1

PM-1 3 1/2"x4" PHOTOGRAPHIC MICROCOPY TARGET
NBS 1010a ANSI/ISO #2 EQUIVALENT

National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Document communiqué

Document communiqué

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

On Asymptotic Algorithms For Approximating Arbitrary Polygons By Simpler Convex Polygons

by

Emran Rafique

A thesis submitted to the Faculty of Graduate Studies
and Research in partial fulfillment of the requirements
for the degree of Master of Computer Science.

Ottawa-Carleton Institute of Computer Science
School of Computer Science
Carleton University
Ottawa, Ontario.
March, 1993.

© Copyright 1993
Emran Rafique.

National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395 rue Wellington
Ottawa (Ontario)
K1A 0N4

Vous êtes l'auteur d'une thèse

Vous êtes l'auteur d'une thèse

The author has granted an irrevocable non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of his/her thesis by any means and in any form or format, making this thesis available to interested persons.

L'auteur a accordé une licence irrévocable et non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de sa thèse de quelque manière et sous quelque forme que ce soit pour mettre des exemplaires de cette thèse à la disposition des personnes intéressées.

The author retains ownership of the copyright in his/her thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without his/her permission.

L'auteur conserve la propriété du droit d'auteur qui protège sa thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

ISBN 0-315-84090-0

Canada

Name _____

Dissertation Abstracts International is arranged by broad, general subject categories. Please select the one subject which most nearly describes the content of your dissertation. Enter the corresponding four-digit code in the spaces provided.

U·M·I

SUBJECT TERM

SUBJECT CODE

Subject Categories

THE HUMANITIES AND SOCIAL SCIENCES

COMMUNICATIONS AND THE ARTS

Architecture 0729
Art History 0377
Cinema 0900
Dance 0378
Fine Arts 0357
Information Science 0723
Journalism 0391
Library Science 0399
Mass Communications 0708
Music 0413
Speech Communication 0459
Theater 0465

EDUCATION

General 0515
Administration 0514
Adult and Continuing 0516
Agricultural 0517
Art 0273
Bilingual and Multicultural 0282
Business 0688
Community College 0275
Curriculum and Instruction 0727
Early Childhood 0518
Elementary 0524
Finance 0277
Guidance and Counseling 0519
Health 0680
Higher 0745
History of 0520
Home Economics 0278
Industrial 0521
Language and Literature 0279
Mathematics 0280
Music 0522
Philosophy of 0998
Physical 0523

Psychology 0525
Reading 0535
Religious 0527
Sciences 0714
Secondary 0533
Social Sciences 0534
Sociology of 0340
Special 0529
Teacher Training 0530
Technology 0710
Tests and Measurements 0288
Vocational 0747

LANGUAGE, LITERATURE AND LINGUISTICS

Language
General 0679
Ancient 0289
Linguistics 0290
Modern 0291
Literature
General 0401
Classical 0294
Comparative 0295
Medieval 0297
Modern 0298
African 0316
American 0591
Asian 0305
Canadian (English) 0352
Canadian (French) 0355
English 0593
Germanic 0311
Latin American 0312
Middle Eastern 0315
Romance 0313
Slavic and East European 0314

PHILOSOPHY, RELIGION AND THEOLOGY

Philosophy 0422
Religion
General 0318
Biblical Studies 0321
Clergy 0319
History of 0320
Philosophy of 0322
Theology 0469

SOCIAL SCIENCES

American Studies 0323
Anthropology
Archaeology 0324
Cultural 0326
Physical 0327
Business Administration
General 0310
Accounting 0272
Banking 0770
Management 0454
Marketing 0338
Canadian Studies 0385
Economics
General 0501
Agricultural 0503
Commerce Business 0505
Finance 0508
History 0509
Labor 0510
Theory 0511
Folklore 0358
Geography 0366
Gerontology 0351
History
General 0578

Ancient 0579
Medieval 0581
Modern 0582
Black 0328
African 0331
Asia, Australia and Oceania 0332
Canadian 0334
European 0335
Latin American 0336
Middle Eastern 0333
United States 0337
History of Science 0585
Law 0398
Political Science
General 0615
International Law and Relations 0616
Public Administration 0617
Recreation 0814
Social Work 0452
Sociology
General 0626
Criminology and Penology 0627
Demography 0938
Ethnic and Racial Studies 0631
Individual and Family Studies 0628
Industrial and Labor Relations 0629
Public and Social Welfare 0630
Social Structure and Development 0700
Theory and Methods 0344
Transportation 0709
Urban and Regional Planning 0999
Women's Studies 0453

