Chapter 1

Introduction

1.1 Background to the problem

The advent of computers has changed how people work and how jobs are done in many professions. This is because in many circumstances computers can be used to handle the routine (and often boring) aspects of some jobs, freeing the professional to devote her/his time to the more mentally stimulating/challenging aspects of the work. Baase [1997] discusses many of the benefits and problems of the computer revolution in her book *A Gift of Fire*.

The general area of architecture (designing houses or office buildings, town planning, etc.) is an area where computers could offer benefits to the professional. In some other areas of design this is already happening. For example, shape grammars can be used to understand designs and to create new designs [Koning and Eizenberg, 1981; Mitchell, 1990; Herbert *et al.*, 1994] and computers have been used to automatically generate floor plans for houses [Eastman, 1972; Rinsma *et al.*, 1990]. Town planning or urban design is an area where the professional uses large data sets in order to be able to understand existing town plans and also to be able to improve existing layouts and to design new layouts. Using computers to assist the town planners could free the town planner from some aspects of the routine jobs and thus allow her/him to concentrate on the design phase. This automation raises a number of societal and ethical issues, which will not be discussed here (see Baase [1997] for more detail), but also raises a number of problems that are of interest to computer scientists. This thesis considers some of the interesting computational problems that arise from the possible automation of a design task – that of understanding and designing urban layouts.

Hillier *et al.* [1983] proposed a method, which they called *space syntax*, to describe and analyse patterns of architectural space both at the building and urban level. Hillier [1996] discusses the use of the method in more detail (see particularly pages 153 to 181). The idea is that with an objective and precise method of description it is possible to investigate how well environments work, rigorously relating social variables to architectural forms. They believe that space syntax can help architects to understand the interaction between space and society and thus can be used as a tool to understand why urban areas have developed as they have and also to design new urban layouts.

An architect or town planner would apply the space syntax method ([Mills, 1992]) to a town or city in 4 main steps.

1. Studying the town plans or an aerial photograph of the town and separating out the "space" (roads, parks, etc.) from the "non-space" (buildings, car parks, schools, etc.). The result of this step would be a "deformed grid" which is the town plan reduced to a number of polygons representing the "non-space" separated by "space" – it is the space that is the real object of interest.

2. Creating a convex map of the area. This convex map is made up of the smallest number of "largest" convex spaces that cover all of the space in the area being analysed. It is a partition of the space into the minimum number of convex polygons. A convex space gives some local information about the area in the sense that every point in the convex space is directly visible and directly accessible to every other point in that convex space.

3. Creating an axial map of the area from the convex map. The axial map is made up of the fewest and longest straight line segments (axial lines) that cover the town, crossing through the convex polygons that make up the convex map, and offers a globalising perspective that takes into account how far one can see (or walk) in the town.

4. Combining the information from the convex map and the axial map to produce an integration factor for the town/city. The integration factor, which is the final result of the analysis, gives an idea of how easy it is to move about in the town.

At present most of this work is done manually by the architect/town planner using pencil and tracing paper over the aerial photograph or map. This seems to be an application where computers can be used to assist, or (in some aspects of the work) replace, the architect in performing the routine tasks required to prepare the data for interpretation. Some tasks are boring and can be more efficiently solved by computers. Some tasks are computationally difficult and are unlikely to be performed optimally by humans so computers could be utilised to provide improved solutions. In order to determine whether the processing can be done automatically it is important to consider the tasks that are performed by the architect where there is potential for automation. The next section of this thesis considers the tasks performed in applying the space syntax analysis method, identifies where there is the possibility for automation and also highlights some of the areas of current and possible future research interest for computer scientists. Note that the focus of this thesis is on the computational problems that could arise with automation. The issue of whether space syntax is a good way of understanding and doing design is not within the scope of the research.

1.2 The scope for automation

The first stage in the process of applying space syntax is the separating of space from non-space in the town plan or aerial photograph. This problem has already been the subject of much research in the field of image processing over a number of years. The general approach here would be to take an aerial photograph, landsat image, or any other image (in digital form) over the inhabited areas and to automatically separate space (roads, parks, etc.) from non-space (buildings, etc.) using image segmentation techniques. Gonzalez and Wintz [1987], Gonzalez and Woods [1992], Castleman [1996], Jähne [1997] or Russ [1999] offer good introductions into the field of image processing and more specifically, general image segmentation techniques. Various authors [Huertas and Nevatia, 1988; Liow and Pavlidis, 1990; Ton *et al.*, 1991; Stilla *et al.*, 1996; Geman and Jedynak, 1996; Barzohar and Cooper, 1996; Levitt and Dwolatzky, 1999] have considered the issues of separating roads or buildings from the background in digital images.

An additional problem that could occur after the separation/segmentation phase is that the segmented areas are unlikely to have smooth boundaries. In order to make these areas suitable for further processing it is necessary to be able to "accurately" and "efficiently" represent each area by a bounding polygon. In essence the area should be described by a bounding polygon that matches the boundary as closely as possible while minimising the number of vertices and edges required to define the polygon. Research into this problem has been going on for a number of years ([Ramer, 1972; Pavlidis and Horowitz, 1974; Sarkar, 1993; Perez and Vidal, 1994; Ruskin, 1997; Zhu and Seneviratne, 1997]). The result of applying segmentation and polygon approximation to instances of the problem would be the "deformed grid" – a number of polygons representing the non-space in the area with the space between as the real area of interest. A bounding polygon can then be put around this area of interest. This deformed grid would then be represented in an appropriate fashion for future processing – finding the convex map and the axial map.

It does, therefore, seem that the initial phase of the space syntax method – separating space and non-space – can benefit from automation. Computer programs can be used to generate a deformed grid from an aerial photograph or town plan. If the town planner feels this automatically deformed grid is acceptable then he/she can use it as is in the next phase of the process. Alternatively he/she could use it as a starting point for generating a deformed grid that they feel is acceptable to use in the next phase of the process. Although many of the problems in this area are well understood there are still some open questions and unresolved issues and there is thus still scope for further research in this area.

Once the deformed grid has been found, the next phase of the work is to find the convex map of the area. The convex map is defined as being the minimum number of non-overlapping convex spaces (convex polygons) that cover the space in the deformed grid. Figure 1.1 shows a section extracted from a map of the Johannesburg region. This is essentially the deformed grid of the region – the space (light coloured) and non-space (darker coloured) in the urban layout represented by polygons. Note that the polygons representing non-space (darker coloured) are not of direct interest in applying space syntax. These non-space polygons can be of arbitrary shape. The polygon[s] representing the space in the deformed grid must be partitioned into convex polygons. Figure 1.2 shows a close up of part of the original region that will be used to give an idea of the application of space syntax. Essentially the problem is that of covering or partitioning a general polygon (the boundary of the area under consideration) with holes (the non-space parts of the area under consideration) by the *minimum* number of convex polygons. The general covering problem (where polygons are allowed to overlap) and the general partitioning problem (where polygons may not overlap) have been quite well researched. In addition partitioning and covering of special classes of polygons has also received a lot of interest. Many of the problems in this category have been shown to be computationally intensive and some have been shown to be NP-Hard. Chapter 2 discusses these problems in more detail. The special case of covering or partitioning a polygon with holes, that represents a town plan, and therefore has special constraints, has not been studied but the complexity of some town plans (as shown in Figure 1.1) would seem to indicate that this problem is also inherently hard. However, even if the problem itself is inherently hard – it takes a long time to find the minimum solution – it is likely that approximations to the optimal solution would be sufficient for the town planner to continue meaningfully with the later phases of the analysis. It might also be the case that some town plans can be modeled by simplifications to the general problem and that these could be solved exactly in a reasonable time. It thus seems likely that this phase of the process could also benefit from automation. Interesting areas of research related to this phase of the process are to study approximation algorithms and special cases of the problem that can be solved exactly in a reasonable time.

From the convex map a town planner can generate the axial map over the area. This axial map is defined as the smallest number of axial lines that will cross all of the shared boundaries between the convex spaces in the convex map. Figures 1.3

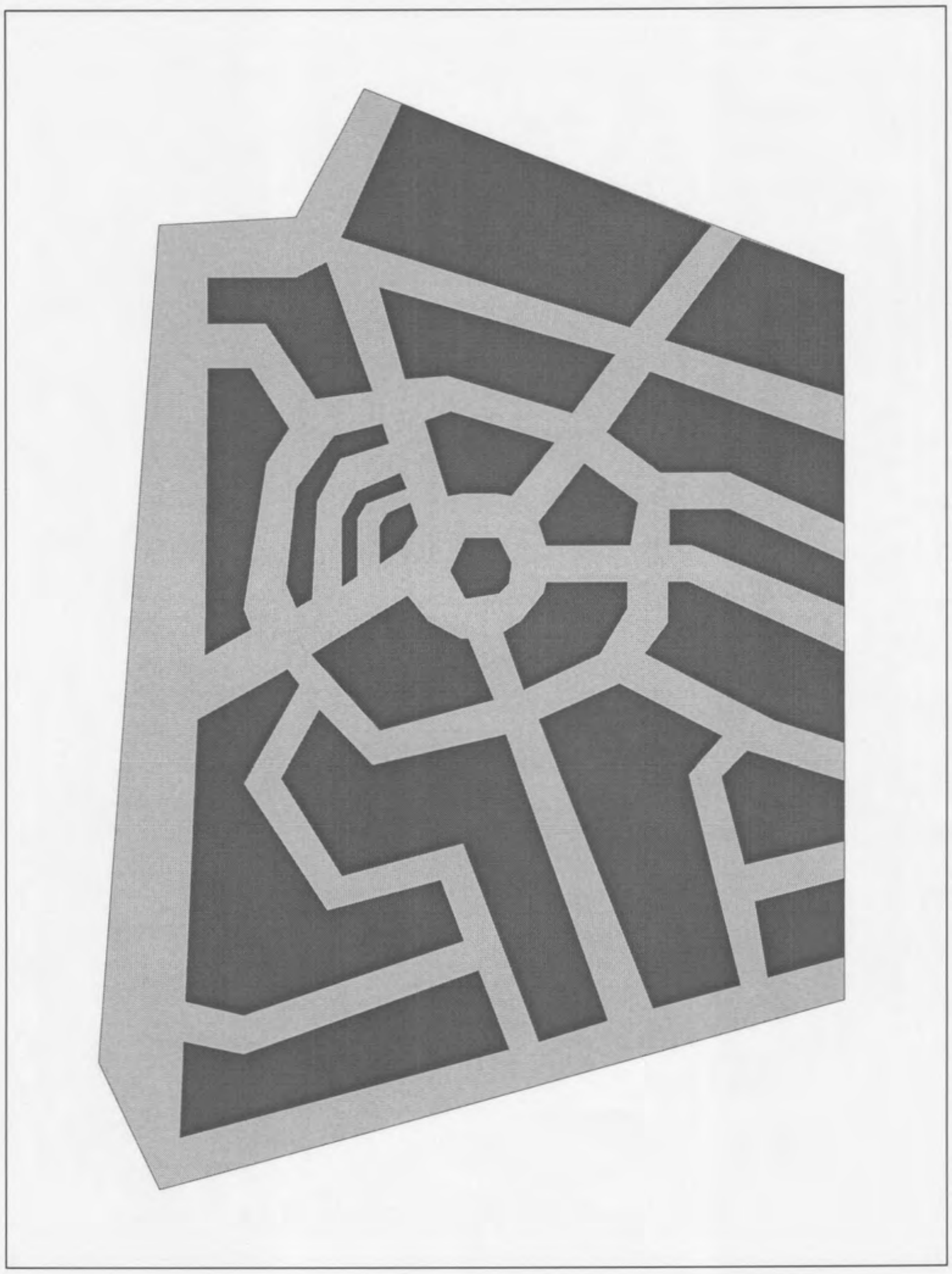

Figure 1.1: An example town plan

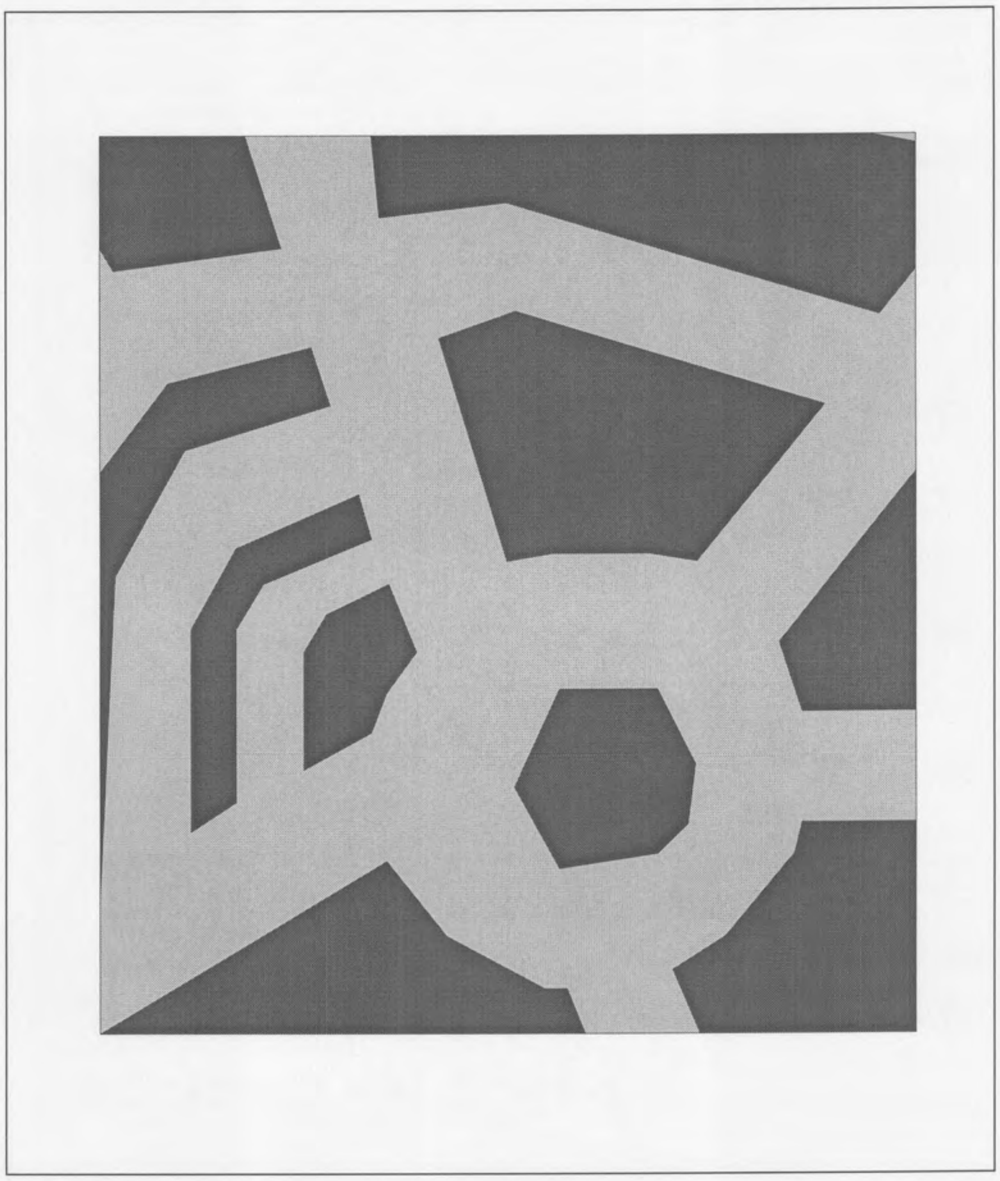

Figure 1.2: A zoomed view of a portion of the example town plan

and 1.4 show how the enlarged portion of the urban layout from Figure 1.1 could be covered by convex polygons and the shared boundaries between the convex polygons crossed by axial lines. It is interesting to note that in the two figures the same number of convex polygons are required to cover the spaces in the layout but that the convex polygons are different. An effect of the different sets of convex polygons in the two partitions is that different numbers of axial lines are required to cross the shared boundaries in the two figures (6 axial lines are required in Figure 1.3 and 7 in Figure 1.4). This example illustrates the inherent difficulty of the problem. It is known to be difficult to find the minimum number of convex polygons to partition a polygon and it seems that is it also difficult to partition the space in a town plan. Then to find the minimum number of axial lines to cross all of the shared boundaries it is necessary to consider all of the combinations of the minimum number of convex polygons and to find the minimum number of axial lines for each of those configurations.

The problem of placing the minimum number of axial lines for any configuration of convex polygons is an area that has not been directly researched in the past. In fact, it was first studied by the author of this thesis. There is, however, an abundance of research in closely related areas – guarding and visibility problems. In guarding problems the aim is to place the minimum number of guards (with different attributes) so that the guards can see the entire area of polygons of different forms. Visibility problems are very similar but focus on determining how much of a polygon can be seen from some point or points inside the polygon. These problems are discussed in detail in Chapter 2. The literature in these areas indicates that finding the minimum number of axial lines could itself be an inherently difficult problem. The problem of finding the axial map from some urban layout (given the convex map) is the major emphasis of this thesis. A number of variations the problem are studied and even more questions are raised. The results presented in this thesis do, however, indicate that this problem can be solved sufficiently well to be of use to the town planner or urban designer.

Having determined the convex map and the axial map the architect would perform the space syntax analysis based on these parameters. This phase of the process is already automated [Hillier, 1996]. This is done essentially using graph algorithms. An area of interest is in determining whether the graph algorithms used here – for example, shortest path algorithms – could be modified for the specific application. In addition, issues related to space usage and representation could be attractive areas for research.

The discussion above makes it clear that the aim of automatically applying space syntax to a town plan or aerial photograph of a town poses many interesting research areas for computer scientists. The range of research areas is too broad to be covered as a single Ph.D. thesis and thus this thesis concentrates on a small part of the overall problem. The next section gives an overview of the research undertaken for

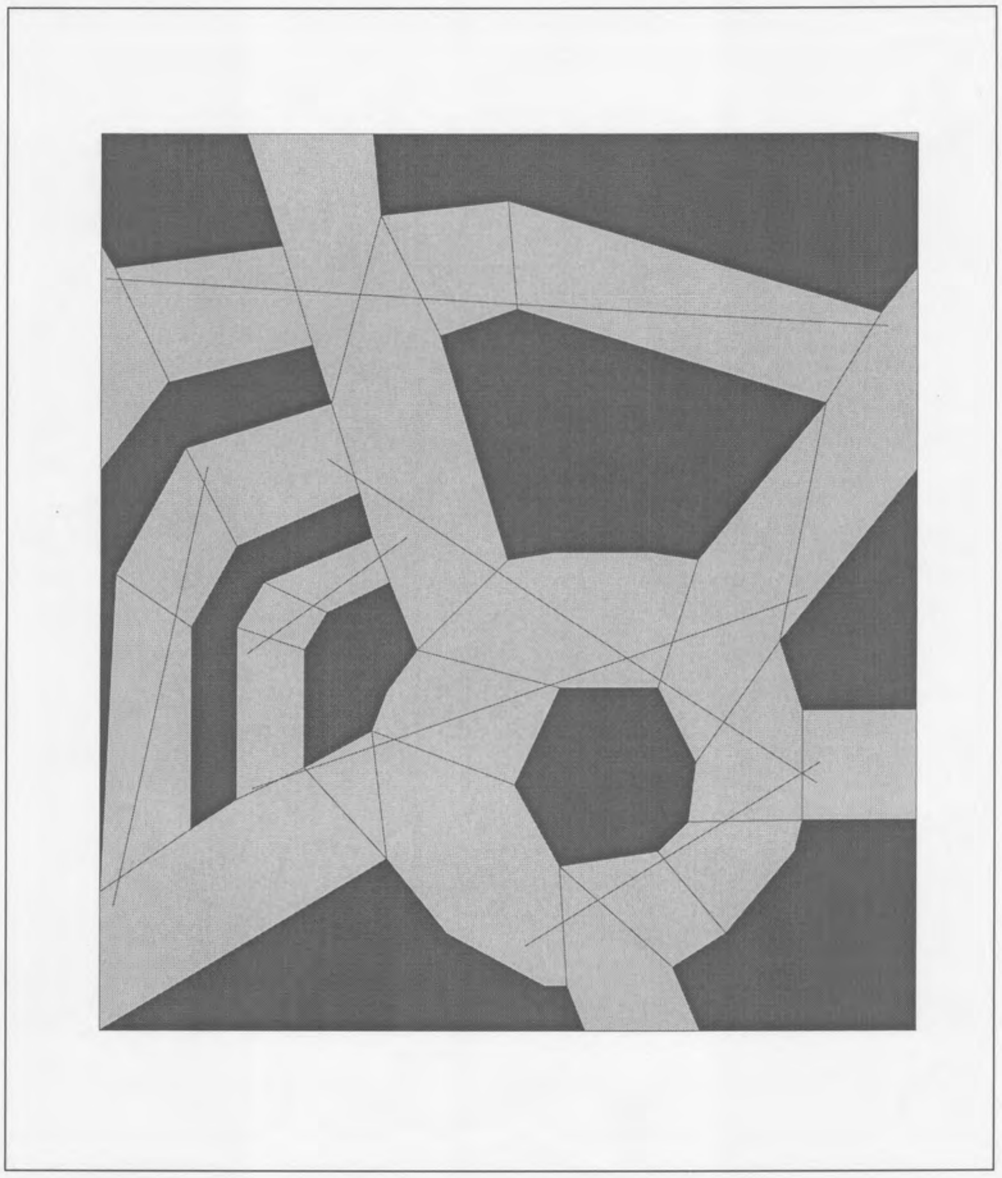

Figure 1.3: A convex map of the enlarged version of the town plan with 24 convex spaces and its associated axial map with 6 axial lines

