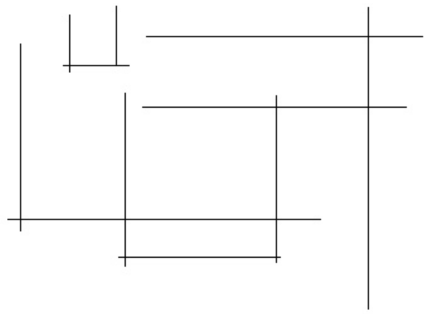
Building



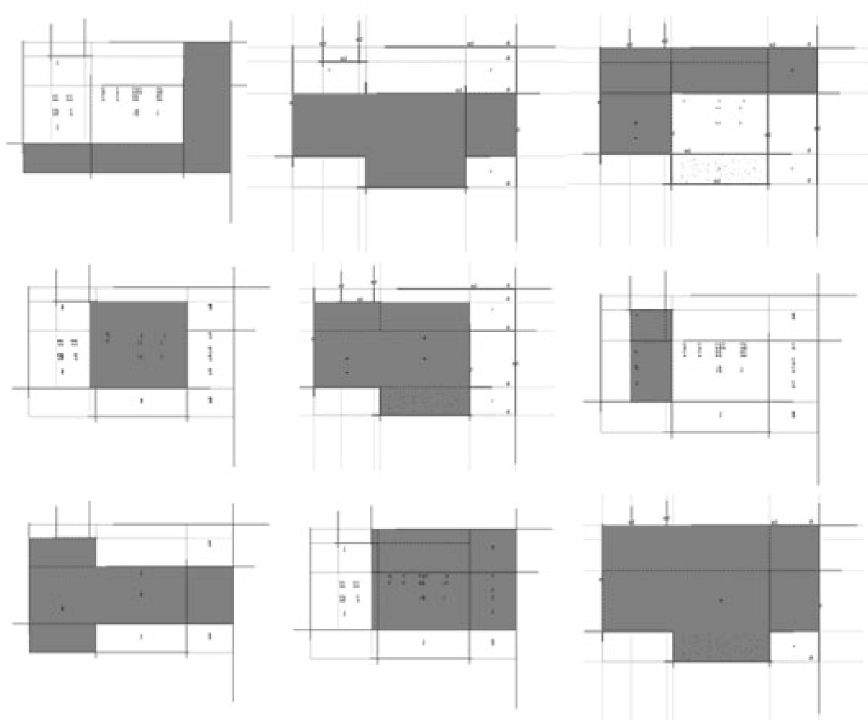
Building

6. Discussion

6.1 TRIGGER as a design exploration tool

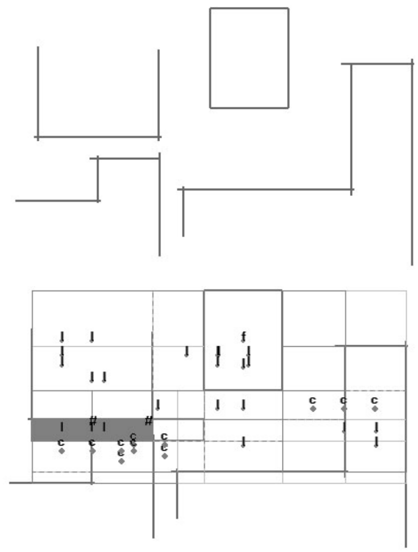


TRIGGER makes explicit a large set of possibilities from simple sketches. Very often some of these possibilities can surprise the human designer and provide interesting opportunities for exploration.



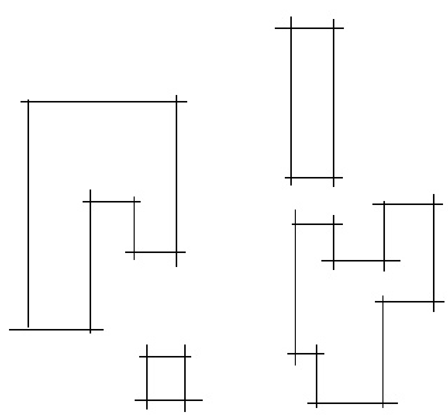
It is interesting to note that the 'solution space' is not defined by the abstraction space (or the sketch) but the memory of TRIGGER. This model closely resembles the interaction between the designer and the sketch, where the solution space is not really defined by the sketch but the designer's mind.

6.2 Sketches with Performance Criteria



TED, I FOUND A NARROW ALLEY..DO YOU WANT TO CONTINUE...