THE SCIENCES AND ENGINEERING

BIOLOGICAL SCIENCES

Agriculture
General 0473
Agronomy 0286
Animal Culture and Nutrition 0475
Animal Pathology 0476
Food Science and Technology 0359
Forestry and Wildlife 0478
Plant Culture 0479
Plant Pathology 0480
Plant Physiology 0817
Range Management 0777
Wood Technology 0746
Biology
General 0306
Anatomy 0287
Biostatistics 0308
Botany 0309
Cell 0370
Ecology 0329
Entomology 0353
Genetics 0369
Immunology 0793
Microbiology 0410
Molecular 0307
Neuroscience 0317
Oceanography 0416
Physiology 0433
Radiation 0821
Veterinary Science 0778
Zoology 0472
Biophysics
General 0786
Medical 0760

EARTH SCIENCES

Biogeochemistry 0425
Geochemistry 0906

Geodesy 0370
Geology 0372
Geophysics 0373
Hydrology 0388
Mineralogy 0411
Paleobotany 0345
Paleoecology 0426
Paleontology 0418
Paleozoology 0985
Palynology 0427
Physical Geography 0368
Physical Oceanography 0415

HEALTH AND ENVIRONMENTAL SCIENCES

Environmental Sciences 0768
Health Sciences
General 0566
Audiology 0300
Chemotherapy 0992
Dentistry 0567
Education 0350
Hospital Management 0769
Human Development 0758
Immunology 0982
Medicine and Surgery 0564
Mental Health 0347
Nursing 0569
Nutrition 0570
Obstetrics and Gynecology 0380
Occupational Health and Therapy 0354
Ophthalmology 0381
Pathology 0571
Pharmacology 0419
Pharmacy 0572
Physical Therapy 0382
Public Health 0573
Radiology 0574
Recreation 0575

Speech Pathology 0460
Taxology 0383
Home Economics 0386

PHYSICAL SCIENCES

Pure Sciences

Chemistry
General 0485
Agricultural 0749
Analytical 0486
Biochemistry 0487
Inorganic 0488
Nuclear 0738
Organic 0490
Pharmaceutical 0491
Physical 0494
Polymer 0495
Radiation 0754
Mathematics 0405
Physics
General 0605
Acoustics 0986
Astronomy and Astrophysics 0606
Atmospheric Science 0608
Atomic 0748
Electronics and Electricity 0607
Elementary Particles and High Energy 0798
Fluid and Plasma 0759
Molecular 0609
Nuclear 0610
Optics 0752
Radiation 0756
Solid State 0611
Statistics 0463

Applied Sciences

Applied Mechanics 0346
Computer Science 0984

Engineering
General 0537
Aerospace 0538
Agricultural 0539
Automotive 0540
Biomedical 0541
Chemical 0542
Civil 0543
Electronics and Electrical 0544
Heat and Thermodynamics 0348
Hydraulic 0545
Industrial 0546
Marine 0547
Materials Science 0794
Mechanical 0548
Metallurgy 0743
Mining 0551
Nuclear 0552
Packaging 0549
Petroleum 0765
Sanitary and Municipal System Science 0554
Geotechnology 0790
Operations Research 0428
Plastics Technology 0796
Textile Technology 0795
0994

PSYCHOLOGY

General 0621
Behavioral 0384
Clinical 0622
Developmental 0620
Experimental 0623
Industrial 0624
Personality 0625
Physiological 0989
Psychobiology 0349
Psychometrics 0632
Social 0451

The undersigned hereby recommend to
the Faculty of Graduate Studies and Research
acceptance of the thesis.

**On Asymptotic Algorithms For Approximating
Arbitrary Polygons By Simpler Convex Polygons**

Submitted by Emran Rafique, B.C.S.,
in partial fulfillment of the requirements for the degree of
Master of Computer Science.

Thesis Supervisor

Thesis Co-supervisor

Director, School of Computer Science
Carleton University

Date

ABSTRACT

In numerous applications of Computational Geometry, there arises the need for approximating complicated geometric objects by simpler ones. This problem of approximating a polygon can be effectively translated into the following *inclusion* and *enclosure* problems. Given an arbitrary polygon P , and a convex polygon R , an approximation of the area of P by R can be achieved by computing, in relation to P , a maximal or minimal area inscribing or circumscribing object of type R , respectively.