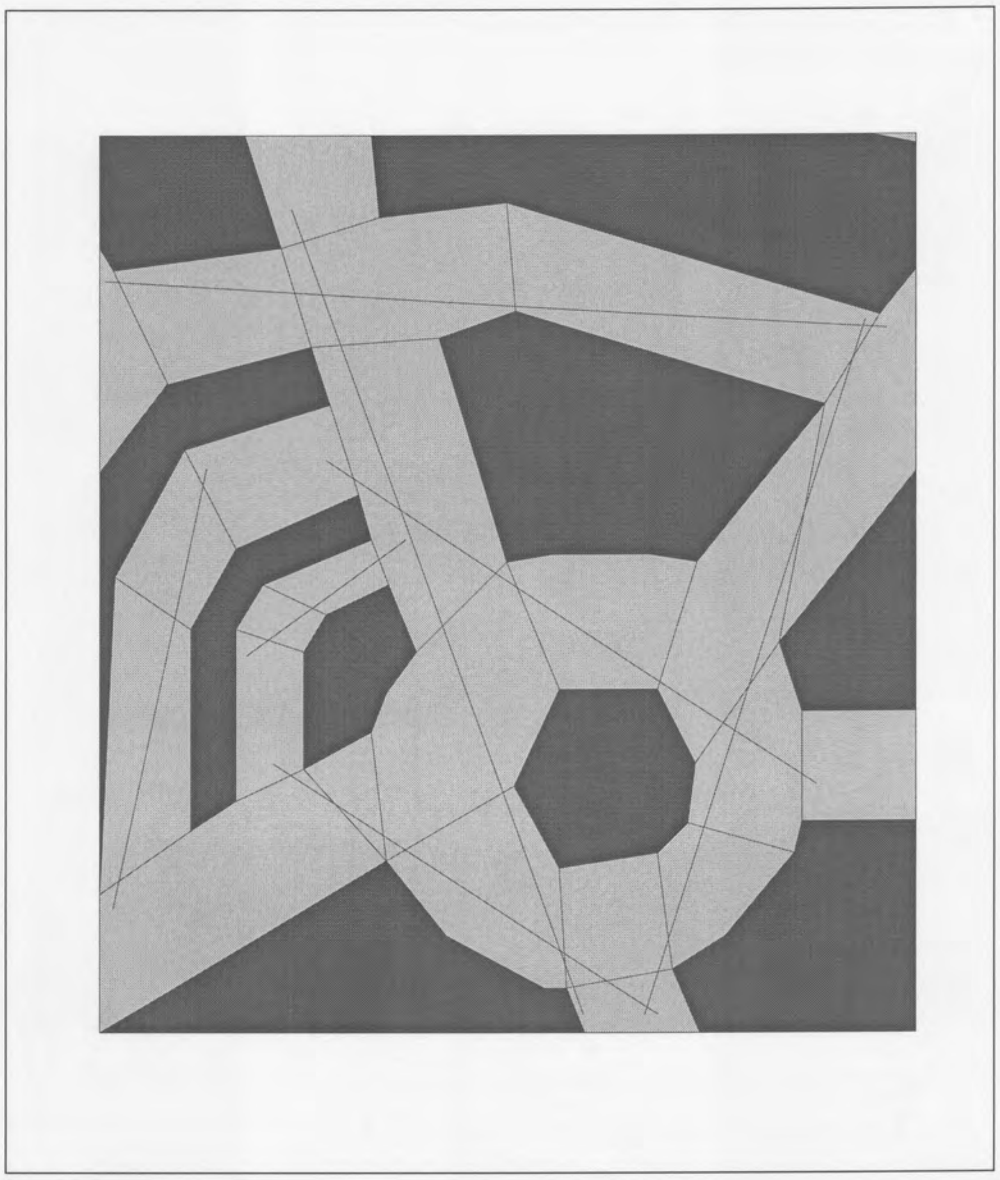

Figure 1.4: A convex map of the enlarged version of the town plan also with 24 convex spaces and its associated axial map with 7 axial lines

the thesis.

1.3 The research focus of the thesis

As discussed in the previous section of this thesis the potential automation of space syntax gives rise to many areas of research. These areas include image processing, computational geometry and algorithms. The range of research questions that could be addressed is very wide and it was thus necessary to concentrate on a subset of the problems. For this reason, the problems of separating space from non-space, determining the convex map and the final analysis stage with its associated algorithms were not considered as part of this research. The decision was made to focus the research for this Ph.D. on the problem of finding the axial lines that cross all of the shared boundaries between the convex polygons in the convex map, that is finding the axial map for a given layout. In the remainder of this thesis the problem will be called the *Axial Line Placement* problem or ***ALP***. This problem on its own is still very big and could not be solved entirely. The research considered a number of simplifications to and variations on ***ALP*** – using simpler types of convex polygons and introducing constraints aimed at making the problems easier to solved. Some new contributions have been made as a result of this research and some progress has been made towards finding solutions to the general problem. There are, however, still a number of unsolved problems and open questions. The next section gives an overview of the thesis, in particular indicating some of the variations of the *Axial Line Placement Problem* that have been considered as part of the research for this thesis.

1.4 Overview of the remainder of the thesis

Many researchers over the years have concentrated on the problems of guarding and visibility in polygons of various shapes. Much attention has also been focussed on the problems of decomposing polygons into smaller more easily handled components. As mentioned in Section 1.3 above these problems have relevance to the current research because they can provide insight into solving ***ALP*** . Chapter 2 of this thesis presents a detailed literature survey of the work in these areas.

Chapter 3 expands somewhat on the range of research possibilities that arise from the problem of finding the convex and axial maps for town plans (urban layouts). This chapter also presents some simplifications or generalisations of the problem that are in themselves interesting problems for further study. The chapter concludes by discussing the specific problems that were tackled as part of this thesis.

The new results, in the form of proofs, algorithms, etc. that arise from tackling these problems constitute the research contribution of the thesis and are discussed in ensuing chapters of the document.

Chapter 4 discusses one of the "simplifications" presented in Chapter 3 in depth – the placement of orthogonal axial lines to cross the shared boundaries between rectangles in a configuration of orthogonal rectangles. The problem is presented and shown to be NP-Complete. A heuristic algorithm that appears to give "acceptable" approximations is tested and compared to a heuristic algorithm that is known to give a solution no worse than twice an optimal solution. Special cases where the problem can be solved exactly in polynomial time are also discussed.

In Chapter 5 the problem discussed in Chapter 4 is changed slightly to allow the axial lines that cross the shared boundaries between rectangles in a configuration of orthogonal rectangles to have arbitrary orientation. This problem is also shown to be NP-Complete. The chapter also introduces some ideas for heuristics for finding acceptable approximations to the exact solution in this case.

The restrictions on problem are relaxed even further in Chapter 6 that considers the axial line placement problem for arbitrary convex polygons (dropping the requirement of orthogonal rectangles). This problem is a generalisation of the problem described in Chapter 5 and thus the NP-Completeness of this problem can be easily proved. The variation of the problem discussed in this chapter is the most general case of ***ALP*** and thus ***ALP*** is NP-Complete.

In Chapter 7 of the thesis, the original problem – placing axial lines to cross the shared boundaries of the convex polygons in the convex map of an urban layout or town plan – is considered. In this thesis the problem of finding the convex map of the area under consideration is not studied in great depth because it has already been shown to be NP-Hard. However in this chapter it is studied as a side issue in the matter of finding the axial map of the area. Here it is shown that although the general problem of partitioning a polygon with holes is NP-hard, there are some instances of the problem that can be solved in polynomial time. In particular it is shown that if the town plan is regular then the convex map can be found in polynomial time and so can the axial map. The chapter also addresses the matter of town plans that are not regular and argues that finding the convex map of such layouts is likely to be NP-Hard but that finding the axial map of such layouts might not be as difficult.

The final chapters of this thesis are devoted to future work and concluding remarks. Chapter 8 discusses some of the problems that have not been tackled in this thesis and Chapter 9 restates the results and conclusions drawn in this thesis. There are still many open questions and unproven conjectures but this thesis has made a significant contribution in tackling some new problems and obtaining some new results.

Chapter 2

Background

2.1 Introduction

As discussed in the introduction to this thesis (Chapter 1) the potential automation of space syntax gives rise to a wide range of research questions and it was thus necessary to concentrate on a subset of the problems. For this reason, the problems of separating space from non-space, determining the convex map and the final analysis stage with its associated algorithms were not considered as part of this research. The decision was made to focus the research for this PhD on ***ALP***, the problem of finding the axial lines that cross all of the shared boundaries between the convex polygons in the convex map. This problem has much in common with other well studied problems in the field of computational geometry.

The most obvious commonality is in the idea of *visibility*. The intent of the axial map of some urban layout is that it offers a globalising perspective that takes into account how far one can *see* (or walk) in the town. In particular placing an axial line to cross the adjacencies between a number of adjacent convex polygons can be thought of as determining a line of sight from some point to another in a given polygon. Visibility in polygons is an area of computational geometry which has received much attention in the last two decades [Asano *et al.*, 1999]. The essential question is this work is: “Can some point in the polygon ‘see’ some other point in the polygon?” However, other visibility questions can also be posed. Included in these are vertex-vertex visibility, vertex-edge visibility, edge-edge visibility, etc.

The problem of *guarding* a polygon is very closely related to visibility in polygons. In fact, all guarding problems are essentially visibility problems. The first “guarding problem” came about as a problem posed by Victor Klee in response to a request by Vasek Chvátal [O’Rourke, 1987]. The original problem was to determine the minimum number of guards necessary to cover the interior of an n-wall art gallery. Many variations on this problem can now be found in the literature. (See the monograph by O’Rourke [1987], the survey papers by Shermer [1992] and

Urrutia [1999], and the summaries of results by O'Rourke [1997] and Suri [1997].) These variations include vertex guards, edge guards, point guards, periscope guards and prison guards.

Another area of computational geometry which in some ways is similar to both visibility and guarding problems is *polygon decomposition* [Keil, 1999]. The focus here is in decomposing given polygons into smaller more easily manageable parts. Polygon decomposition problems occur frequently in such areas as pattern recognition, image processing, computer graphics and VLSI. In polygon decomposition problems the aim is to break the polygon down into constituent parts which can be more easily processed. Polygon decomposition is categorised in two different ways – covering and partitioning. The basic covering problem is to find the minimum number of polygons, with some predefined characteristics, to cover the complete area of an enclosing polygon. This problem has some commonality with the guarding problem – covering a polygon with the minimum number of polygons of some specified type is equivalent to the placement of the minimum number of guards of some specified type so that each point inside the polygon is visible to some guard. The partitioning problem is the same as the covering problem except that the polygons into which the enclosing polygon is decomposed are not permitted to overlap. The phase of the space syntax analysis method which produces a convex map of a given urban layout is clearly a polygon decomposition problem. The town plan is a polygon with holes and the convex map is a minimum partition of that polygon.

The focus of this chapter is to explore the commonalties (and some cases the differences) between ***ALP*** and the work which has been done in the areas of visibility, guarding and polygon decomposition. This is accomplished by looking at the results which have been published in the other areas and relating these to ***ALP***. Before discussing the previous results, however, it is worthwhile introducing some general terms which are used in the literature. Section 2.2 introduces terms which are important for understanding the research in these areas which is discussed. Other more specific terms are introduced only when required.

In addition, a number of problems which are discussed in this chapter have been proved to be NP-Complete or NP-Hard. In addition, the new results which are presented in this thesis rely on a number of NP-Completeness proofs. It thus seems appropriate that the results concerning NP-Complete and NP-Hard problems are summarised here (Section 2.3). Because of the size and the complexity of the topic the presentation here focusses on the more practical aspects of the use and application of the theory.

Once this general background has been covered, the chapter addresses the more directly related background literature and discusses its relevance to ***ALP*** (Section 2.4). The chapter concludes (Section 2.5) by reiterating that ***ALP*** is different from the other problems discussed. ***ALP*** is thus a previously unstudied problem and is worth being investigated. This discussion leads on to the posing of the research questions in Chapter 3.

2.2 Terminology

The general definitions in the fields of computational geometry and graph theory appear below to make the material covered in this chapter more understandable. In addition, many of these definitions are also used in later chapters of the thesis. Most of the definitions come from Manber [1988], O'Rourke [1987], and Shermer [1992] but can be found in other sources as well. A reader who is familiar with the area could skip this section of the thesis.

- A *point* p is represented by a pair of coordinates (x, y) in euclidean space.
- A *line* is represented by two points p and q (which can be any two distinct points on the line) and is denoted $-pq-$.
- A line is *orthogonal*, or *orthogonally aligned*, if it is parallel to one of the Cartesian axes.
- A *line segment* is represented by a pair of points p and q where the points are the endpoints of the line segment and is denoted by pq.
- A line segment is *orthogonal*, or *orthogonally aligned*, if it is parallel to one of the Cartesian axes.
- A *ray* is represented by a pair of points p and q where one point is the endpoint of the ray and the other point is any other distinct point on the ray.
- A ray is *orthogonal*, or *orthogonally aligned*, if it is parallel to one of the Cartesian axes.
- A *path* is a sequence of points $p_1, p_2, \ldots, p_n$ and the line segments joining them.
- The line segments in a path are called *edges*.
- A *closed path* is a path whose last point is the same as its first point.
- A closed path is also called a *polygon*.
- The points defining the polygon are called the *vertices* of the polygon.
- A *polygon* P can also be defined as a collection of n vertices, $v_1, v_2, \ldots, v_n$, and n edges, $v_1v_2, v_2v_3, \ldots, v_{n-1}v_n, v_nv_1$.

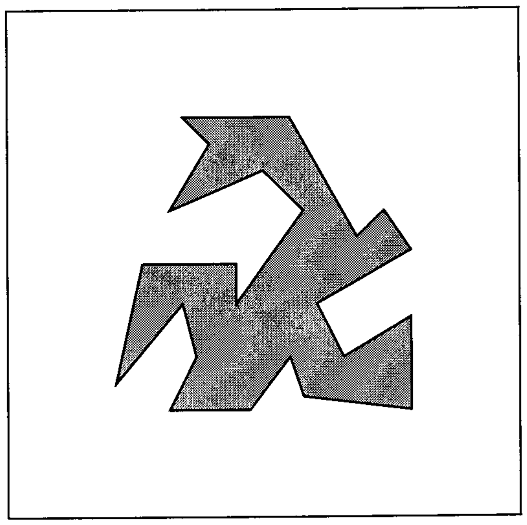

Figure 2.1: A simple polygon (Shermer [1992])

- A *simple polygon* is a polygon where no two non-consecutive edges intersect. Figure 2.1 shows an example of a simple polygon.
- The set of points in the plane enclosed by a simple polygon forms the *interior* of the polygon.
- The set of points on the polygon itself forms the *boundary* of the polygon.
- The set of points surrounding the polygon forms its *exterior*.
- A *hole* in a simple polygon P is another polygon H enclosed by the boundary of P.
- If a simple polygon P contains holes then P is said to be *multiply connected*; if P contains no holes then it is said to be *simply connected*.
- A simple polygon is *convex* if, given any two points on its boundary or in its interior, all points on the line segment joining them are contained in the polygon's boundary or interior.
- A polygon P is a *star* or *star-shaped* if there is some point x in the polygon from which every other point in the polygon can be seen.
- A *kernel* in a star polygon is a point x in the polygon from which every other point in the polygon can be seen.
- A *comb* polygon is as shown in Figure 2.3. Comb polygons exist for any number of vertices which is a multiple of 3.

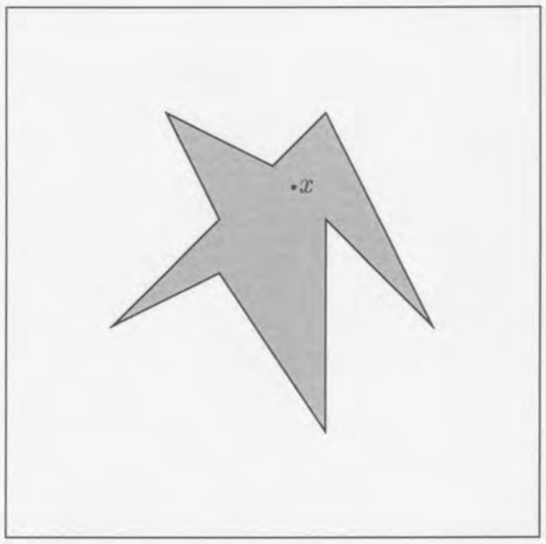

Figure 2.2: A star polygon – x is a kernel of the polygon

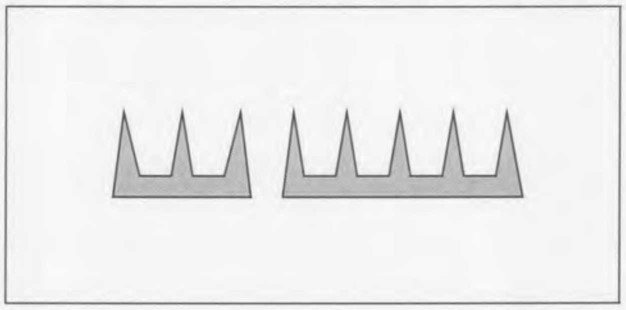

Figure 2.3: Comb polygons (Shermer [1992])

- A *quadrilateral* is a simple polygon with four edges and four vertices.
- An *interior angle* in a polygon is the angle between two consecutive edges of the polygon on the inside of the polygon.
- A *rectangle* is a quadrilateral where all four interior angles are right angles. The pairs of opposite sides of the rectangle are equal in length. If all four sides are of equal length then the rectangle is a square.
- If all of the edges of a rectangle are orthogonally aligned then this rectangle is referred to as an *orthogonal rectangle*.
- A polygon P is an *orthogonal polygon* if all of its edges are parallel to the major axes.
  Note: These polygons have commonly been called "rectilinear" polygons in the literature but O'Rourke [1987] prefers to call them orthogonal because

"rectilinear" (as was pointed out to him by Grunbaum) has a well established meaning: "characterised by straight lines". Orthogonal polygons have also been called *isothetic* polygons [Wood, 1985]. In this thesis O'Rourke's nomenclature is used.

- A *trapezoid* or *trapezium* is a quadrilateral that has one pair of opposite sides parallel, the other pair being nonparallel.
- A vertex v is a *reflex vertex* if it has interior angle $\geq$ 180 degrees.
- A polygon P is *orthogonally convex* if it is orthogonal and any horizontal or vertical line (that is not co-linear with an edge) intersects the boundary of P in at most two points. Figure 2.4 shows two orthogonally convex polygons. The second polygon is an orthogonally convex star.

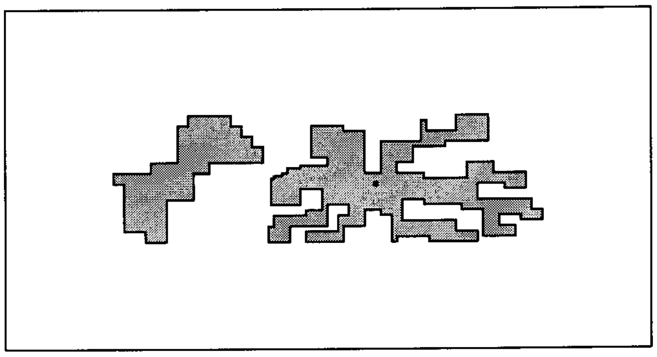

Figure 2.4: An orthogonally convex polygon and orthogonally convex star (Shermer [1992])

- A polygon P is *horizontally (vertically) convex* if any horizontal (vertical) line that is not co-linear with an edge intersects P in at most two points.
- A path inside a polygon P is orthogonally convex if it consists of orthogonal segments and any horizontal or vertical line that is not co-linear with a segment intersects the path in at most one point.
- An *orthogonal comb polygon* is as shown in Figure 2.5.
- A polygon P is said to be *covered* by a collection of subpolygons of P if the union of these subpolygons is exactly P. The collection of subpolygons is called a *cover* of P.
- A cover of a polygon P is said to be a *partition* of P if the intersection of each pair of subpolygons in the cover has zero area. (Note, in some work a partition is also called a *decomposition* but this is misleading terminology and is not used here. In this work *decomposition* includes covering and partitioning.)