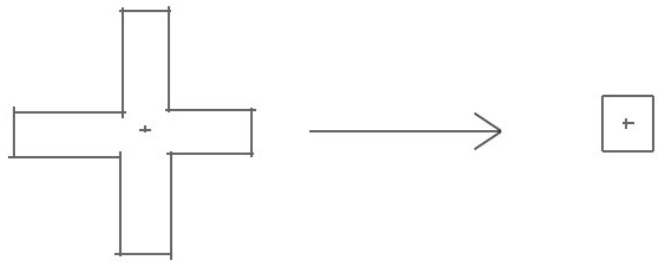
TRIGGER can be augmented with evaluation modules, which can interrupt the designer when it finds 'failure' or 'bad' configurations. Also the action-rules associated with the concepts can be used to *automatically* drive design generation. The human designer can make use of this feature to *look-ahead* into possible high-performance design moves. This is aligned with Nicholas Negroponte's vision of the Architecture Machine. [Negroponte 1970]



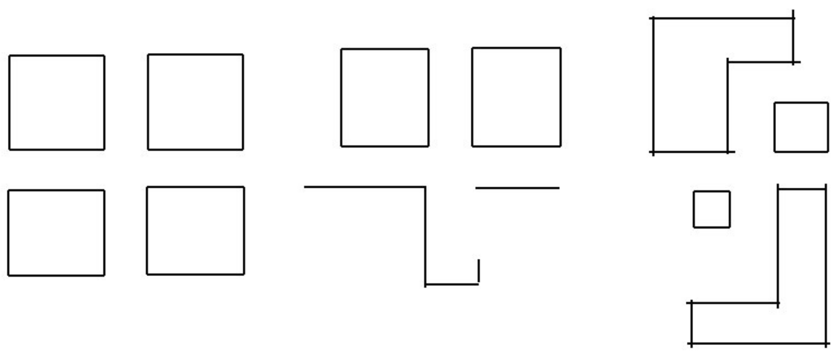
TED, THE DISTANCE BETWEEN THE COURTYARD AND THE ENTRANCE IS MORE THAN 30 FT… CONTINUE?

6.3 Shape Grammar Implementation

TRIGGER releases the dependence on structure; hence shape rules like the one below does not require an exact structural match to trigger.

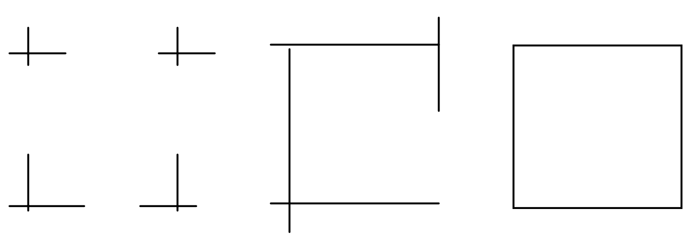


With an appropriate schema, TRIGGER can find the cross-concept in all the following three shape assertions,



6.4 Schema Recognition and Pattern Recognition

There is a lure of comparing visual interpretation with pattern recognition algorithms. While sketching designers can conceptualize beyond the obvious geometry of the sketch. For example the concept of 'enclosure' can be triggered from any of the following sketches,



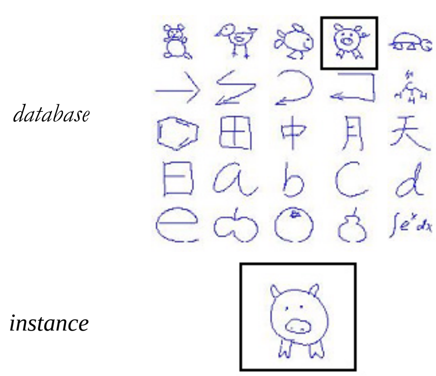
Pattern recognizers would depend on strict structural comparison from a pre-configured database. So to identify a concept, the database has to contain all possible structures of the concept of enclosure. The Electronic Cocktail Napkin project [Gross 1996] attempts to manage a drawing archive using an indexed visual database. However to maintain generality the database would be so large that it would be a serious limitation both in terms of storage and retrieval. Though pattern recognizers are useful and in some ways can emulate the effect of 'seeing', schema-recognition is more efficient in visual interpretation, in the sense that it releases exact geometry associated with instance and manages to encode class-information.

Schemas can be thought of as patterns in one level, but then the word *pattern* is a global metaphor. Even our lives and our thoughts have pattern, so to say. However the subtle differences are in the way we represent knowledge and how we define our pattern. Knowledge in a typical 'pattern' recognizer is just structural and geometric (meaning pixels will be matched with pixels) while in 'schema' recognition knowledge is procedural (deductions based on a set of visual assertions). Instead of a structural comparison, a schema would treat a shape as visual *assertions* and through reasoning generates concepts.

Interpretation via Schemas (Reasoning)

(edge-concepts) triggers -> (direction-concepts) triggers -> (closure-concepts) triggers -> (enclosure-concept)

Recognition via Pattern Matching (Searching)



7. Contributions

In this thesis, I have

- Developed a working model of Visual Interpretation, which manages to capture the notions of 'multiplicity of semantics' and 'ambiguity' in shapes.