This thesis gives efficient approximation algorithms for solving the above mentioned inclusion and enclosure problems for simple and convex polygons. The results presented in this thesis compare to the best known results found in the literature. The major contributions of this thesis include asymptotic algorithms for inclusion problems for squares, rectangles, equilateral triangles and arbitrary triangles for simple and convex polygons, and for enclosure problems for squares, rectangles and equilateral triangles for arbitrary polygons.

ACKNOWLEDGMENTS

The author gratefully acknowledges the consistent guidance and support provided by his thesis advisor, Dr. E. Kranakis. The financial assistance, constant encouragement and constructive criticism received from him is highly appreciated. Guidance and support from Dr. J. R.. Sack, the cosupervisor, is also appreciated. Many thanks to Dr. F. Fiala for his efforts and time in carefully reading this thesis. The author is also indebted to Dr. J. Uruttia for useful suggestions.

The author further acknowledges the School of Computer Science, Carleton University, for providing him with the opportunity for completing this research. The author is further obliged to colleague F. Chang for relevant discussions.

Table of Contents

CHAPTER 1

INTRODUCTION.....	1
1.1 Formal Definition of the Problem.....	1
1.2 Some Practical Interests.....	2
1.3 Background Work.....	3
1.4 Thesis Terminology and Notation	5
1.5 Major Results	7
1.5.1 Verification of the Finiteness Criterion	7
1.5.2 The Proposed Approach.....	8
1.5.3 Why Approximation?.....	9
1.5.4 The Model	10
1.5.5 Major Contributions	11
1.6 Outline of the Thesis.	15

CHAPTER 2

INCLUSION PROBLEMS FOR SPECIAL POLYGONS.....	16
2.1 Centrosymmetric Convex Polygons.....	16
2.1.1 Inscribed Square	18
2.1.2 Inscribed Circle.....	23
2.2 Semicircular Polygon	26
2.2.1 Inscribed Right Triangle.....	26
2.2.2 Inscribed Rectangle	37
2.2.3 Inscribed Triangle.....	41
2.2.4 Inscribed Circle.....	44
2.3 Dome Polygons.....	49
Chapter Summary.....	50

CHAPTER 3

INCLUSION PROBLEMS FOR CONVEX POLYGONS.....	52
3.1 Suboptimal Inscribed Triangle	52
3.1.1 Algorithm with Optimality ratio of $1/3$	53
3.1.2 Algorithm with Optimality ratio of $1/2$	55
3.2 Suboptimal Quadrilateral.....	58

3.3 Asymptotically Optimal Inscribed Square	62
3.3.1 Algorithm for the Axial Square Problem	66
3.4 Equilateral Triangles.....	73
3.4.1 Axial Equilateral Triangles	78
3.5 Asymptotically Optimal Inscribed Rectangle	80
3.5.1 Ratio of a Convex Polygon	81
3.6 Asymptotically Optimal Inscribed Arbitrary Triangle.....	91
3.6.1 Fixed Similarity Triangles.....	91
Chapter Summary.....	102

CHAPTER 4

INCLUSION PROBLEMS FOR SIMPLE POLYGONS.....	104
4.0.1 Ratio of a Simple Polygon	104
4.1 Asymptotically Optimal Inscribed Square	105
4.2 Algorithm for Inscribed Axial Square	107
4.2.1 Extension to the Case of Fixed Ratio Axial Rectangles.....	125
4.3 Asymptotically Optimal Inscribed Rectangles.....	126
4.4 Inscribed Equilateral Triangle	129
4.5 Inscribed Axial Equilateral Triangle	130
4.6 Asymptotically Optimal Inscribed Arbitrary Triangle.....	141
Chapter Summary.....	146

CHAPTER 5

ENCLOSURE ALGORITHMS.....	148
5.1 Circumscribed Square.....	149
5.1.1 Minimal Area Axial Covering Square.....	149
5.2 Circumscribed Equilateral Triangle	152
5.3 Circumscribed Rectangle	153
Chapter Summary.....	158

CHAPTER 6

CONCLUSIONS.....	160
6.1 Summary of the Thesis	160
6.2 Directions for Future Research.....	161

REFERENCES.....	164
List of Lemmas.....	167
List of Tables.....	169

Chapter 1

INTRODUCTION

This chapter defines and presents a brief history of the problems discussed in this thesis. The motivations behind such work are discussed in lieu of some practical applications as well as theoretical interests. We summarize our contribution to the area and compare our results with the ones found in the literature. Also included is the thesis summary, and a section on the notation and terminology used throughout this work.