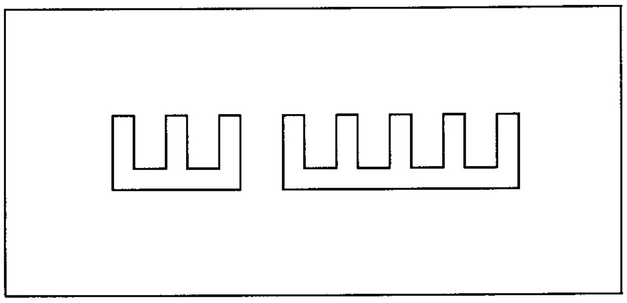

Figure 2.5: Orthogonal comb polygons (Shermer [1992])

- A *triangulation* of a polygon P is a decomposition of the polygon into triangles without adding vertices. This is accomplished by chopping the polygon with diagonals (line segments between nonadjacent vertices). See Figure 2.6 for an example of triangulating a simple polygon.

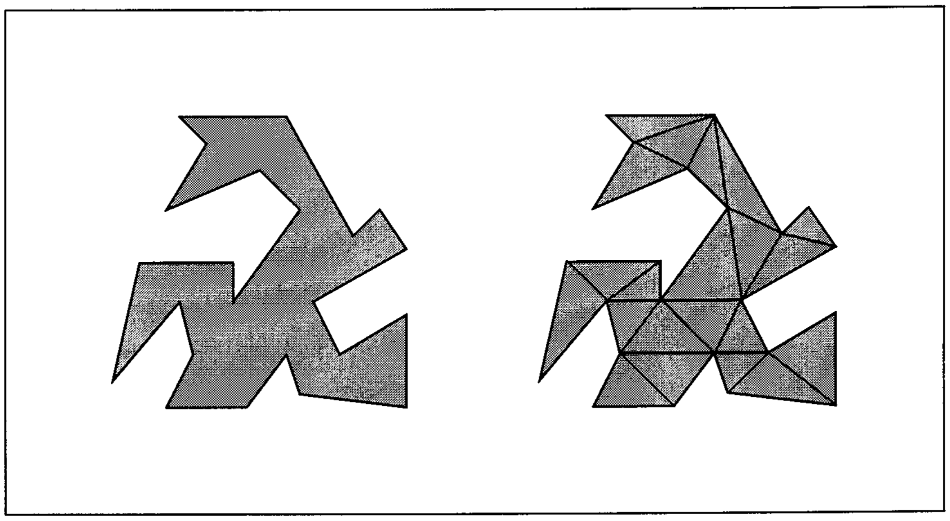

Figure 2.6: A simple polygon and one of its triangulations (Shermer [1992])

- A *Steiner point* is a vertex which is not one of the original points in the polygon. Steiner points are often used in covering and partitioning.
- A *chain* is a sequence $p_1, \ldots, p_k$ of vertices.
- A *reflex chain* of a polygon is a sequence of consecutive reflex vertices.
- A polygon P is *spiral* if it has at most one reflex chain, see Figure 2.7 for an example.

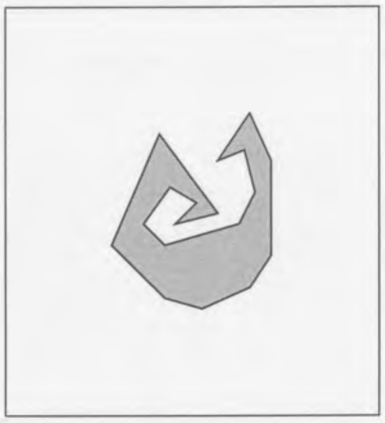

Figure 2.7: A spiral polygon (Shermer [1992])

- Two points in a polygon P are said to be *visible* if the straight line joining them does not intersect the exterior of P [Asano *et al.*, 1999]. Visible points are said to "see" one another. See Figure 2.8 for an example.

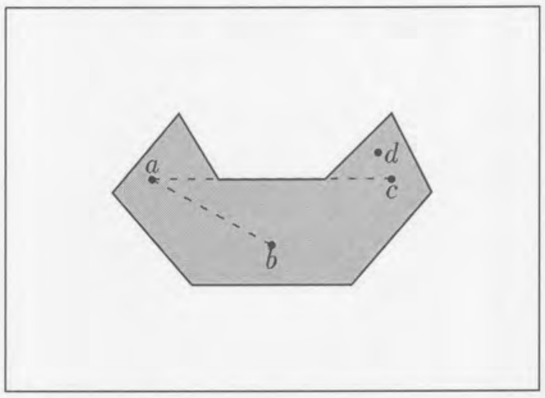

Figure 2.8: Point a can "see" b and c, but not d (Shermer [1992])

- Two points in a polygon are said to be link-j visible (L_j visible) if they can be joined by a path of j or fewer line segments that lie entirely inside the polygon. See Figure 2.9 for an example of L_3 visibility. L_1 visibility is the usual visibility – two points are visible if they can be joined by a path of one segment lying inside the polygon.
- An alternative way to define convexity is to call a set convex if each pair of points in the set is L_1 visible.
- The definition above can be generalised by calling a set link-k convex or L_k convex if every pair of points in the set is L_k visible.

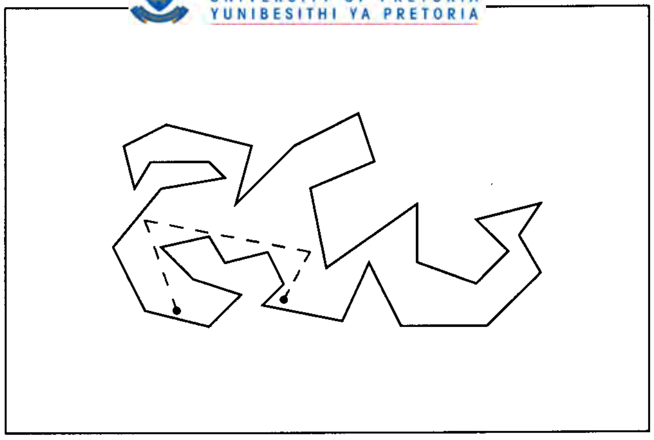

Figure 2.9: A pair of L_3 (or link-3) visible points (Shermer [1992])

- A *visibility polygon* of a point y in some polygon P contains all the points of P visible to y. Figure 2.10 shows the visibility polygon of the point y in the star polygon of Figure 2.2

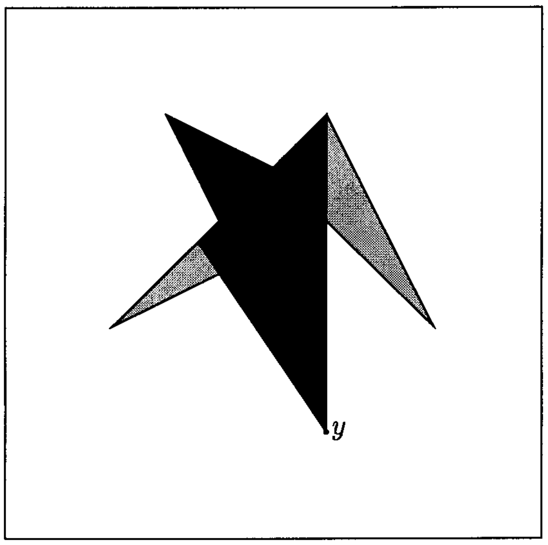

Figure 2.10: The visibility polygon of the point y (shown as the darker shaded subpolygon) – y is the kernel of the visibility polygon but not of the original polygon

- In some contexts, a point in a polygon is identified with a *guard*. A set of such points is called a *guard set*. If all of the elements in a guard set G are vertices of P then G is called a *vertex guard set* and the elements of G are called *vertex guards*. Otherwise G is called a point guard set and its elements are called *point guards*.

- A guard set G is said to cover a polygon P if the union of the visibility polygons of the guards in G is a cover of P. Figure 2.11 shows a covering guard set of a polygon P.

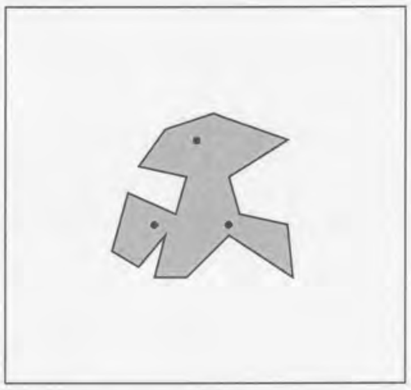

Figure 2.11: A covering guard set (Shermer [1992])

- A *star* polygon can thus be covered by a single guard.
- The *art gallery problem* for a polygon P is to find a minimum-cardinality covering guard set G for P.
- A *mobile guard* is a guard that can patrol an edge or diagonal (or some other line segment) of a polygon.
- An *edge guard* is a mobile guard who is allowed to patrol individual edges of a polygon rather than being restricted to one point.
- The *edge guard problem* then asks for the minimum number of edge guards necessary to cover any polygon of n vertices.
- An edge is called a *guard edge* if a mobile edge guard patrolling along the edge can see every point in the polygon.
- A *diagonal* of a polygon P is a line segment between two non consecutive vertices of P.
- A *chord* of a polygon P is a line segment between two points a and b on two distinct edges of P.
- A *diagonal guard* is a guard capable of moving along an edge or internal diagonal of a polygon [Lu *et al.*, 1998].
- A *chord guard* is a guard capable of moving along a chord wholly contained inside a simple polygon [Lu *et al.*, 1998].

i 16034880
b15432580

- A *line guard* is a guard capable of moving along a line segment wholly contained inside a simple polygon [Lu *et al.*, 1998].

- A polygon which can be guarded by a diagonal guard, chord guard or line guard is called *diagonal-visible*, *chord-visible* or *line-visible* respectively [Lu *et al.*, 1998].

- A *hidden set* is a set of points H in a polygon P such that no pair of points of H is visible. Figure 2.12 shows an example of a hidden set.

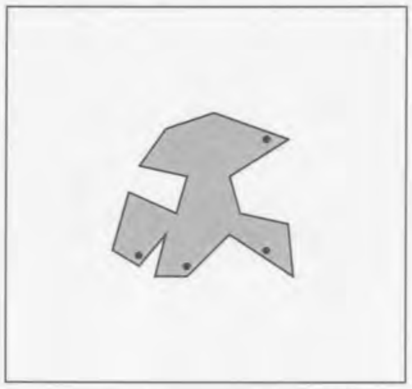

Figure 2.12: A hidden set (Shermer [1992])

- A *hidden guard set* is a hidden set which is also a guard set.

- A *window* of a point x (the *generator*) in a polygon P is an edge of the visibility polygon of x which is not an edge of P and is delimited by two event points b (a reflex vertex called the *base*) and a point q on the boundary of P. Windows constitute borders a guard has to pass when being moved in order to lose (or gain) sight of x since inside P only reflex vertices might block the view.

- A subset of P from which a generator is not visible is called a *pocket*. A pocket is joined to the visibility polygon by a window.

- A *dent edge* is any edge of some polygon where both the vertices are reflex vertices [Wood and Yamamoto, 1993].

- A chain $p_1, \ldots, p_k$ is called *monotone with respect to a line* L if the projections of $p_1, \ldots, p_k$ onto L are ordered the same as in the chain. Two adjacent vertices may project to the same point on L without destroying monotonicity.

- A polygon P is *monotone* if it can be partitioned into two chains that are monotone with respect to the same line.

- A *staircase path* is an orthogonal path such that the path is monotone with respect to the coordinate axes [Wood and Yamamoto, 1993].
- A *staircase polygon* is an orthogonal polygon, a subset of whose vertices constitute a staircase path (see Figure 2.13).

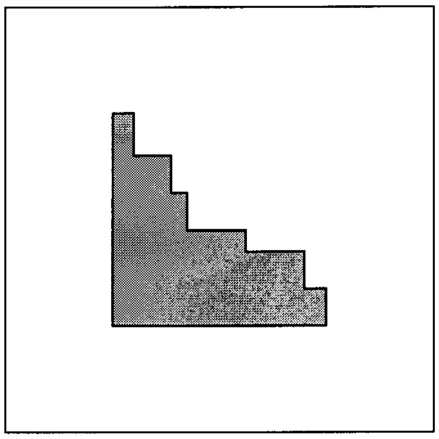

Figure 2.13: A staircase polygon

- The maximum visibility problem asks for locating a point inside the polygon from which the visible area is maximised [Gewali, 1993]. The minimum visibility problem is similarly defined.
- A simple polygon P is said to be an LR-visibility polygon if there exist two points s and t on the boundary of P such that every point of the clockwise boundary of P from s to t (denoted as L) is visible from some point of the counterclockwise boundary of P from s to t (denoted as R) and vice versa [Bhattacharya and Ghosh, 1998].
- A planar graph is a graph which can be drawn or embedded in the plane in such a way that the edges of the embedding intersect only at the vertices of the graph.
- A cut vertex is a vertex whose removal increases the number of components in a graph.
- A biconnected graph is a graph which contains no cut vertices.
- A biconnected planar graph is a planar graph which contains no cut vertices.
- A face in a planar representation of a graph is a planar region bounded by edges and vertices of the representation and containing no edges or vertices in its interior.

This list of terms is not complete. Many of the articles cited in the remainder of the chapter introduce more specific terminology or definitions, in addition to using many of those presented here, to discuss new problems or special cases of existing problems. The terms presented here should, however, be enough to give the reader an understanding of the discussion of the related literature in Section 2.4. Before this discussion of the related literature Section 2.3 presents a brief overview of complexity theory, particularly approaches using in proving problems NP-Complete. This material is presented to help the reader understand related literature presented in Section 2.4. A reader who is familiar with the material is encouraged to skip to Section 2.4

2.3 NP-Complete problems

2.3.1 Introduction

The following sections of the literature review chapter of this thesis discuss a number of problems which have been proved to be NP-Complete or NP-Hard. This section of the chapter is thus aimed at giving the reader an overview of the theory. More detail and more rigorous presentations can be found in Garey and Johnson [1979] and Papadimitriou [1994] and a more accessible discussion appears in Harel [1992]. Section 2.3.2 presents the theory of NP-Completeness and Section 2.3.3 discusses how a new problem can be shown to be NP-Complete. Section 2.3.4 discusses the relationship between NP-Complete and NP-Hard problems.

2.3.2 NP-Complete Problems

The theory of NP-Completeness is designed to be applied to *decision problems* – problems which only have "yes" or "no" answers. Abstractly a decision problem Π consists simply of a set D_Π of *instances* and a subset $Y_\Pi \subseteq D_\Pi$ of *yes-instances*. These decision problems are studied because they have a natural, formal counterpart – "languages" which can be studied in a mathematically precise theory of computation. The correspondence of decision problems and languages is brought about by encoding schemes which can be used to specify problem instances for study. The relationship between recognising languages and solving decision problems is straightforward. A deterministic Turing machine (DTM) program M solves a decision problem Π under an encoding scheme e if M halts for all input strings over its input alphabet, Σ, and $L_M = L[\Pi, e]$. L_M is the language recognised by the program M, that is $L_M = \{x \in \Sigma^* \mid M \text{ accepts } x\}$ and $L[\Pi, e]$ is an encoding of an instance of Π under encoding scheme e. Refer to Garey and Johnson [1979] and Papadimitriou [1994] for more detail on the application of this theory.

The class P is defined as follows:
P $= \{L \mid$ there is a polynomial time DTM program M for which $L = L_M\}$.

Then a decision problem Π belongs to P if there is a polynomial time DTM program that "solves" Π. More informally we could say that $\Pi \in$ P if there is a polynomial time algorithm which solves Π. The class NP can be defined (see Garey and Johnson [1979], page 28 for more detail) in terms of a *nondeterministic algorithm*. Such an algorithm is viewed as being composed of two stages – a *guessing* stage and a *checking* stage. Given a problem instance I, the first stage merely guesses some structure S. The structure is, of course, algorithm-dependent and represents, more or less explicitly, a possible answer to the problem. For example, in the travelling salesperson problem, the structure is a route through the cities to be visited. The checking stage then uses I and S as inputs and proceeds to compute in a normal deterministic manner either eventually halting with an answer "yes", eventually halting with an answer "no" or computing forever without halting. (The last two cases do not always need to be distinguished). A nondeterministic algorithm "solves" a decision problem Π if the following two properties hold for all instances $I \in D_\Pi$:

1. If $I \in Y_\Pi$ then there exists some structure S that when guessed for input I will lead the checking stage to respond "yes" for I and S.

2. If $I \notin Y_\Pi$ then there exists no structure S that when guessed for input I will lead the checking stage to respond "yes" for I and S.

The class NP is then defined informally to be the class of all decision problems Π that, under reasonable encoding schemes, can be solved by polynomial time nondeterministic algorithms.

The relationship between P and NP is fundamental in the theory of NP-Completeness. It is clear that P $\subseteq$ NP – every problem solvable by a polynomial time deterministic algorithm is also solvable by a polynomial time nondeterministic algorithm. Thus if $\Pi \in$ P then $\Pi \in$ NP. The current conjecture is that P $\subset$ NP but this is still an open problem. If P is different from NP then all problems in P can be solved with polynomial time algorithms and the problems in NP$-$P are termed intractable. The theory of NP-Completeness focuses on proving results of a weaker form – if P$\neq$NP then there are problems in NP which are neither solvable in polynomial time nor NP-Complete. The class P can be viewed as consisting of the "easiest" problems in NP and the class of NP-Complete problems contains the hardest problems in NP.

A language L is defined to be NP-Complete if $L \in$ NP and for all other languages $L' \in$ NP, $L' \propto L$ (where $L_1 \propto L_2$ implies a polynomial transformation from a language L_1 to a language L_2). Following from this, a decision problem Π is said to be NP-Complete if $\Pi \in$ NP and for all other decision problems $\Pi' \in$ NP, $\Pi' \propto \Pi$. This implies "the common fate phenomenon" of NP-Complete problems – every NP-Complete problem is polynomially transformable to every other one. If a single NP-Complete problem can be solved in polynomial time, then all problems in NP can be so solved. If any problem in NP is intractable, then so are all NP-Complete problems. At this stage it has not been proven that any NP-Complete problem is inherently intractable. In addition, no one has yet found a polynomial time solution for any NP-Complete problem. Even without a proof that NP-Complete problems are intractable, we know that any new problem which can be proved to be NP-Complete is at least as hard as the other NP-Complete problems and that a major breakthrough will be needed to solve such a problem with a polynomial time algorithm.

2.3.3 Proving the NP-Completeness of a new problem

From the above it seems that in order to prove a new problem Π NP-Complete, one must show that *every* problem in NP transforms to the new problem. *A priori*, it is not even clear that any NP-Complete problem need exist. It turns out that if *at least* one NP-Complete problem is known to exist that it is only necessary to show that

1. $\Pi \in$ NP
2. some known NP-Complete problem Π' transforms polynomially to Π.

This result arises because if L_1 belongs to NP and L_1 is NP-Complete then every other $L' \in$ NP transforms to L_1. Now if L_2 is also in NP and there exists a polynomial time transformation from L_1 to L_2 then there exists a transformation from every other $L' \in$ NP to L_2.

Harel [1992] presents the argument in a slightly different form. He states that to prove that Π is NP-Complete it is not necessary to find polynomial transformations from all of the other NP-Complete problems. It is only necessary to transform Π to some problem Π'' which is known to be NP-Complete ($\Pi \propto \Pi''$) and to transform another (or the same) problem Π' already known to be NP-Complete to Π ($\Pi' \propto \Pi$). The first transformation shows that in terms of tractability Π cannot be worse than Π'', that is that Π is in fact in NP (as required above). Then the second transformation shows that in terms of tractability Π cannot be better than Π'. Since Π' and Π'' are both NP-Complete and stand or fall together then Π must be NP-Complete too. Thus if one problem is known to be NP-Complete then other problems can be proved to be NP-Complete too by using the transformation approach twice for each new problem. Harel [1992] notes that in practice only the second of these transformations is carried out in an NP-Completeness proof. To show that Π cannot be any worse than the NP-Complete problems, that is that it is in fact in NP, can often be done more easily directly – by showing that Π can be solved by a polynomial time nondeterministic algorithm.

The “first” NP-Complete problem is the *satisfiability* problem and Cook’s Theorem is used to prove that this problem is in fact NP-Complete. Now that a single problem has been shown to be NP-Complete the process of devising an NP-Completeness proof for any decision problem Π consists of the following four steps

1. showing that Π is in NP,
2. selecting a known NP-Complete problem Π'
3. constructing a transformation f from Π' to Π, and
4. proving that f is a polynomial transformation.

This approach is taken in the NP-Completeness proofs in this thesis.