- Identified some limitations of the current computation models that constrain design exploration

- Described the key features of 'sketches' as a powerful design tool.

- Elaborated the necessity of non-hierarchic architectures for design exploration.

- Developed the idea of Visual Schemas. They schematically store real world knowledge and form the reasoning structure for visual interpretation.

- Suggested the notion of a visual concept as separate from shapes and developed a multi-modal representation scheme for it.

- Implemented a LISP machine, which can reflect on simple sketches and form context-based conceptions. It can 'see' emergent ideas using reasoning structures.

8. Conclusion

This thesis models 'see-ing' as reasoning structure based on combinatorial algorithms. I do believe that there is some kind of reasoning structure in our mind that allows us to both recognize what we see and attach higher-level concepts to the visual assertions. However, at the end of this project, I do question the dependency on the combinatorial algorithms for 'see-ing. There is a combinatorial explosion of options even with relatively simple collection of visual assertions. With the ease with which we 'see' I feel in addition to the cognitive and perceptual filters that I proposed there must be some other heuristics involved in the process as well. I believe that the dependency on combinatorial algorithms has its roots in the symbolic representations that we use and an inquiry into the basic representation systems is necessary at this juncture. Stephen Larson's thesis on intrinsic representations [Larson 2003] is a promising work and shows a possible direction forward.

In this thesis, I have hard-coded the 'memory' (collection of schemas) of the system to work within the limited time frame. However, this model remains incomplete without an augmented learning system. Ideally the system should be able to generate 'schemas' automatically by interacting with the user. Such a learning interface is by no means a trivial extension and requires a substantial amount of research and enquiry.

9. Bibliography

Alexander, C., Ishikawa, S., Silverstein, M., Jacobson, M., Fiksdahl-King, I., and Angel, S. (1977). *A Pattern Language: Towns, Buildings, Construction.* Oxford University Press, New York.

Gross, M. (1996). *The Electronic Cocktail Napkin - A computational environment for working with design diagrams*. Design Studies, 17(1): pp 53–69.

Holland, J. (1996). *Hidden order*. Redwood City, CA: Addison Wesley Longman Publishing Co., Inc

Klee, P. (1953). *Pedagogical Sketchbook*. New York, NY: Frederick A. Praeger, Inc

Larson, S. (2003). *"Intrinsic Representation : Bootstrapping Symbols from Experience"*, Masters Thesis, Massachusetts Institute of Technology Department of Electrical Engineering and Computer Science.

Lenat, D. (1998). *The Dimensions of Context-Space*. Austin, TX: Cycorp.

Minsky, M. (1974). *A Framework for Representing Knowledge*. Massachusetts Institute of Technology Artificial Intelligence Memo No 306.

Minsky, M. *(Draft). Emotion machine.*

Minsky, M. (1979). *K-Lines: A Theory of Memory*. Massachusetts Institute of Technology Artificial Intelligence Memo No 516.

Minsky, M. (1985). *Society Of Mind*. New York, NY: Simon and Schuster, Inc.

Minsky, M. (Ed.) (1968). *Semantic Information Processing*. Cambridge, MA: MIT Press.

Minsky, M., and Papert, S. (1969). *Perceptrons: An Introduction to Computational Geometry*. Cambridge, MA: MIT Press

Negroponte, N. (1970). *The Architecture Machine*. Cambridge, MA: MIT Press

Papazian, P. (1991). *Principles, Opportunism and Seeing in Design: A Computational Approach*. Massachusetts Institute of Technology Artificial Intelligence Memo No 1309.

Rao, S. (1998). *Visual Routines and Attention. PhD Dissertation, Massachusetts Institute of Technology.*

Schon, D. A. (1987). *Educating the reflective practitioner: Toward a new design for teaching and learning in the professions*. San Francisco, CA: Jossey-Bass

Siebel, P. (2003). *Practical Common Lisp*. Berkeley, CA: Apress, Inc

Simon H. (1981). *Sciences of the Artificial (2nd Edition)*. Cambridge, MA: MIT Press

Stiny, G. (2001). *How To Calculate With Shapes, Formal Engineering Design Synthesis, 20-64*. New York, NY: Cambridge Univ. Press

Stiny, G. (1994). *Shape rules: closure, continuity, and emergence*. Environment and Planning B; Planning and Design 1994, volume 21, pp s49-s78

Ullman, S, Vidal-Naquet, M and Sali, E. (2002). *Visual features of intermediate complexity and their use in classification*. Nature Neuroscience, Vol 5 no 7.

Ullman, S. (1996). *High Level Vision.* Chapter 9, pp. 263-315. Cambridge, MA: MIT Press

Winograd T, Flores F. (1987). *Understanding Computers and Cognition: A New Foundation for Design.* Reading, MA: Addison-Wesley

Winston, P. (1977). *Artificial Intelligence.* Reading, MA: Addison-Wesley Publishing Company, Inc.