1.1 Formal Definition of the Problem

Let P be a polygon and let R be a given convex polygon such as a triangle, a square, etc. Approximating P by an object of type R , using some objective function such as area, involves the computation of an object of type R which best approximates the area of P . We consider two methods of achieving this goal:

INCLUSION : *Compute the maximal area object of type R which can be completely contained in P .*

ENCLOSURE: *Compute the minimal area object of type R which can completely cover P .*

We shall refer to the solution of the former problem as an inscribing object and the latter as a circumscribing object, and the problems as inclusion and enclosure problems, respectively. Furthermore, the given polygons are the approximated objects and the inscribing or circumscribing objects are the approximating objects.

In the literature, such problems have also been termed as *inscription* and *circumscription* problems. The class of inclusion problems can be viewed as the dual to the class of enclosure problems, and vice versa. This duality becomes apparent in Chapter 5 when we discuss ways of transforming algorithms for inclusion problems to the respective algorithms for the enclosure problems. The objective function for these optimization problems can vary. For instance, perimeters of polygons may also be approximated by simpler geometric figures, where the respective inclusion and enclosure algorithms compute maximal or minimal perimeter inscribing or circumscribing approximating objects, respectively. However, in this thesis, we shall only deal with area as the objective function.

In the light of our motivations thus, this thesis will deal with the following type of questions:

- *What is the maximal area triangle that can be inscribed in a convex polygon ?*
- *What is the maximal area rectangle that can be circumscribed on a convex polygon?*
- *Compute the maximal area square that can be inscribed in a given simple polygon.*

1.2 Some Practical Interests

Apart from the obvious theoretical interests in these problems, the inclusion and enclosure problems mentioned in the previous section have a number of practical applications as well. For instance, stock cutting is one such practical application in the manufacturing industry. Given a sheet of metal or some other material, automated tools cut out largest possible chunks whose shapes are usually fixed geometric objects like triangles

or squares. Another application is in the area of VLSI manufacturing. Given a rectangular silicon wafer with several points of impurities, the problem of finding the largest rectangular array, void of any impurities, boils down to the problem of finding the largest empty rectangle in a given rectangular point space [Dehne, 1990]. Because the area of a polygon can be bounded above by the area of a circumscribing object and below by the area of an inscribing object of some kind, one can represent complicated geometric objects by simpler more manageable ones [Chang and Yap, 1986]. This idea leads to a whole class of other applications in Robotics, Computer Vision, Image Processing, Computer Graphics and Geographical Information Systems. For example, Robots often continuously execute algorithms in connection with Motion Planning, where it is clearly much easier to plan motion trajectories among simpler objects than complicated polygons. In addition to the above motivations to Computational Geometry, there exist many other interests arising from the much older field of Operations Research.

1.3 Background Work

As mentioned previously, the original necessity for solving such problems arose from the much older field of Operations Research. The focus of this section is on the background of such work associated within the realm of Computational Geometry.

The most general type of such problems is, the so called *potato peeling* problem, as follows. Find the convex polygon, with maximal area or perimeter, inscribed inside a given n vertex simple polygon. This problem was first posed by Goodman [Goodman, 1981] who was successful in deriving various mathematical properties of the solutions of this problem. However, Goodman was only able to give criteria for a finite solution for a simple polygon with no more than 5 vertices. This problem has also been studied under the name of the *Convex Skull* problem [Chang and Yap, 1986]. The best known solution is due to Chang and Yap [Chang and Yap, 1986], who give an $O(n^2)$

algorithm when the objective function is area and an $O(n^6)$ when it is the perimeter. Note that the dual of this problem is the familiar problem of computing the convex hull of a simple polygon, which has a linear time algorithm [McCallum and Avis, 1979]. The problem of finding the minimal area k -gon circumscribing a given convex polygon has also been studied. For this problem, Chang and Yap [Chang and Yap, 1984] reduced DePano's [DePano, 1987b] $O(n^{k-2} \log^2 n)$ algorithm down to one with a complexity in $O(n^3 \log k)$. Aggarwal, Chang and Yap further refined this algorithm to $O(n^2 \log n \log k)$ [Aggarwal, Chang and Yap, 1985]. The analogous version of the potato peeling problem has also been studied in the Rectilinear Geometry [O'Rourke, 1984; Toussaint, 1980; Chazelle *et. al.*, 1984].