Determining whether the decision problem, Π, is in NP is a matter of showing that given any solution for an instance I it is possible to verify in polynomial time whether or not that solution “proves” that the answer for I is “yes”. For example, a nondeterministic algorithm for ***travelling salesperson*** could be constructed by using a guessing stage that simply guesses an arbitrary sequence of cities and a checking stage that checks whether the guessed solution would result in a tour of the desired length. The existence of such a polynomial time nondeterministic algorithm shows that Π is in NP.

Once the new problem has been shown to be in NP, then a known NP-Complete problem must be selected and a transformation must be constructed from the known problem to the new problem. There are three general transformation approaches that are used in NP-Completeness proofs and that can provide some ideas about how to tackle a specific NP-Completeness proof. Note that these approaches cannot be applied to all NP-Completeness proofs. They just give some ideas about how one might approach the task of proving a new problem to be NP-Complete.

These approaches are called [Garey and Johnson, 1979]

1. restriction
2. local replacement
3. component design

Restriction This is the easiest and perhaps most frequently applied of the three types of NP-Completeness proofs. An NP-Completeness proof by restriction for a given problem $\Pi \in$ NP consists simply of showing that Π contains a known NP-Complete problem Π' as a special case. The heart of such a proof lies in the specification of the additional restrictions to be placed on the instances of Π so that the resulting restricted problem will be identical to Π'. There should be an obvious one-to-one correspondence between their instances that preserves “yes” and “no” answers. This one-to-one correspondence, which provides the required transformation from Π' to Π (see point (3)) above is usually so apparent that it need not even be given explicitly. The approach taken in this kind of proof is often to focus on the target problem itself and attempt to restrict away its "inessential" aspects until a known problem appears.

Local replacement In this type of proof the transformations are sufficiently non-trivial to warrant spelling out in the standard proof format but they are still relatively uncomplicated. Proofs of this type rely on picking some aspect of the known NP-Complete problem instance to make up a collection of basic units and then obtaining an instance of the target problem by replacing each basic unit in a uniform way with a different structure.

Component design Proofs of this type tend to be the most complicated. The basic idea is to use the constituents of the target problem instance to design certain "components" (also called gadgets and widgets) that can be combined to realise an instance of the known NP-Complete problem.

Garey and Johnson [1979] discuss these transformations in more detail and give examples of the application of each approach.

2.3.4 NP-Hard problems

The techniques for proving NP-Completeness can also be used for proving that problems outside of NP are hard. Any decision problem Π, whether a member of NP or not, to which we can transform an NP-Complete problem will have the property that it cannot be solved in polynomial time unless P=NP. Such a problem could be said to be "NP-Hard" since it is at least as hard as the NP-Complete problems. The idea of NP-Hardness can be generalised in such a way that not only decision problems can be proved to be "at least as hard" as the NP-Complete problems.

In this thesis only the notion of decision problems will be used. The reader is referred to Garey and Johnson [1979] and Papadimitriou [1994] for more detail on NP-Hardness.

2.3.5 Summary

In summary, the approach taken to prove a new problem, say R, to be NP-Complete is to

1. show R is in NP – this involves showing that R can be solved by a nondeterministic polynomial algorithm,
2. selecting a known NP-Complete problem, say T,
3. constructing a transformation f from T to R, and
4. proving that f is a polynomial transformation.

To show that a problem is NP-Hard only steps 2 to 4 are required.

Many of the problems raised in the areas of guarding, covering, partitioning and visibility have been proved to be "hard" in the sense discussed above so the theory presented in this section is useful in understanding many of the papers presented in the literature and in the section below (Sections 2.4. This theory is also a useful starting point for some of the new results proved in the body of this thesis – Chapters 4 to 7.

2.4 Relation of previous work to ALP

2.4.1 Overview

In Section 2.1 above the commonalities between ***ALP*** and visibility, guarding and polygon decomposition were briefly introduced. These problems, which all have their roots in real world problems, can in some sense be thought of as different castings of the same problem and results presented in one area often apply elsewhere as well. Problems of this type have been studied extensively over the last 30 years and the results of these studies have been collected in a number of very comprehensive surveys. The survey article by Wood [1985] discusses (amongst other problems) research in visibility in orthogonal (he calls them isothetic) polygons. The monograph by O'Rourke [1987] "Art Gallery Theorems and Algorithms" provides an excellent overview of the current state of knowledge of Art Gallery Guarding problems at the time it was written. The survey paper by Shermer [1992] "Recent Results in Art Galleries" extended O'Rourke's work. O'Rourke [1993] also discusses many of these problems in his textbook. The survey papers by Urrutia [1999] "Art Gallery and Illumination Problems", Asano *et al.* [1999] "Visibility in the plane" , and Keil [1999] "Polygon Decomposition" are all published in the Handbook of Computational Geometry and give a very thorough coverage of the material. In addition the summaries of results by Suri [1997] "Polygons" and O'Rourke [1997] "Visibility" published in the "Handbook of Discrete and Computational Geometry" are excellent sources of reference for results in these areas. There is thus little point in trying to repeat these surveys and the reader is encouraged to consult them to get an overall feeling of the current state of research in these areas.

The next section gives a very brief overview of some of the important general results in the areas of guarding, visibility and polygon decomposition. A reader who is very familiar with these areas could skip this section of the thesis. The following section puts ***ALP*** into context with the previous and related research work. The subsequent section focusses in greater depth on results, problems or techniques which give special insight into ways of dealing with ***ALP***.

2.4.2 Short historical perspective

2.4.2.1 Guarding problems

The original art gallery guarding problem, posed by Klee in 1973, asks for the number of guards required to survey an art gallery with n walls and no interior obstructions [O'Rourke, 1987]. The problem was first solved by Chvatal [1975] who showed that there exist polygons where $\lfloor n/3 \rfloor$ vertex guards are both necessary and sufficient to guard the entire area of the polygon. He first showed using comb polygons that any guard set for such a polygon with n vertices, $g(n)$, must have at least $\lfloor n/3 \rfloor$ guards and then by an inductive argument on triangulation graphs of polygons showed that the size of the guard set is less than or equal to $\lfloor n/3 \rfloor$. Thus $g(n) = \lfloor n/3 \rfloor$. A different and very simple sufficiency proof, using three-colouring on triangulation graphs, of the same result was given in 1978 by Fisk [1978]. O'Rourke [1987] presents very lucid discussion on both of these proofs. Avis and Toussaint [1981] used Fisk's proof to implement an $O(n \log n)$ algorithm to assign positions to the guards. Algorithms such as this for finding guard sets are often called "guard placement algorithms". Most guard placement algorithms work by imitating upper-bound art gallery proofs. The algorithm by Avis and Toussaint [1981] is in fact an algorithmic imitation of Fisk's proof.

Later Lee and Lin [1986] tackled the issue of the computational complexity of the art gallery guarding problem. Their work was aimed at the problem of finding the minimum number of guards to cover a given polygon. They proved that the problem for a simply connected simple polygon is NP-Hard for the *minimum vertex guard* problem (where guards must be located at the vertices of the polygon), *minimum point guard* problem (where guards can be placed anywhere in the interior of the polygon or on its boundary) and *minimum edge guard* problem (where guards can only be placed on the edges which make up the polygon boundary and are allowed to move along the edge on which they are placed). Their proof is based on a transformation from Boolean Three Satisfiability (3SAT). (Note: Shermer [1992] credits Aggarwal with first proving the result for the minimum point guard problem but the author of this thesis was unable to obtain a copy of Aggarwal's Ph.D. thesis.)

Lee and Lin's results imply that it is in general impractical to find a minimum set of guards for a given polygon. Some polygons can be guarded by a single guard and there exist polygons where $\lfloor n/3 \rfloor$ are required. Finding the exact number of guards required for a given polygon cannot always be done in reasonable time. Algorithms such as that by Avis and Toussaint [1981] do not guarantee a minimum solution for a given polygon but will place a guard set which is always sufficient (and sometimes necessary) to guard the polygon.

If the art gallery is allowed to have obstructions (pillars etc.) in its interior, the corresponding floor plan is a simple polygon with other simple disjoint polygons, called holes, inside it. In such a polygon, $\lfloor n/3 \rfloor$ guards are no longer sufficient.

O'Rourke showed that $\lfloor (n+2h)/3 \rfloor$ point or vertex guards are always sufficient to guard any polygon with n vertices and h holes. Shermer showed that $\lfloor (n+h)/3 \rfloor$ guards are necessary for some polygons with n vertices and h holes and conjectured that this number of guards was also sufficient for any polygon with h holes. In 1984 Shermer showed that $\lfloor (n+1)/3 \rfloor$ guards are always sufficient and sometimes necessary for any polygon with one hole. All of these results appear in O'Rourke [1987].

In 1991 two independent and dramatically different proofs were generated for the fact that $\lfloor (n+h)/3 \rfloor$ point guards are also sufficient ([Bjorling-Sachs and Souvaine, 1991; Hoffmann *et al.*, 1991]). Neither paper gave details of algorithms for placing the guards but the algorithm derived from the Hoffmann *et al.* [1991] proof has complexity $O(n^3 \log n)$. In 1995 Bjorling-Sachs and Souvaine [1995] presented an $O(n^2)$ algorithm to place the guards. This algorithm is based on a constructive proof. The proof and the algorithm work by first connecting each hole in the polygon to the exterior of the polygon and then triangulating the new hole-free version of the polygon. The channels are constructed in such a way that only one new vertex is added for each channel. In addition, there is always a triangle in the new hole free polygon that "sees" all of the channel – any vertex of this triangle sees the whole channel. These special triangles are included so that a guard placement based on three-colouring in the hole free polygon will automatically cover the channels and so the original polygon would have been guarded.

Other variations of the classic problem arise when specified subsets of the polygon, rather than just points, are allowed as elements of guard sets. The first of these variations is the *edge guard problem*. An edge guard is a guard who is allowed to patrol individual edges of a polygon rather than being restricted to one point. The edge guard problem then asks for the minimum number of edge guards necessary to cover any polygon of n vertices. Toussaint (as cited in Shermer [1992]) conjectured that (if a small number of polygons are excluded) $g^E(n) = \lfloor n/4 \rfloor$ guards are necessary. Figure 2.14 shows the type of polygons where $\lfloor n/4 \rfloor$ edge guards are required. Two types of polygons are known which require $\lfloor (n+1)/4 \rfloor$ guards (see Figure 2.15) but these are thought to be exceptions. O'Rourke [1983] made some progress on Toussaint's conjecture. He was unable to establish an upper bound on the number of edge guards required but was able to place a bound on the number of *mobile guards* necessary. A mobile guard is a guard that can patrol an edge or diagonal of a polygon. Every edge guard is thus a mobile guard and the upper bound on the number of mobile guards thus gives the least upper bound on the number of edge guards $g^M(n) \leq g^E(n)$. O'Rourke [1983] showed that $g^M(n) = \lfloor n/4 \rfloor$ mobile guards are necessary. Shermer [1992] investigated *diagonal guards* and showed that $\lfloor (2n+2)/7 \rfloor \leq g^D(n) \leq \lfloor (n-1)/3 \rfloor$. He also showed that, aside from a small number of exceptions, $\lfloor n/4 \rfloor \leq g^E(n) \leq \lfloor 3n/10 \rfloor$. Sack and Suri [1990] have given an O(n) algorithm to detect if a given polygon can be guarded by one edge guard and Ke [1989] (as cited in Shermer [1992] – the original reference could not be obtained) gave an $O(n \log n)$ algorithm for the related problem of detecting if a given polygon can be guarded by one line-segment guard.

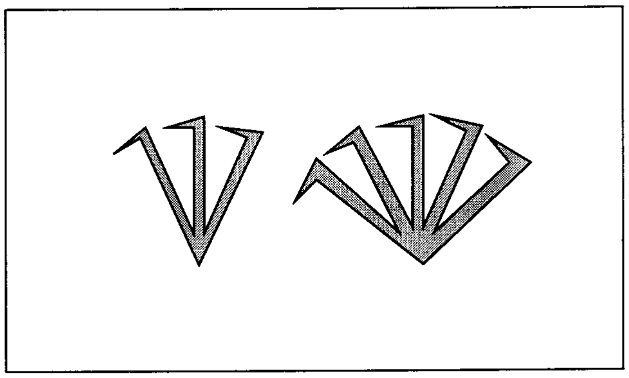

Figure 2.14: Polygons requiring $\lfloor n/4 \rfloor$ edge guards (Shermer [1992])

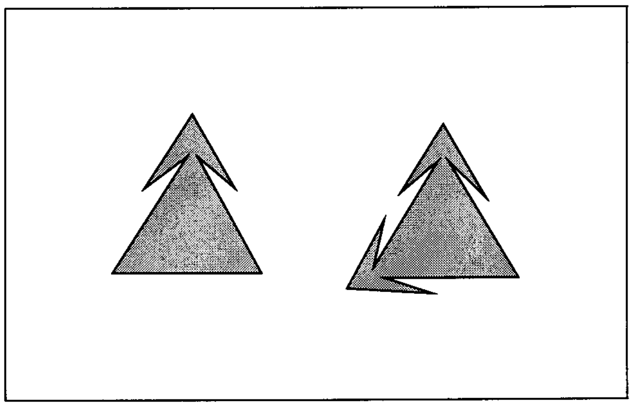

Figure 2.15: The two polygons requiring $\lfloor (n+1)/4 \rfloor$ edge guards (Shermer [1992])

Other variations of the general problem are the orthogonal art gallery theorem [Kahn *et al.*, 1983] (which is discussed in more detail below); guarding rectangular art galleries [Czyzowicz *et al.*, 1994]; generalised guarding of rectilinear polygons [Györi *et al.*, 1996]; the prison guard problem [Kooshesh *et al.*, 1990; Füredi and Kleitman, 1994]; the treasury guard problem [Deene and Joshi, 1992; Carlsson and Jonsson, 1993]; edge guards [Viswanathan, 1993]; edge guards in star polygons [Subramaniyam and Diwan, 1991]; diagonal and chord guards [Lu *et al.*, 1998]; guard edges [Park *et al.*, 1993]; optimally placing k guards in a polygon to maximise the area or portion of boundary visible [Ntafos and Tsoukalas, 1994]; floodlight guards [Bose *et al.*, 1993; Czyzowicz *et al.*, 1993; Contreras *et al.*, 1998b,a]; the searchlight scheduling problem [Sugihara *et al.*, 1990]; hidden guard sets [Shermer, 1989]; watchman paths [Carlsson and Jonsson, 1995]; watchmen in grids [Ntafos, 1986]; periscope guards in grids [Gewali and Ntafos, 1993] and other problems [Liaw *et al.*, 1993; Venkatasubramanian and Cullum, 1993]. Methods for developing approximations [Avis and Toussaint, 1981; Ntafos and Tsoukalas, 1994] or for bounding approximations have also been studied [Eidenbenz *et al.*, 1998].

2.4.2.2 Visibility Problems

Visibility problems have been studied from at least the early part of the twentieth century and a number of different topics have been studied [Asano *et al.*, 1999]. This section gives a brief overview of some of the main topics. More detail can be found in O'Rourke [1987], Asano *et al.* [1999] and the original papers.

The basic question in visibility can be stated as

Given a polygon P and two points $x, y \in P$, are x and y visible?

In this case x and y are visible if the line segment xy contains no points of the exterior of P [Asano *et al.*, 1999]. In a polygon with holes, the holes are taken to be part of the exterior of the polygon and so the same idea of visibility applies.

This idea leads naturally to the problem of determining what part of some polygon P can be seen from a given point a. The *visibility polygon* $V(a)$ of a point a in a polygon P is the set of all points visible to a ($V(a) = \{q \in P | a \text{ sees } q\}$). Any visibility polygon $V(a)$ is star shaped – a is its kernel. The problem of determining the visibility polygon for some point a in a polygon has been well studied (see for example ElGindy and Avis [1981]; Heffernan and Mitchell [1995]).

Avis and Toussaint [1981] extended this work by defining the concepts of *complete visibility*, *strong visibility* and *weak visibility* from some fixed edge uv of P.

1. P is said to be completely visible from an edge uv if for every $z \in P$ and every $w \in uv$, w and z are visible.

2. P is said to be strongly visible from an edge uv if there exists a $w \in uv$, such that for every $z \in P$, w and z are visible.

3. P is said to be weakly visible from an edge uv if for each $z \in P$ there exists a $w \in uv$ such that w and z are visible.

Avis and Toussaint [1981] then presented an $O(n)$ algorithm to determine whether a given polygon P is completely visible, strongly visible or weakly visible from a particular edge in P. Some related work on weak visibility of polygons is due to Sack and Suri [1990], Ghosh *et al.* [1993] and Doh and Chwa [1993].

An area which is related to the above is edge-to-edge visibility in polygons. Here again there are degrees of visibility between the edges concerned [Avis *et al.*, 1986].

1. Edge uv is said to be completely visible from edge xy if for all points on z on edge xy and all points w on edge uv, w and z are visible.

2. Edge uv is said to be strongly visible from edge xy if there exists a point z on edge xy such that for all points w on edge uv, w and z are visible.

3. Edge uv is said to be weakly visible from edge xy if for each point w on edge uv there exists a point z on edge xy such that w and z are visible.

4. Edge uv is said to be partially visible from edge xy if there exists a point w on edge uv and a point z on edge xy such that w and z are visible.

Avis *et al.* [1986] presents a linear algorithm to compute these four edge-to-edge visibilities. This algorithm is discussed is more detail in Section 2.4.4.3.1 because it has some direct relevance to ***ALP***.

The problem of computing the first intersection of the boundary of some polygon P with a light ray from some edge or point in P is also of interest in the field of visibility [Chazelle and Guibas, 1989].

Link visibility is another area that has been the subject of research [Asano *et al.*, 1999]. The *link* distance between two points p and q in a polygon P is the minimum number of line segments (links) in a polygonal path from p to q that stays in P. Two points are called *link-j* visible if the link distance between them is at most j. Letting $j = 1$ gives the standard visibility. Note that this form of visibility is also referred to as L_j visibility.

Another important area of research is in computing visibility graphs of polygons [Asano *et al.*, 1999]. A visibility graph of a polygon is a graph whose vertices are the vertices of the polygon and whose edges are the pairs of visible vertices (see Figure 2.16 for an example). Related to the computing of visibility graphs is the problem of determining for a given graph G if there exist some polygon P that has G as its visibility graph. This is called the visibility graph *recognition* problem. The problem of actually constructing such a P is called the visibility graph *reconstruction* problem. In this area see for example Ghosh [1991], Andreae [1992], Srinivasaraghavan and Mukhopadhyay [1993], Ghosh [1996] and Ghosh [1997].

Other types of visibility have also been studied (see Asano *et al.* [1999] for more information) – minimum and maximum visibility [Gewali, 1993]; clear visibility; dent and staircase visibility [Wood and Yamamoto, 1993]; $\mathcal{O}$-visibility; rectangular visibility; circular visibility; visibility with reflection; X-ray visibility; LR-visibility [Das *et al.*, 1993; Bhattacharya and Ghosh, 1998]; etc.

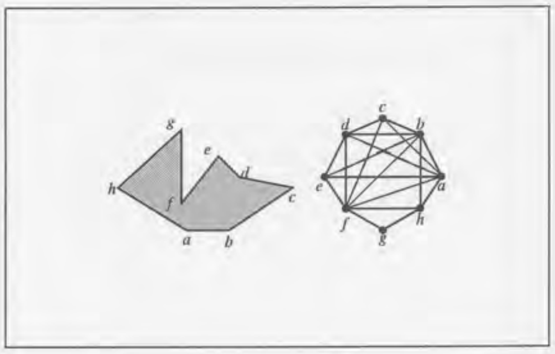

Figure 2.16: A polygon and its visibility graph

2.4.2.3 Polygon Decomposition

2.4.2.3.1 Overview As discussed in Section 2.1 polygon decomposition is categorised in two different ways – covering and partitioning. The basic covering problem is to find the minimum number of polygons, with some predefined characteristics, to cover the complete area of an enclosing polygon. This problem is intimately related to the guarding problem – guarding a polygon with guards chosen from some set S is the same as covering the polygon with visibility polygons of the members of S. The partitioning problem is the same as the covering problem except that the polygons into which the enclosing polygon is decomposed are not permitted to overlap. In addition to the survey articles mentioned earlier Keil and Sack [1985] presents a very readable but now somewhat dated overview of polygon decomposition. The partitioning problem is the version of the decomposition problem which has relevance to this research. Finding the convex map of an urban area is equivalent to partitioning a polygon with holes into the smallest number of convex polygons. This section discusses some of the previous research in both covering and partitioning.

2.4.2.3.2 Covering The problem of covering a polygon with simpler polygons has been the subject of much research and many of the problems in this class have been shown to be NP-Hard. Many NP-Hardness results for covering problems were first established for polygons with holes [O'Rourke and Supowit, 1983]. Using a transformation from 3SAT they proved that minimum convex cover for a multiply connected polygon (a polygon with holes); that minimum star-shaped cover for a multiply connected polygon; and minimum spiral cover for a multiply connected polygon are NP-Hard. Lee and Lin [1986] were the first to establish some NP-Hard results for polygons without holes – in particular they showed that the star cover problem (covering a simple simply connected polygon with the smallest number of star polygons) was NP-Hard. This result follows from the fact that they proved that the minimum point guard problem was NP-Hard – star covering and point guard are different castings of the same problem. They also showed that vertex guarding and edge guarding are NP-Hard and thus that the corresponding cover problems are NP-Hard as well. Later Culberson and Reckhow [1988, 1994] established the NP-Hardness for convex cover and similar problems using a transformation from SAT.

One approach to dealing with the difficulty of covering problems is to consider restrictions of the general problem. For instance, the problem of covering orthogonal polygons with many types of subpolygons has received a lot of attention. Some examples are given below but the reader is referred to the survey articles mentioned above for more complete coverage. Masek has shown that finding the minimum cover of a orthogonal (or rectilinear) polygon with rectangles is NP-Hard (the problem at that stage was termed *rectilinear picture compression*). This result at the time was only known to apply to orthogonal polygons with holes. Franzblau and Kleitman [1984] then showed that the problem can be solved in polynomial time if the orthogonal polygon is vertically convex. Franzblau [1989] presents an approximation algorithm for this problem which gives a solution which is at worst twice the minimum solution if the polygon has no holes. Lubiw [1990] has also worked on this problem – showing it is a special case of the boolean basis problem.

In orthogonal polygons with holes, Conn and O'Rourke [1987] (as cited in Shermer [1992]) have shown that covering either the boundary or the reflex vertices is NP-Complete but they found an $O(n^{2.5})$ time algorithm for covering the convex vertices. Culberson and Reckhow [1988] have shown that covering an arbitrary orthogonal polygon with rectangles is NP-Complete even if only the boundary of the orthogonal polygon is to be covered. They have also done similar work with Dent diagrams in orthogonal polygons [Culberson and Reckhow, 1989b,a]. Motwani *et al.* [1990b] showed that a polynomial time algorithm can be found for orthogonally convex polygon coverings of orthogonal polygons with three dent orientations but not for four dent orientations. Motwani *et al.* [1990a] showed that covering orthogonal polygons with star polygons can be accomplished in polynomial time and Gewali *et al.* [1992] showed that a minimum orthogonal star polygon cover for a horizontally convex orthogonal polygon can also be found in polynomial time.

2.4.2.3.3 Partitioning In the previous section (Section 2.4.2.3.2) many of the polygon covering problems are shown to be NP-Hard. The partitioning problem, however, has been shown to be solvable in polynomial time in many cases.

The early work in partitioning polygons without holes into the minimum number of convex polygons [Feng and Pavlidis, 1975; Schachter, 1978] produced algorithms which could not guarantee a minimum of components. Then Chazelle and Dobkin [1985] were able to show that finding a minimum convex partition of a simple simply connected polygon (a cover by *nonoverlapping* convex pieces) has polynomial-time complexity and presented a polynomial-time algorithm for this problem. Their algorithm is $O(n+c^3)$ where n is the number of vertices in the polygon and c is the number of reflex angles. The algorithm allows for the introduction of Steiner points. It begins by producing a naive decomposition of the polygon by removing each notch (reflex angle) in turn by means of a simple line segment drawn from the notch until it encounters a line already in the decomposition. This naive decomposition is then "improved" until an optimal convex decomposition is achieved. The process for doing the improvement is based on the idea of X-patterns (and Y-patterns) which describe particular interactions of notches. The reader is referred to the original paper for details. Keil [1985] considered partitioning which does not allow Steiner points and presented polynomial-time dynamic programming algorithms for partitioning a simple simply connected polygon into the minimum number of convex polygons, spiral polygons, star polygons and monotone polygons. Recently Keil and Snoeyink [1998] presented an improved algorithm for the same problem.

The convex polygon partitioning problem is NP-Hard in polygons with holes – simple multiply connected polygons [Lingas, 1982]. The proof by Lingas [1982] is based on a transformation from a planar version of 3SAT. Lingas's 1982 paper is a revision of an earlier paper and in the revision he uses some insights from O'Rourke and Supowit [1983] about the truth setting components of the transformation to simplify his paper. Lingas [1982] credits O'Rourke and Supowit [1983] with having proved the same result independently of him.

Partitioning problems where the original polygon is to be partitioned into different types of subpolygons (spiral polygons, star-shaped polygons, etc.) have also been studied (see [Keil, 1999] for an overview of these).

As is the case with guarding, visibility and covering, work has been done on considering approximations to the solution for the general problems and in studying restrictions on the general problems. Some examples of this work are given below. (More detail can be found in the survey articles mentioned earlier).

In studying restrictions of the general problem to partition a orthogonal polygon with the minimum number of rectangles, Imai and Asano [1986] present an $O(n^{1.5}\log n)$ algorithm if the polygon has holes; Liou *et al.* [1989] (as cited in Shermer [1992]) present an $O(n)$ algorithm for this problem if there are no holes; and Soltan and Gorpinevich [1993] showed that this problem can be solved in polynomial time even if the original polygon contains degenerate (point) holes. In addition, Ku and Leong [1995] have studied optimal partitions of rectilinear layouts used in VLSI design.

Other types of restrictions have also been studied. Asano *et al.* [1986] discuss the problem of partitioning a polygon into a minimum number of trapezoids with two horizontal sides. The problem is shown to be NP-Complete for polygons with holes but solvable in polynomial time for polygons without holes. Lingas and Soltan

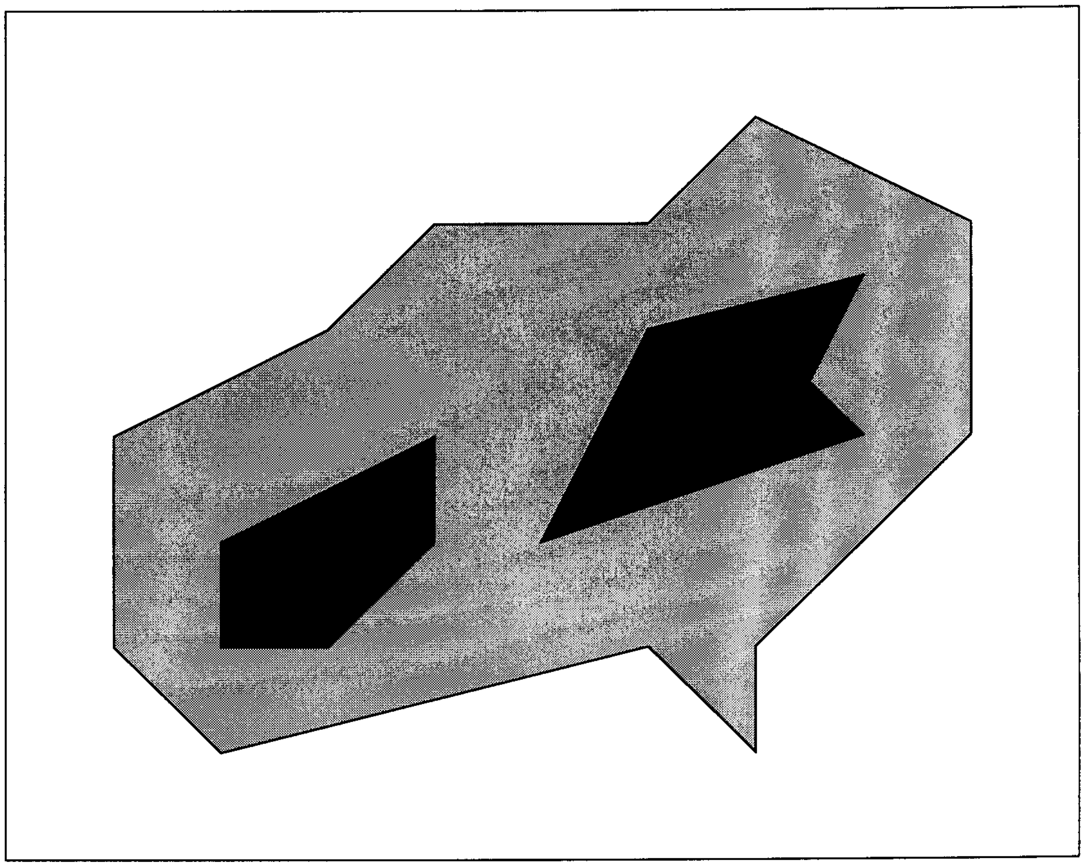

Figure 2.17: A multiply connected simple polygon (a simple polygon with holes)

[1996] prove that the problem of partitioning a polygon with holes into a minimum number of convex polygons by cuts in a family of directions F is NP-Hard if $|F| \geq 3$ and polynomial time for $|F| \leq 2$. Keil [1999] gives additional examples.

2.4.3 Putting *ALP* into context with other research

In Section 2.1 above the fact that there are some commonalities and some differences between ***ALP*** and visibility, guarding and polygon decomposition problems is briefly discussed. The aim of this section is to highlight these commonalities and differences in the context of the previous and related research (see Section 2.4.2).

Consider the polygon with holes shown in Figure 2.17. This polygon is the basis for much of the discussion that follows in this section. This polygon is chosen to be representative of a class of polygons which research in visibility, guarding and decomposition focusses on – a multiply connected simple polygon or a simple polygon with holes. It is a relatively uncomplicated example but is still complicated enough to illustrate the complexity of these problems. It can also be considered as an example of the types of polygons which are used in space syntax and so is of

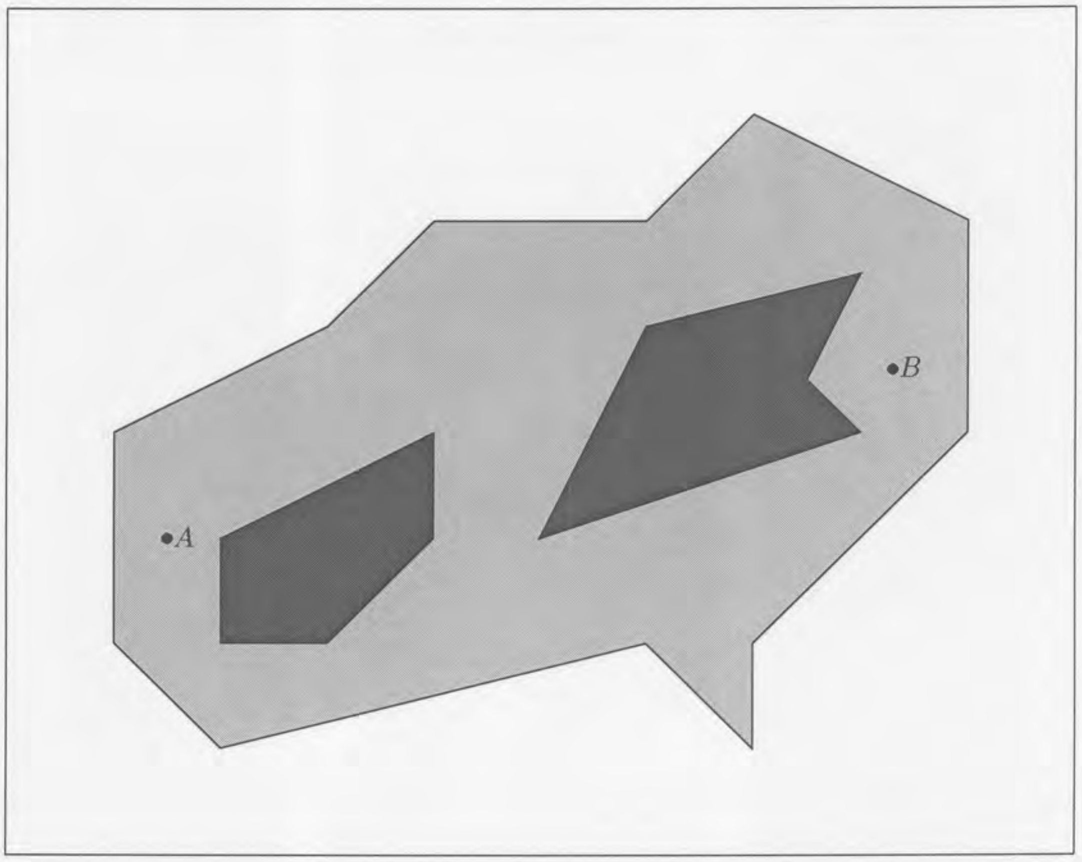

Figure 2.18: Point-point visibility

specific interest in ***ALP***.

As mentioned earlier, the intent of the convex map of some urban layout is to give a localising perspective of the urban layout or town plan. It tells one what can be seen (and directly accessed) from a particular point in the town. The intent of the axial map of some urban layout is that it offers a globalising perspective that takes into account how far one can *see* (or walk) in the town. These two concepts relate to the idea of *visibility* in a polygon. The basic question in visibility research is "can some point in the polygon see some other point in the polygon?". This idea of "point-point" visibility is shown in Figure 2.18. Here the question is "Can point A see point B?" or "Is there a direct line of sight between A and B?". Clearly here the answer is "No". This can be related to space syntax in the sense that A and B are not in the same "locality". Some effort must be put in to get from A to B or if one starts at A to get to a point where one is able to see B. The axial map of the layout will address how much effort is required. This point is addressed later in the discussion. Note however, at this point, that A and B are L_2 visible or *link*-2 visible. They can be joined by a path made up of two line segments. This point is also returned to later.

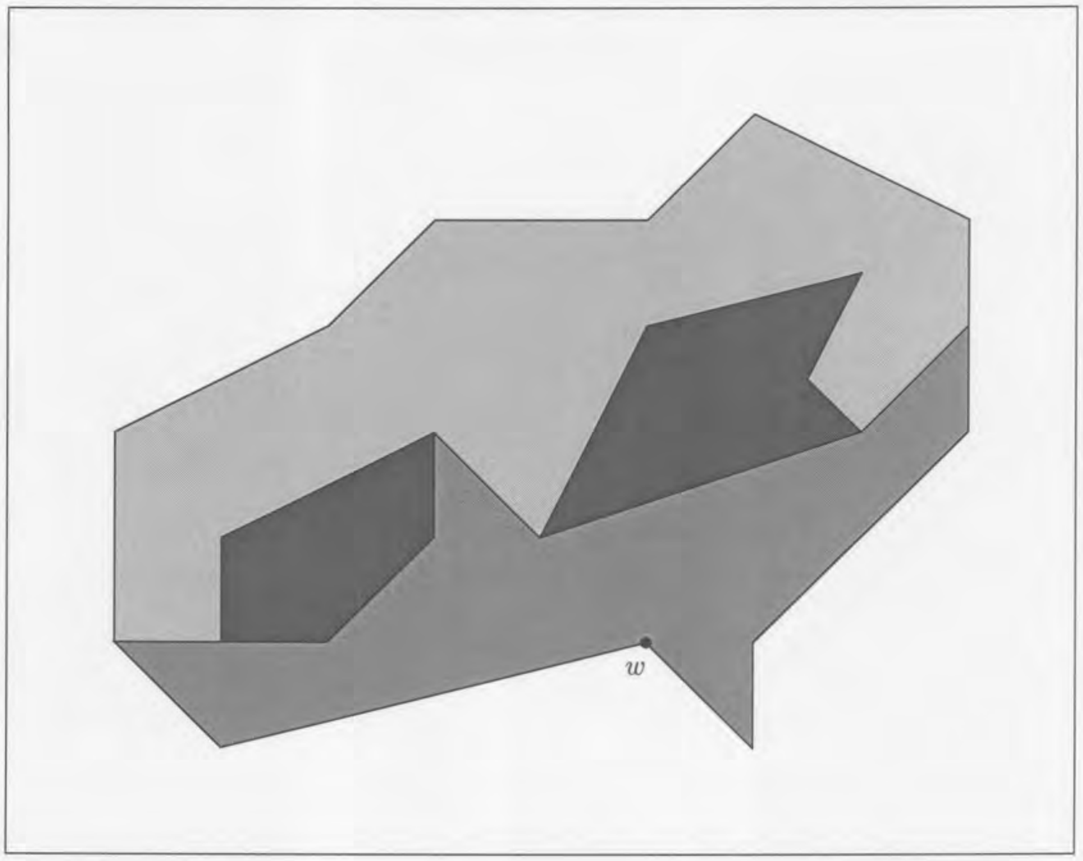

Figure 2.19: Placing a vertex guard

The traditional art gallery guarding problem is to determine the minimum number of guards necessary to cover the interior of an n-wall art gallery [O'Rourke, 1987]. In the example polygon given in Figure 2.17 the aim of guarding would be place the minimum number of guards so that all of the interior of the polygon can be seen by at least one of the guards. From this it is easily seen that the problem of guarding a polygon is very closely related to visibility in polygons. In fact, all guarding problems are essentially visibility problems. In this discussion vertex guards (that is any guard must be placed at a vertex on the exterior of the polygon or at a vertex on the boundary of one of the holes) are used to describe the guarding problem. In Figure 2.19 a single vertex guard, w, has been placed. The darker shaded portion of the interior of the original polygon is the area which can be seen or guarded by this guard. Note that this more darkly shaded area of the interior of the original is also the *visibility polygon* from the point w. Clearly the guard w cannot see all of the interior of the polygon and additional guards are required.

A second guard can be placed at x on the boundary of the original polygon – see Figure 2.20. This guard can see a different subset of the original polygon and thus generates a different visibility polygon. Note that there are some areas of overlap

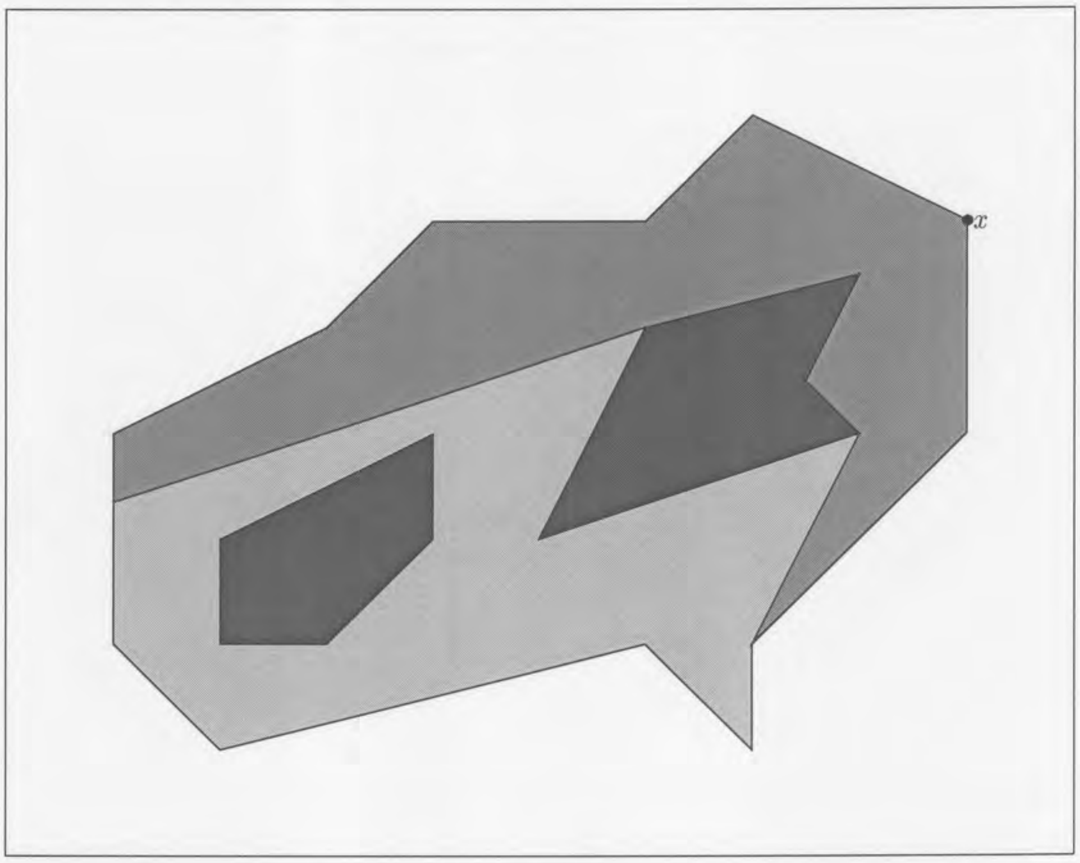

Figure 2.20: Placing the next vertex guard

between the two visibility polygons. These are areas which are guarded by both w and x.

A minimal vertex guard set for the original polygon is shown in Figure 2.21. There could be other guard placements but for this example 4 guards are required. If the problem was that of placing point guards, edge guards, diagonal guards, etc. then similar results would apply.

Note that each guard which is placed generates its own visibility polygon. Each of these visibility polygons is a star polygon – the guard position is the one point in the polygon which is guaranteed to be able to see and be seen by every other point in this polygon. Thus the guard placement gives a star polygon *cover* for this polygon.

As with this specific example, many guarding problems have some commonality with the area of *polygon decomposition* – covering a polygon with the minimum number of polygons of some specified type is equivalent to the placement of the minimum number of guards of some specified type so that each point inside the polygon is visible to some guard. In polygon decomposition the focus is in decomposing given polygons into smaller more easily manageable parts. Polygon de-

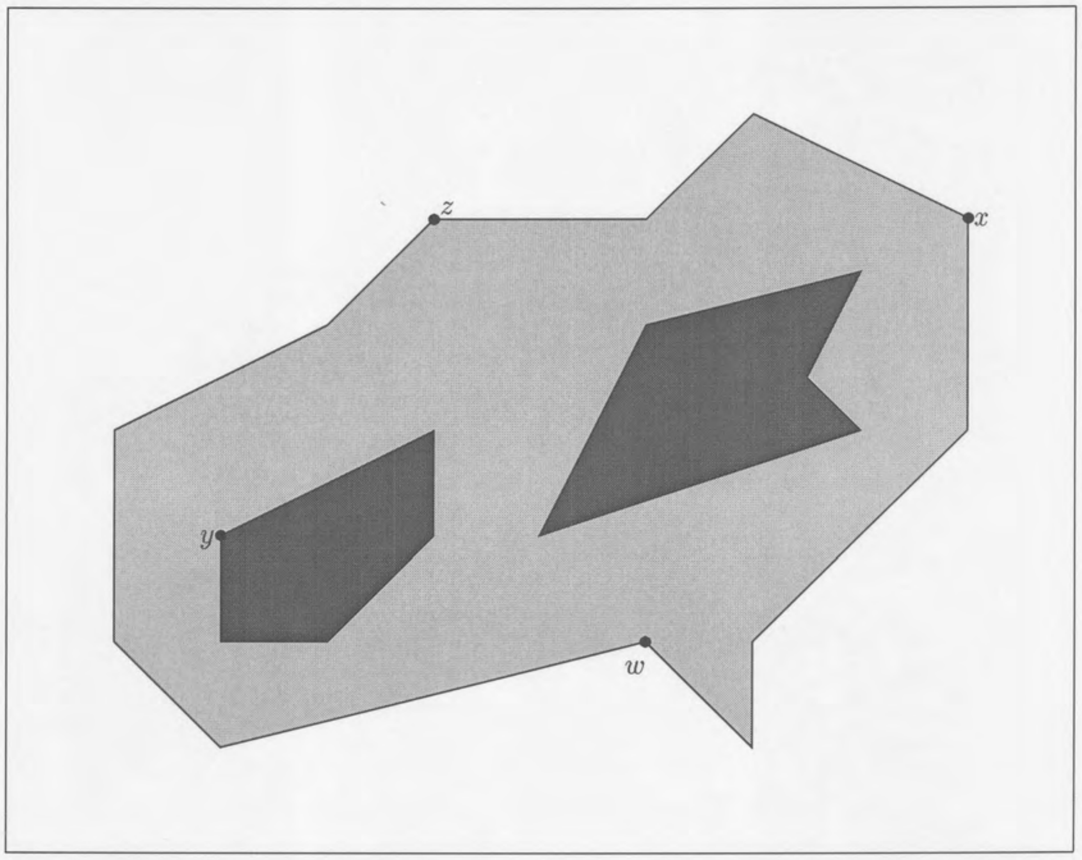

Figure 2.21: A vertex guard set for the example polygon

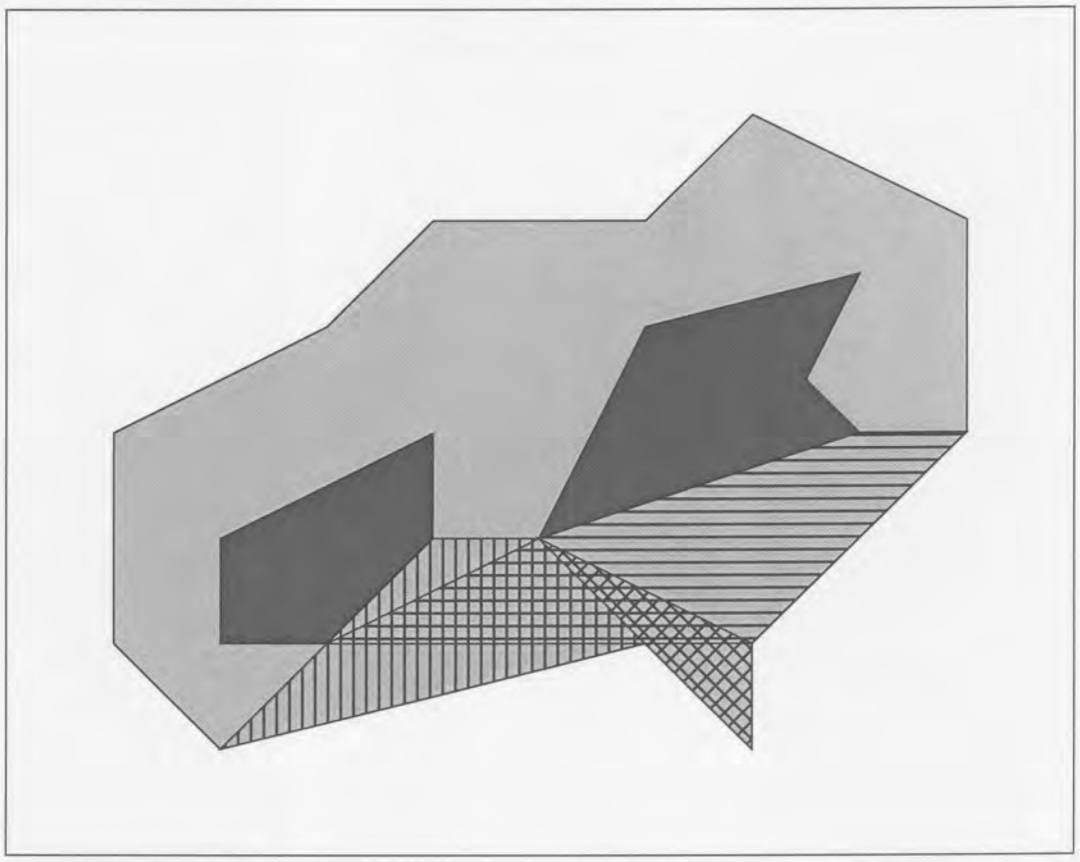

Figure 2.22: Part of a *cover* with convex polygons (note the overlapping of the convex polygons)

composition can be categorised in two different ways – *covering* and *partitioning*. The basic covering problem is to find the minimum number of polygons, with some predefined characteristics, to cover the complete area of an enclosing polygon. The partitioning problem is the same as the covering problem except that the polygons into which the enclosing polygon is decomposed are not permitted to overlap.

The phase of the space syntax analysis method which produces a convex map of a given urban layout is clearly a polygon decomposition problem. The town plan is a polygon with holes and the convex map is a minimum partition of that polygon. In space syntax the convex map of the urban layout is made up of the smallest number of convex spaces that "cover" all of the space in the area being analysed. A convex space gives some local information about the area in the sense that every point in the convex space is directly visible and directly accessible to every other point in that convex space. There are two ways of interpreting this definition. The first, that it is a *cover* in the computational geometry sense of covering. That is that overlapping convex polygons are allowed. Figure 2.22 shows part of a *cover* of the original polygon by convex polygons. The second way to interpret the convex map

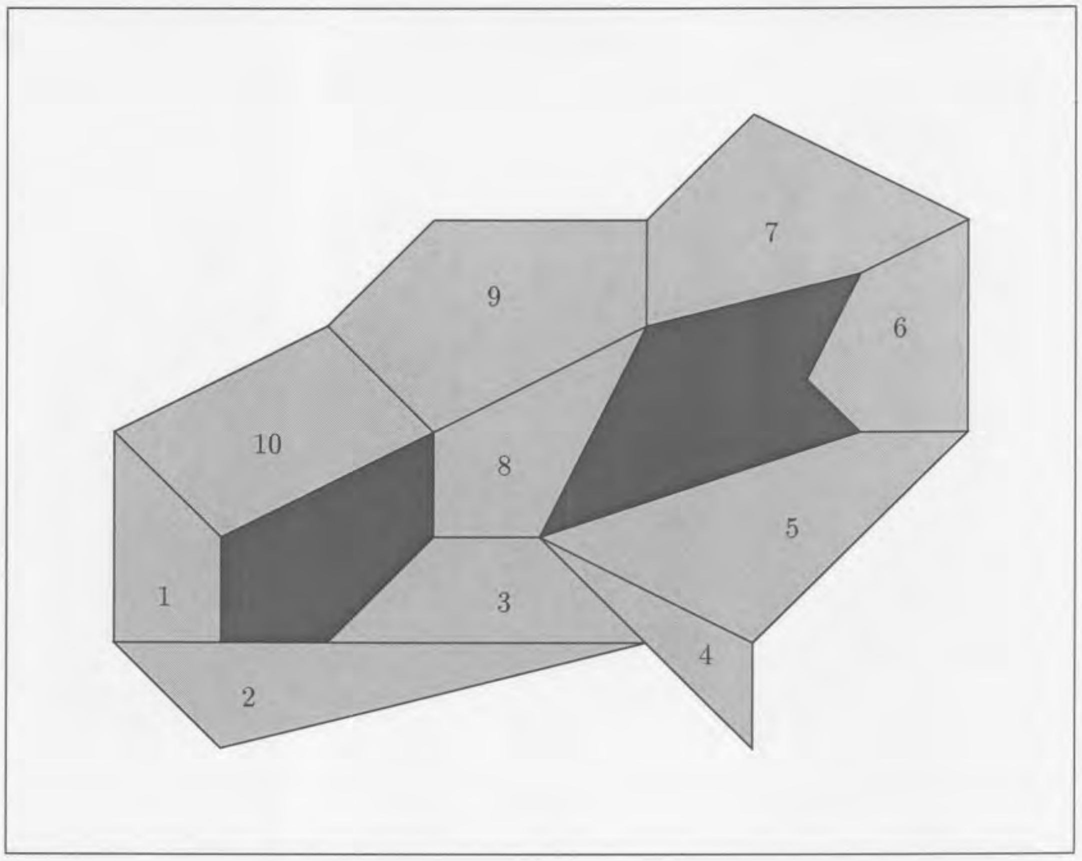

Figure 2.23: A minimum partition with convex polygons

definition is that it is a *partition* of the space into the minimum number of convex polygons. In this case the convex polygons are not allowed to overlap. Figure 2.23 shows a minimum partition of the original polygon into convex polygons. There are other partitions which are also made up of 10 convex polygons but there are no partitions with fewer than 10 such polygons.

In this research the second interpretation of the convex map is adopted. Thus a convex map is a *partition* into the smallest number of convex polygons. The reason for this is that the original work by Hillier *et al.* [1983] clearly intends *partition* rather than *covering*. Mills [1992] also demonstrates that partition is intended. This means that any time the space syntax method is applied to an urban layout, the polygon with holes which represents the layout must be partitioned into the smallest number of convex polygons. This problem has been shown to be NP-Hard [Lingas, 1982] which means that in general the problem cannot be solved exactly in polynomial time and so approximations are required.

Once a convex map (exact or a reasonable approximation) has been found, the next step in space syntax is to place the axial lines which cross all of the shared edges between the convex polygons in the partition. Figure 2.24 shows a placement

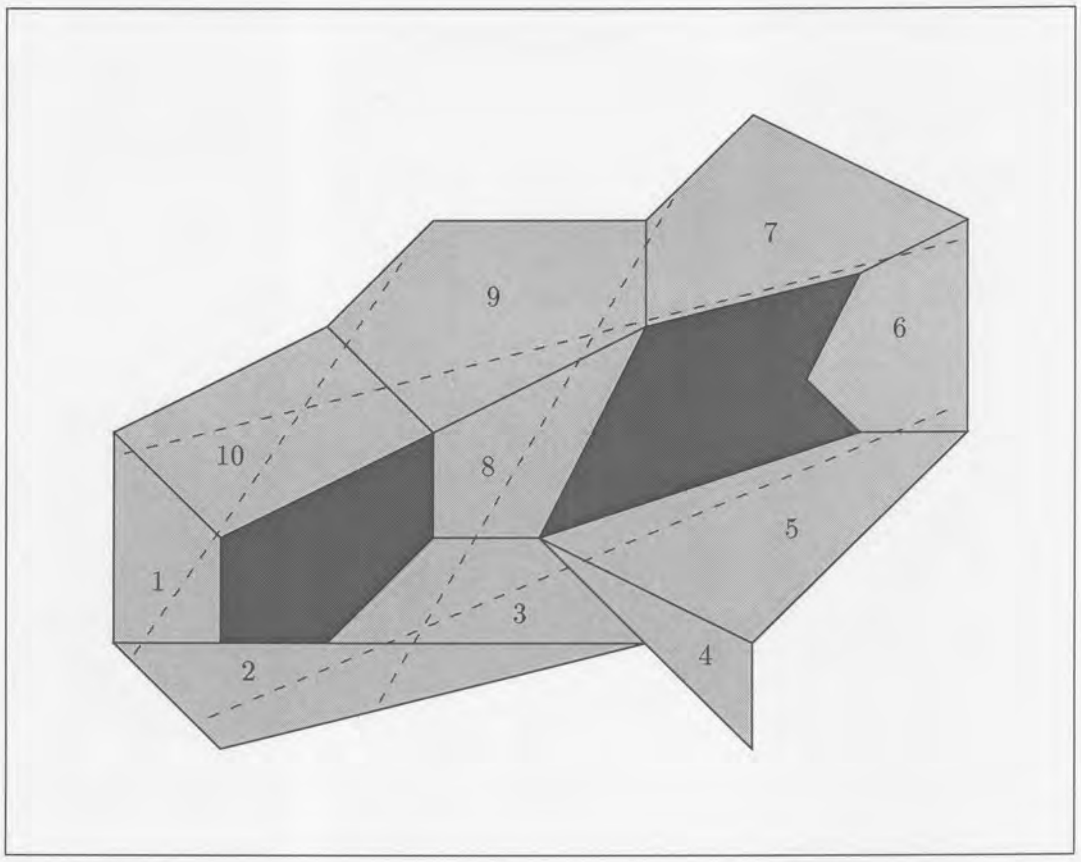

Figure 2.24: An example of placing a minimum number of maximal axial lines

of the minimum number of maximal axial lines to cross all of the shared edges between the convex polygons of the partition in Figure 2.23.

The focus of this research is in finding ways of placing the minimum number of axial lines to cross all of the shared edges in the minimum partition of some urban layout. This is a problem which has not been tackled elsewhere but the problem does benefit from research in the area of visibility in polygons.

A problem which seems similar to axial line placement is that of *stabbing* [Pellegrini and Shor, 1992; Pellegrini, 1993, 1997]. A *stabber* is a line l that intersects every member of a collection $\mathcal{P} = \{P_1, \ldots, P_k\}$ of polyhedral sets. Figure 2.25 shows a two-dimensional simplification of the problem. In this case the problem would be to determine if one line can be found which intersects all of the squares. Here no stabber exists – 3 lines are required to stab the boxes. This problem is different from that of placing axial lines because the squares can overlap; the lines do not have to remain within the squares; and the problem is to determine whether a line exists to stab all of the squares rather than finding the minimum number of lines to stab all of the squares.

In order to determine the minimum number of axial lines, it is necessary at some

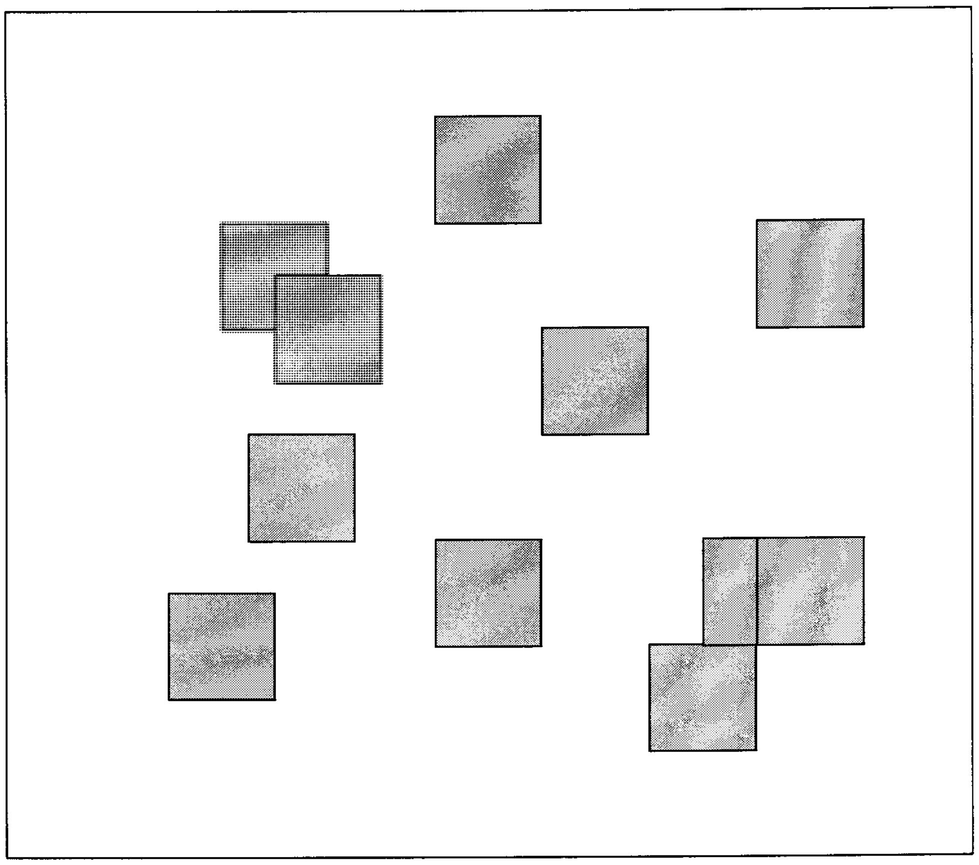

Figure 2.25: An example of stabbing boxes in two-dimensions – no stabbing line exists in this case

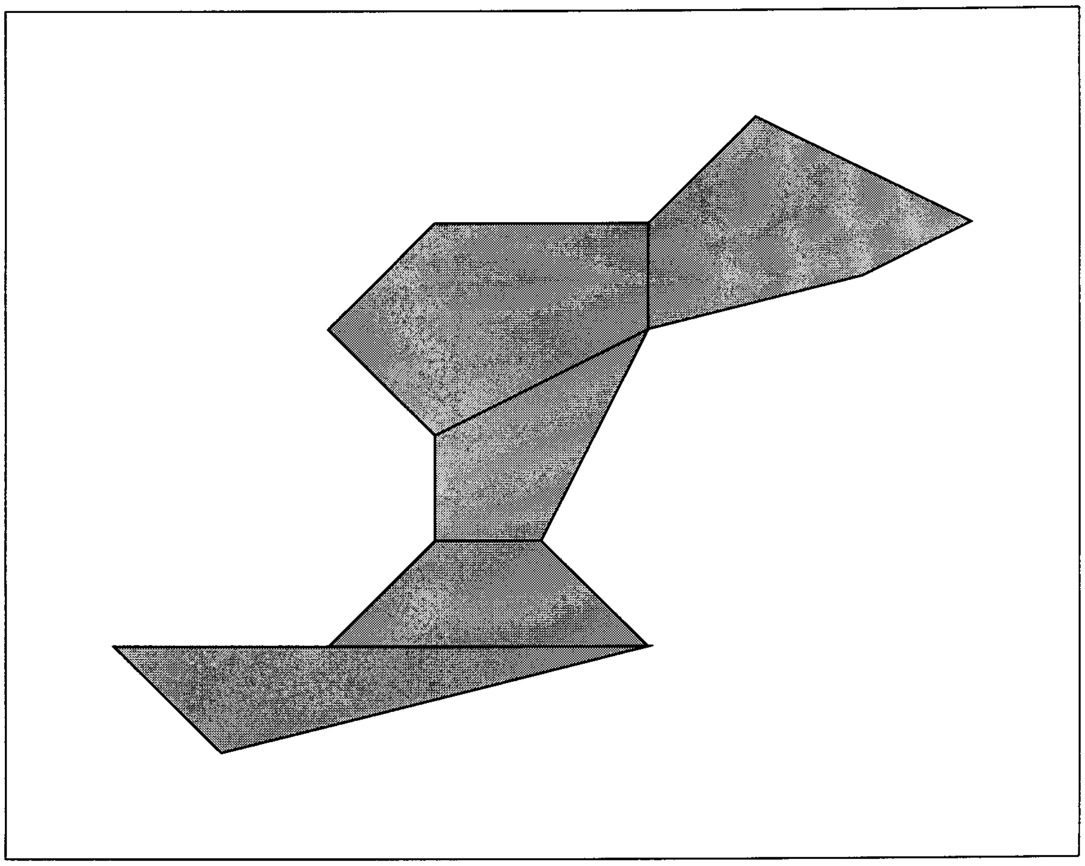

Figure 2.26: A subset of the minimum partition where it is necessary to determine if an axial line can be placed to cross the adjacencies between the convex polygons

point in the process to determine if an axial line can be placed to cross the adjacencies between any subset of the polygons in the partition. Consider the situation where it is necessary to determine whether a single axial line can be placed to cross all of the adjacencies in the subset of convex polygons as shown in Figure 2.26.

The convex polygons in this configuration can be said to form a "chain" of polygons. ***ALP*** requires that any axial line to cross the adjacencies in this chain would have to remain inside the union of the convex polygons concerned. Clearly this problem could be solved by determining if there exists a new line segment which intersects all four of the line segments representing the adjacencies and is always inside the union of the five convex polygons concerned. Solving this problem can also be posed as a visibility problem. Can some part of the edge labeled *first* in the heavily outlined polygon of Figure 2.27 see some part of the edge labeled *last* in this same polygon? The visibility problem would give the answer "yes" if there is a line of sight from some point (or points) on edge *first* to some point (or points) on edge *last*. The definition of visibility means that any line of sight must not leave the polygon. If there is this type of visibility then an axial line could be placed from

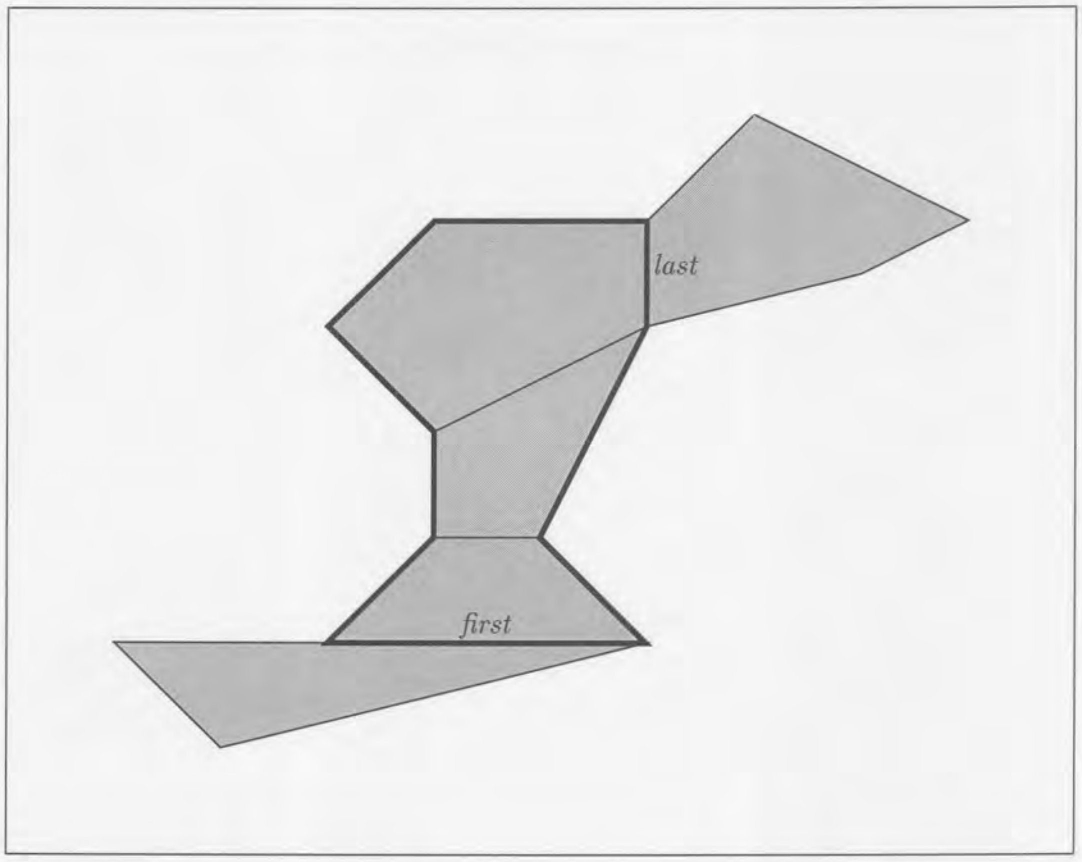

Figure 2.27: The relationship between attempting to place an axial line through a number of adjacencies and edge to edge visibility in a polygon

the first to the last adjacency in the original collection of convex polygons. The fact that the convex polygons concerned form a chain means that this line would also cross all of the other adjacencies in the chain. If there is no visibility then one axial line could not be placed to cross all of the adjacencies and a smaller subset of the original partition would have to be considered. The solution to this visibility problem, which is due to Avis *et al.* [1986], is discussed in detail in Section 2.4.4.3.1 below. Clearly all possible chains would have to be considered to find the minimum number of axial lines but these other chains can be treated in a similar fashion.

In the final analysis stage of space syntax the axial lines are transformed into a "depth graph" which gives information about how easy it is to get from one point in the layout to another point. This ease of movement is based on the number of axial lines a person in the layout would have to move along to get from the start point to the end point. Suppose that someone wanted to know how easy it was to get from point A to point B in Figure 2.28. The starting point A is in the convex polygon 1 which has axial line a passing through it. From the point of view of space syntax, all points in polygon 1 are equivalent so A can be interpreted as a point on axial

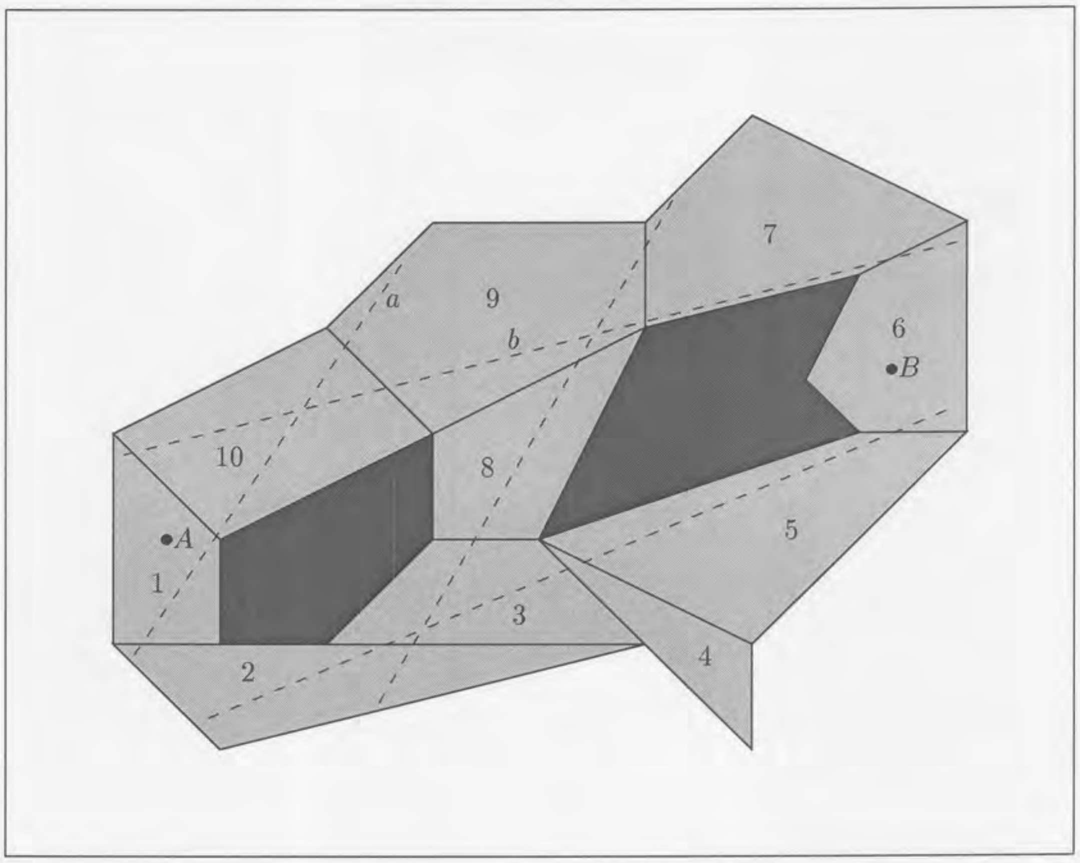

Figure 2.28: Relating L_k visibility to axial lines

line a. The person would then travel along the axial line a until it intersects axial line b. The person would then turn onto line b and travel along that line until convex polygon 6 is reached. This convex polygon contains B so the path is complete. Thus getting from A to B requires two axial lines (and one turn). As discussed earlier A and B are L_2 visible. They can be joined by a path made up of two line segments. There is thus some similarity between the placing of axial lines and determining L_k visibility between points in the polygon.

As many of the problems in the areas of guarding and polygon decomposition have been proven to be NP-hard or NP-Complete, researchers have focussed on restrictions to the general problems. Orthogonal or orthogonally convex polygons have received a lot of attention by researchers. One instance of this is the problem of guarding a *traditional art gallery* [Urrutia, 1999]. Such an art gallery is housed in a rectangular building subdivided into rectangular rooms. Any two rooms have a door connecting them. See Figure 2.29 for an example of such a gallery. In this situation, if a guard is placed in a doorway connecting two rooms then she can guard both rooms at once. Also no guard can completely guard three rooms. This means that at least $\lceil n/2 \rceil$ guards are required if there are n rooms in the gallery. Czyzowicz

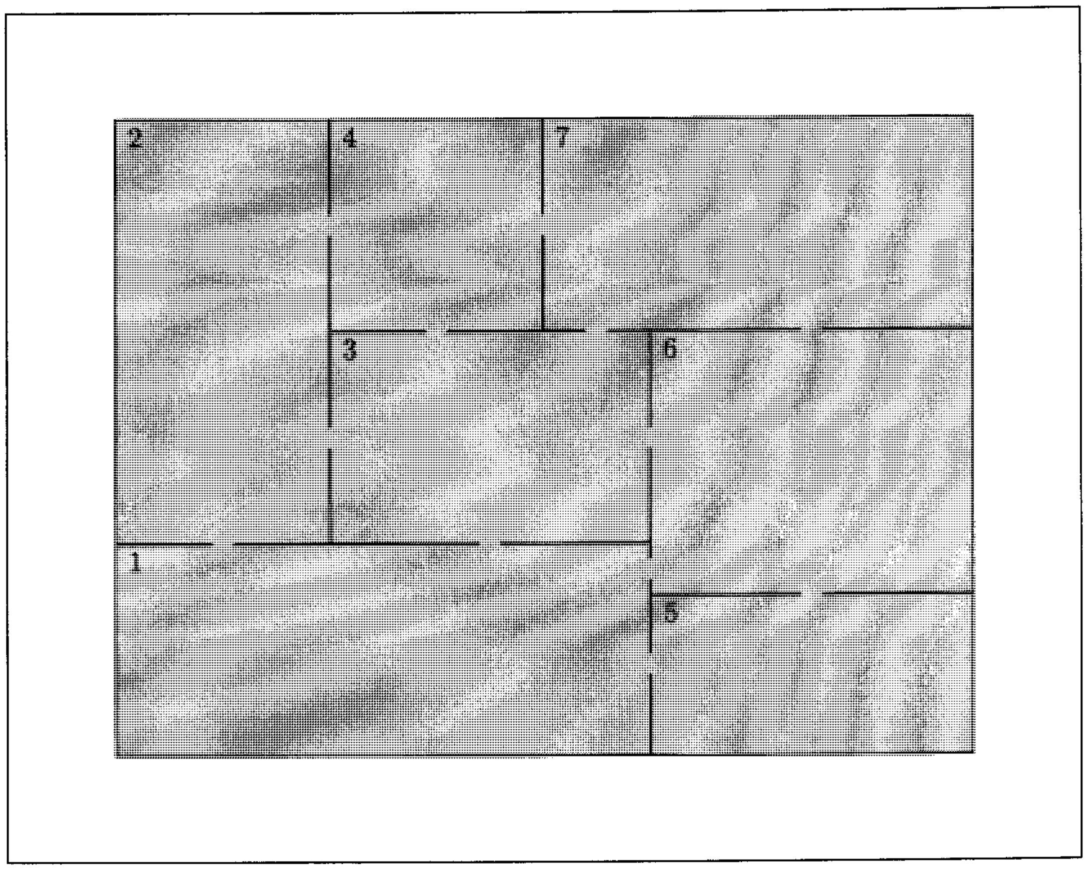

Figure 2.29: A traditional art gallery

et al. [1994] prove that any rectangular art gallery with n rooms can be guarded by exactly $\lceil n/2 \rceil$ guards.

The early work on the axial line placement problem was in some sense quite similar to the idea of guarding a traditional art gallery. This early work in ***ALP*** also focussed on a restriction of the original problem. Instead of attempting to place axial lines to cross the adjacencies between the convex polygons of some minimum partition of a polygon with holes the work focussed on placing axial lines in configurations of adjacent orthogonal rectangles. In the first instance the problem was further restricted by insisting that the axial lines were also orthogonal (see Chapter 4). This restriction was later relaxed and lines of arbitrary orientation were allowed (see Chapter 5). These two problems were originally termed "ray guarding". The idea was that one would place "guards" who could see along some "ray". Each ray guard could be thought of as placing a video camera (or some other monitoring device) which is fixed to point in a certain direction. In an art gallery situation, one could consider this as guarding the flow between the rooms which make up the gallery by monitoring each doorway. ***ALP*** requires that the minimum number of axial lines must be placed to cross the adjacencies between the room rectangles. The

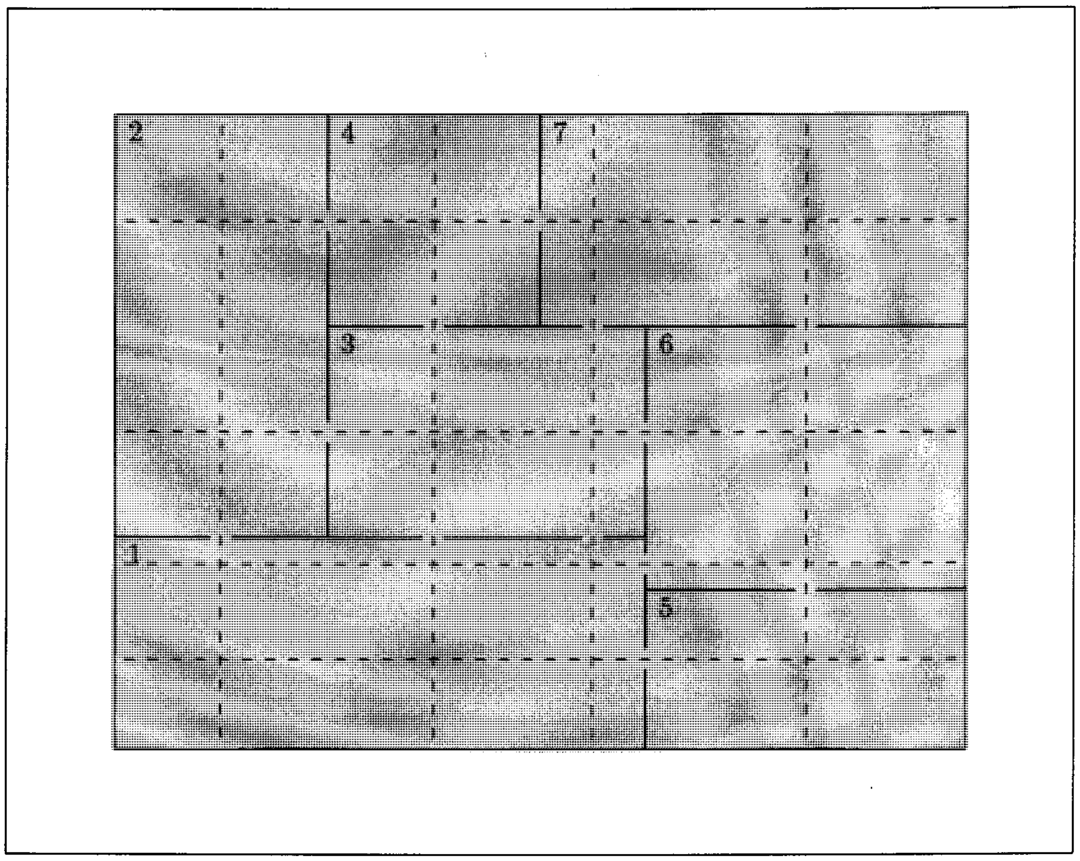

Figure 2.30: "Ray guarding" a traditional art gallery with orthogonal ray guards

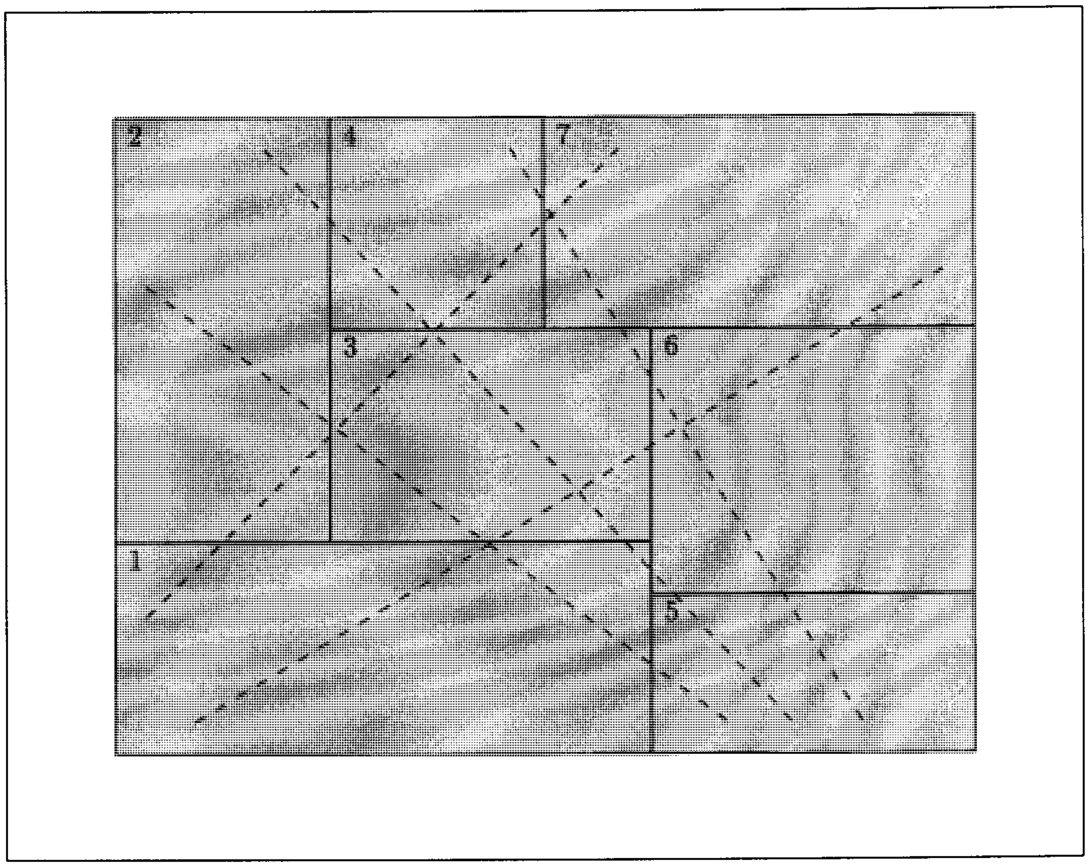

Figure 2.31: A placement of axial lines of arbitrary orientation in a traditional art gallery

point where an axial lines crosses an adjacency can be considered as an appropriate place to position a door between the two adjacent rooms. The doorway can then be guarded by a ray guard placed at the beginning or end of the axial line which crosses the adjacency. As few of these ray guards as possible are required. Placing as few axial lines as possible to cross the adjacencies is then equivalent to determining where to place the doorways between the rooms such that each doorway is guarded by a ray guard. ***ALP*** also requires that each axial line should be as long as possible. This means that any axial lines which is placed to cross a particular adjacency should be extended to cross any other adjacencies that it can even if these adjacencies have already been crossed by other axial lines. In the art gallery scenario this is equivalent to placing additional doors anywhere that a ray guard's vision is obstructed by an interior wall. These additional doorways would still be guarded by at least one ray guard. Figure 2.30 shows the traditional art gallery of Figure 2.29 guarded by ray guards whose lines of sight are restricted to being orthogonal. Note that the wall between rooms 1 and 3 has two doorways in it rather than only one. This arises because an axial line is required to cross the adjacency between rectanges 3 and 4 and an axial line is required to cross the adjacency between rooms 3 and 7. When these axial lines are extended to cross as many adjacencies as possible they both cross the adjacency between rectangles 1 and 3.

If the ray guards' lines of sight are not restricted to being parallel to the coordinate axes then a different positioning of guards results. This would correspond to a placement of not necessarily orthogonal axial lines. A placement of axial lines of arbitrary orientation is shown in Figure 2.31. This placement of axial lines could be converted to a ray guarding situation by placing a door at each point where an axial line crosses an adjacency (a wall between two rooms). Again some of the walls would have more than one doorway.

The discussion in this section has been focussed on some of the similarities and differences between ***ALP*** and the research which has been done in visibility in polygons; guarding of polygons; and decomposition of polygons. ***ALP*** has not been studied before but clearly because the problem areas are not that far removed research into solving ***ALP*** should be informed by work in the other areas. The next section discusses in more detail some specific research which has relevance to ***ALP***.

2.4.4 Results that informed the research on *ALP*

2.4.4.1 Overview

As discussed above there are some commonalities and some differences between ***ALP*** and other research areas in the field of computational geometry. There also general approaches taken and specific techniques used in this other research which informed the research undertaken in this thesis. These ideas are discussed in this section. First some of the general approaches are discussed and then some specific techniques are focussed on.

2.4.4.2 General Approaches

2.4.4.2.1 NP-Hardness results Many of the problems studied in guarding, decomposition or visibility have been shown to be NP-Hard or NP-Complete (see for example the work of Lee and Lin [1986]). As ***ALP*** seems to be a very similar type of problem to these other problems it seemed likely that ***ALP*** too could be NP-Hard. This insight informed the way that the research questions to be tackled in this research were posed. See Chapter 3 for more detail on this. It also meant that the work on ***ALP*** progressed on two fronts – attempting to find a polynomial time algorithm to solve the problem and attempting to prove that the problem was NP-Hard/NP-Complete. In the NP-Completeness proofs the same standard NP-Complete problems used in these other problems were considered (***3SAT*** [Lee and Lin, 1986] and ***vertex cover*** [Ntafos, 1986]). Garey *et al.* [1976], Garey and Johnson [1979] and Lichtenstein [1982] were also consulted.

A specific result that has direct relevance to the possible automation of the space syntax method is the fact that partitioning a simple multiply connected polygon (a polygon with holes) is NP-Hard [Lingas, 1982; O'Rourke and Supowit, 1983]. An urban layout would be modelled as a polygon with holes and this result means that a convex map for the urban layout cannot, in general, be found in polynomial time. There might be cases (urban layouts) where the convex map could be found in polynomial time and determining and describing these cases would be a useful research area. Another interesting research area would be in finding good approximations to the exact solution in reasonable time. However, since it was known that partitioning a polygon with holes is NP-Hard and less was known about the problem of placing axial lines in the urban layout, a decision was made to focus this research on the new problem rather than to extend the research on the known domain. Chapter 3 expands on this decision. Chapter 7 dicusses some work which was done in looking at special cases of urban layouts where the partition can be found in polynomial time but much more work in this area can still be done.

2.4.4.2.2 Restrictions and approximations As mentioned earlier many of the problems which are similar to ***ALP*** have been proven to be NP-Hard (or NP-Complete). This has meant that subsequent work on these problems has taken one of two routes – considering restrictions of the general problem in the hope that these problems can be solved in polynomial time or devising algorithms to find solutions which approximate the optimal ones. Both of these approaches are discussed below.

Restricting the problem A common restriction is to consider orthogonal polygons rather than general poygons. In the area of guarding, the orthogonal art gallery theorem was first formulated and proved by Kahn *et al.* [1983]. It states that $\lfloor n/4 \rfloor$ guards are always sufficient to see the interior of an orthogonal art gallery room and that in some circumstances this number of guards is necessary. Their proof is similar to that of Fisk [1978] using orthogonal comb polygons. The main idea of their proof was to partition an orthogonal polygon into convex quadrilaterals, add the internal diagonals of these quadrilaterals and then four-vertex colour the resulting graph. For orthogonal polygons with holes Shermer conjectured that $\lfloor \frac{n+h}{4} \rfloor$ vertex guards are sufficient to guard any orthogonal polygon with holes [O'Rourke, 1987]. This conjecture is still open [Urrutia, 1999].

The traditional art gallery problem – placing guards to cover rectangular rooms in a rectangular building – is another restriction of the general problem. Restricting the problem in this way has enabled researchers to prove a tight bound on the problem and to determine how to place the guards [Czyzowicz *et al.*, 1994].

The partitioning of orthogonal polygons, with and without holes, has also been studied. Much of the work done in this area is to partition orthogonal polygons into the minimum number of rectangles which generally means that Steiner points are required [Keil, 1999]. If Steiner points are disallowed (which is the case in the space syntax method) then the attention is focussed on partitioning the orthogonal polygon into quadrilaterals. Other work has focussed on partitioning orthogonally convex polygons.

In the area of visibility, work has also focussed on orthogonal polygons particularly with respect to staircase and dent visibility [Asano *et al.*, 1999].

The results in guarding, partitioning and visibility with regard to orthogonal polygons and rectangles show that in some cases these problems are easier to solve than the general problems. This indicated that studying similar restrictions in ***ALP*** would be reasonable approach to take. In particular, a configuration of adjacenct rectangles could be considered as a rectangular partition of some orthogonal polygon representing an urban layout. In ***ALP*** the shared edges between the rectangles must be crossed by the minimum number of axial lines. These axial lines could be orthogonal or have arbitrary orientation. Chapter 3 discusses these restrictions of the general problem in more detail and they are addressed in Chapters 4 and 5 respectively.

Approximations A number of the general problems in guarding, partitioning and visibility, and even a number of the restrictions of these general problems, have been shown to be NP-Hard (or NP-Complete). It is, however, still worthwhile in many instances to have an approximation to the exact solution.

In the problem of guarding an general art gallery Avis and Toussaint [1981] used Fisk's proof as the basis for implementing an $O(n \log n)$ algorithm to assign positions to the guards. Algorithms such this do not guarantee a minimum solution for a given polygon but will place a guard set which is always sufficient (and sometimes necessary) to guard the polygon. Bjorling-Sachs and Souvaine [1995] also used an upper bound proof as the basis of their algorithm to place $\lceil \frac{n+h}{3} \rceil$ guards in a polygon with n vertices and h holes. Other such results can be found in Urrutia [1999].

In partioning polygon without holes many of the early results were approximations to the exact solution (see Schachter [1978]) and it was only later that exact solutions where found [Keil, 1985]. When the polygon contains holes the problem is NP-Hard [Lingas, 1982], and so approximations are required. Keil [1999] discusses some approximation algorithms for various versions of the problem of partitioning a polygon with holes.

This thesis considers the computational geometry problems which arise out of a possible automation of the space syntax method. If ***ALP*** is NP-Hard or NP-Complete (or if variations/restrictions of ***ALP*** are NP-Hard or NP-Complete) then an automated process would not be able to generate the exact axial map of an urban area in a reasonable time and so approaches to find approximate solutions which would be acceptable to someone who wishes to apply the method would have to be found. Chapters 4 and 5 discuss some approaches for finding approximate solutions to some variations of ***ALP***.

In addition, the fact that partitioning a polygon with holes is NP-Hard means that it cannot be guaranteed that the convex map of an urban area could be found in reasonable time. Thus research into finding efficient approximations to the convex map would be a worthwhile research endeavour. This thesis does not focus on the problem of finding the convex map of an urban area but Chapter 7 introduces the idea of a "deformed urban grid" and discusses an algorithm which finds a not necessarily minimum partition of such a polygon with holes.

2.4.4.3 Specific results of importance

2.4.4.3.1 Partial edge visibility ***ALP*** is the problem of creating the axial map of an urban layout given the convex map of the layout. This involves finding the minimum number of axial lines to cross the adjacencies between the convex spaces (convex polygons) which partition an urban area (a polygon with holes). As stated previously, the placing of an axial line to cross the adjacencies in a chain of adjacent convex poloygons can be thought of as determining edge to edge visibility between the adjacency between the first two polygons in the chain and the adjacency between the last two polygons in the chain. In this case, all that is required is that some point on the first adjacency can see some point on the last adjacency. This is partial visibility – edge uv is said to be partially visible from edge xy if there exists a point w on edge uv and a point z on edge xy such that w and z are visible.

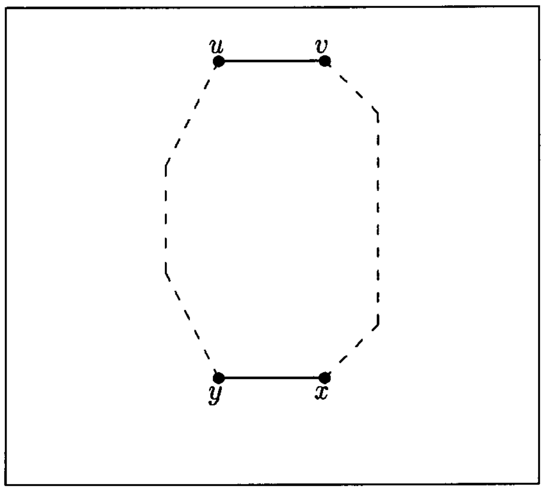

Figure 2.32: The edge to edge visibility algorithm [Avis *et al.*, 1986] – totally facing edges

Avis *et al.* [1986] give a linear time algorithm for determining edge to edge visibility in a simple polygon. This algorithm is important to this work as it can be used in determining whether axial lines can be placed in chains of convex polygons. The algorithm is used in developing the heuristics discussed in Chapter 5.

The algorithm (actually two algorithms) is described in detail by Avis *et al.* [1986]. The approach is quite complicated and so a greatly simplified overview is given below. The reader is referred to the original article for more details.

The input to the algorithm would be a simple polygon $P = (p_1, p_2, \ldots, p_n)$ and two edges in P, uv and xy. For the sake of this explanation assume that the two edges, uv and xy, are as shown in Figure 2.32. The dashed lines connecting the two edges of interest indicate that the boundary of the polygon could have any shape between the two points connected by the dashed lines. Avis *et al.* [1986] define this situation as two edges that *totally face* each other. This is the simplest situation which could occur.

The algorithm works as follows.

1. Construct the quadrilateral $Q(u, v, x, y)$
2. Construct the chains $C(v, x)$ and $C(y, u)$ of the vertices on the path from v to x and y to u respectively.
3. If $C(v, x)$ or $C(y, u)$ cuts through $Q(u, v, x, y)$ then there can be no visibility. Figure 2.33 shows an example of $C(y, u)$ cutting through $Q(u, v, x, y)$.
4. If neither $C(v, x)$ nor $C(y, u)$ cuts through $Q(u, v, x, y)$ then there can be visibility. In this case, find the reduced chains $R(v, x)$ and $R(y, u)$ by determining which parts of the chains $C(v, x)$ and $C(y, u)$ would be in $Q(u, v, x, y)$.

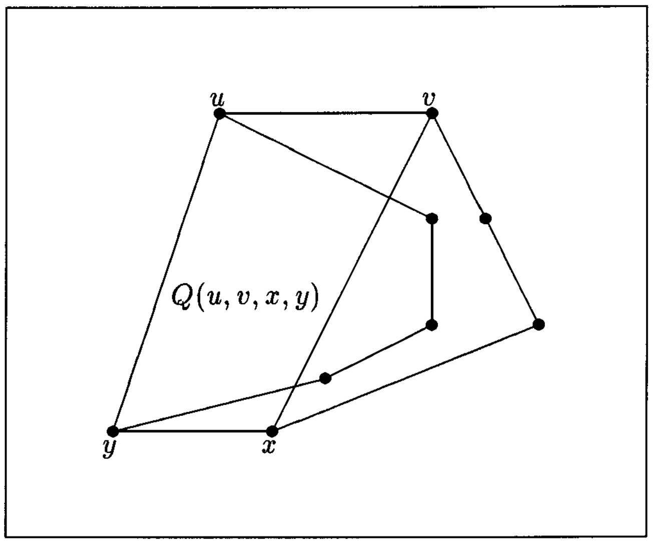

Figure 2.33: The edge to edge visibility algorithm – chain $C(y, u)$ cutting through quadrilateral $Q(u, v, x, y)$, no visibility is possible

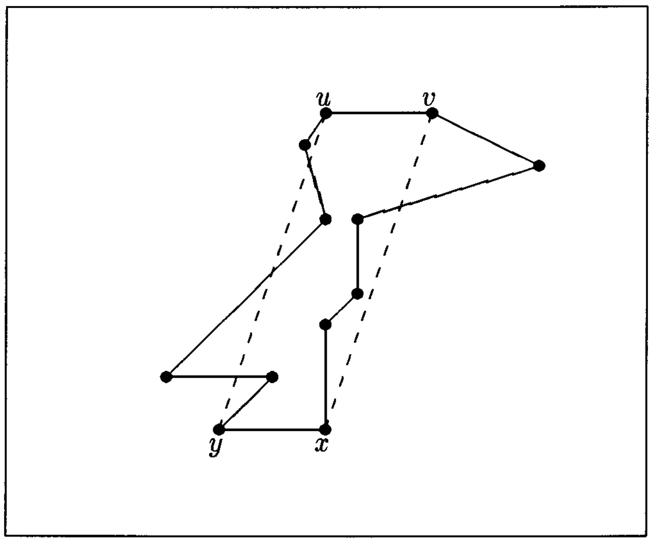

Figure 2.34: An example of the edge to edge visibility algorithm – the input polygon and $Q(u, v, x, y)$

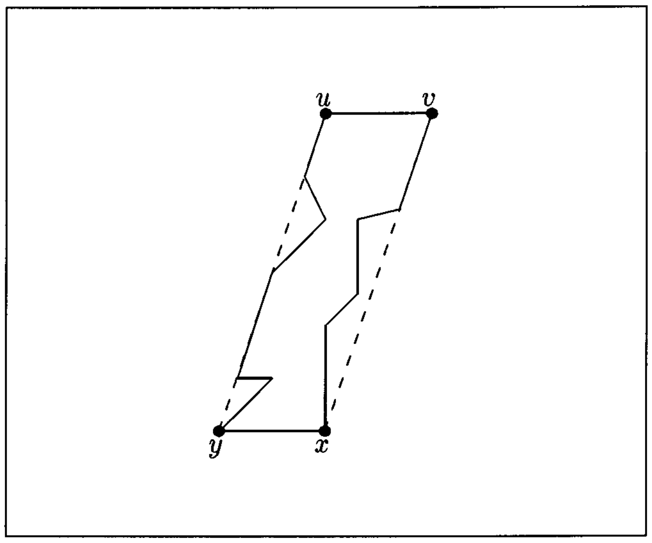

Figure 2.35: An example of the edge to edge visibility algorithm – the reduced chains

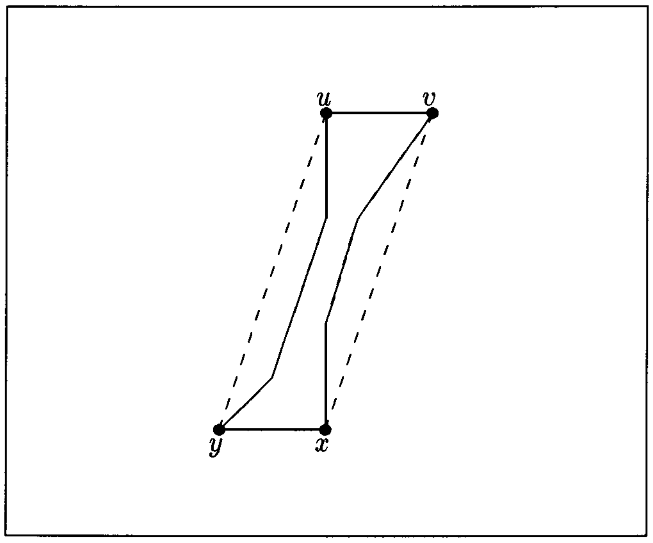

Figure 2.36: An example of the edge to edge visibility algorithm – the inner convex hulls

Figure 2.34 shows an example of an input polygon and Figure 2.35 shows the corresponding reduced chains.

5. Calculate the inner convex hulls $ICH(v, x)$ and $ICH(y, u)$ of $R(v, x)$ and $R(y, u)$. Figure 2.36 shows the inner convex hulls for the example polygon.
6. Construct the polygon $H = ICH(v, x), xy, ICH(y, u), uv$
7. If H is a simple polygon then edge uv is partially visible from edge xy and vice versa. In the example (Figure 2.36), H is a simple polygon so there is partial visibility between the edges uv and xy.

The algorithm also identifies complete visibility, strong visibility and weak visibility in the input polygon but these are not of interest here. More important is that, in the case of partial visibility, the algorithm determines which region of edge uv is visible to edge xy and *vice versa*. This gives the information about where an axial line could be placed in ***ALP***.

2.4.4.3.2 Guarding in grids Ntafos [1986] and Gewali and Ntafos [1993] define the *complete two-dimensional grid* of size n as the graph with vertex set $V = \{1, 2, \ldots, n\} \times \{1, 2, \ldots, n\}$ and the edge set $E = \{\{(i, j), (k, m)\} : |i - k| + |j - m| = 1\}$ where all edges are parallel to the major axes – see Figure 2.37. In a geometric setting, the grid edges can be thought of as corridors and the grid vertices as intersections of corridors. A (partial) grid is any subgraph of the complete grid. Gewali and Ntafos [1993] also define a *grid segment* as a succession of grid edges along a straight line bounded at either end by a missing edge. A *simple grid* is a grid where all of the endpoints of the grid segments lie on the outer face of the planar subdivision formed by the grid. A *general grid* is a grid which can have holes – some of the endpoints of the segments may lie on the inner face of the planar subdivision.

The star cover or star guard problem in a grid is then to find the minimum number of guards that need to be stationed in the grid so that each point in the grid is visible to some guard [Ntafos, 1986]. If the grid is complete then n guards are necessary and sufficient for a two-dimensional grid of size n. If the grid has obstacles in it – there are missing portions of the grid – then the problem becomes more interesting. Ntafos [1986] shows that a minimum cover for a grid with obstacles can be found in polynomial time by reducing the problem to that of finding a maximum mapping in a bipartite graph.

The idea of a grid is useful in ***ALP*** because part of many cities can be considered as grid-like structures. The result that the star cover can be found in polynomial time suggested that ***ALP*** might also be solvable in polynomial time. Any guard in the star cover guards at most two grid segments – one vertical and one horizontal. It

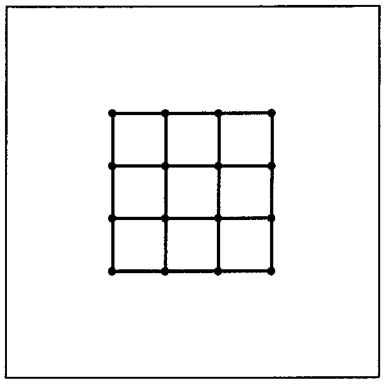

Figure 2.37: A complete grid of size 4

thus seems as though two axial lines could be placed to cover the lines of sight of each star guard. This idea is further developed in Chapter 7.

2.5 Conclusion

In Chapter 1 the idea of space syntax is introduced and the broad areas of computer science research which would arise from attempting to automate the approach are discussed. The range of research questions that could have been addressed is very wide and so it was necessary to choose a smaller area of research for this thesis. The decision was made to focus the research on the problem of finding the axial lines that cross all of the shared boundaries between the convex polygons in the convex map, that is finding the axial map for a given layout – ***ALP***. The problems of separating space from non-space, determining the convex map and the final analysis stage with its associated algorithms were not considered as part of this research.

This chapter begins by introducing some terminology in the fields of computational geometry and graph theory and giving an overview of NP-Completeness. This is done to provide the reader with a framework for understanding the summary of the research which is related to automating the space syntax method and in particular ***ALP***. This related research can be loosely categorised into three areas – guarding of polygons, visibility in polygons and polygon decomposition. This chapter gives an overview of the research in each of these related areas before addressing the commonalities and differences between ***ALP*** and guarding, visibility and polygon decomposition problems. From the dicussion above, it should be clear that ***ALP*** is a new area of research (there are significant differences between ***ALP*** and the related research). A detailed investigation of ***ALP*** is thus warranted. In addition, the fact that many of the related problems are computationally hard indicates that solving this problem or making progress to solving this problem would be a significant research contribution.

The last section of the chapter identifies some related research that informed that research into ***ALP***. First, the fact that many of the problems are NP-Hard or NP-Complete seemed to have relevance for the research in ***ALP***. The fact that these problems are hard means that finding ways of restricting the general problem or of producing reasonable approximate solutions are viable areas of research. This might apply in the case of ***ALP*** as well. Second, some specific results were identified as having direct relevance to this research. The section concludes by discussing two such specific results – partial edge visibility and guarding in grids.

Chapter 3 considers the possible research questions which arise from the decision to focus the work for this research on ***ALP***. The approach taken in posing these questions is based on the related work presented in this chapter.