More closely related to the work in this thesis are the results associated with inclusion and enclosure problems related to simpler convex figures such as triangles and rectangles. In this respect, Dobkin and Snyder [Dobkin and Snyder, 1979] have a linear time algorithm for the problem of finding the maximal area triangle inscribed in a convex polygon. While Melissaratos and Souvaine [Melissaratos and Souvaine, 1992] have an $O(n^4)$ algorithm for inscribing maximal area triangles in a simple polygon. For the associated circumscribing problems for a simple polygon, O'Rourke *et al.* [O'Rourke *et. al.*, 1986] improved an $O(n \log^2 n)$ algorithm of Klee and Laskowski [Klee and Laskowski, 1985] to a linear one. In fact, assuming that the given polygon is cyclically sorted, they prove that their $O(n)$ algorithm for finding minimal area enclosing triangle is optimal. Furthermore, DePano, Yan Ke and O'Rourke [DePano *et. al.*, 1987a] give an $O(n^2)$ algorithm for finding the maximal area inscribed square and equilateral triangle in a convex polygon. In the same paper they also extend their result to the case of finding the maximal area inscribed equilateral triangle in an arbitrary polygon in time $O(n^3)$. The major contributions of this thesis are mainly concerned with the work associated with these last two papers.

The associated three dimensional problems have also been studied. And in this respect, O'Rourke [O'Rourke, 1984] gives an $O(n^3)$ algorithm for finding the minimal volume rectangular box enclosing a polyhedron.

1.4 Thesis Terminology and Notation

This part specifies the general notation and defines the terminology used throughout the rest of this work.

Polygon :

In the plane, a *polygon* is defined by the finite set of segments such that the end points of the segments are shared by exactly two edges and no subset of edges has the same property. The segments are known as the *edges* and their extremes as *vertices* of the polygon [Hart and Sharir, 1984].

Simple/Arbitrary Polygon:

If no two edges of a polygon intersect at points other than their vertices, then it is called a *simple* or *arbitrary* polygon. A simple polygon divides the plane into an unbounded *exterior* and a bounded *interior* region. Unless otherwise stated, an n vertex simple polygon is given as a sequence of its vertices $v_0, v_1, v_2, \dots, v_n (= v_0)$. The edges are denoted by $e_1, e_2, e_3, \dots, e_n$, where $e_i = v_{i-1} v_i$, $1 \leq i \leq n$. For simplicity, it is assumed that in the given polygons no three consecutive vertices are collinear.

Reflex vertex:

A vertex in a polygon is *reflex* if the edges adjacent to it make an angle, interior to the polygon, of more than 180° .

Convex Polygon:

A simple polygon with no *reflex* vertex is called a convex polygon.

Containing Edge :

If the projection of every point of a polygon on a line passing through an edge e , lies in e , then e is called a *containing edge*.

Dome Polygon:

A convex polygon one of whose edges is a containing edge.

Semicircular or Half convex Polygon:

A dome polygon whose containing edge is also its longest chord is called a *semicircular polygon* or a *half Convex polygon*

Centrosymmetric Polygon:

The *center* of a polygon is defined to be a point inside the polygon which bisects every chord passing through it. A polygon that has a center is known as a centrosymmetric polygon.

Ratio of Optimality:

If R_{opt} is the optimal object of some type according to a particular optimization criterion, and R is an object similar to R_{opt} , then the ratio of the area of R to the area of R_{opt} will be referred to as the *ratio of optimality of R_{opt}* with respect to area. For instance, if S_{opt} is the maximal area square inscribed in a given polygon P , and there is an algorithm that produces an inscribed square S , in P , such that the ratio of the area of S to S_{opt} is some fraction r , then we say that the ratio of optimality of S or the algorithm, for that matter, is r .

Binding vertex/ Corner:

A binding corner of an inscribed or circumscribed object is the corner that is adjacent to the boundary of the respective polygon.

Fixed Similarity Triangle:

A triangle whose internal angles are fixed (eg, an equilateral triangle).

Fixed Ratio Rectangle :

A rectangle whose ratio of width to length is fixed. (eg, a square).

$|R|$:

Denotes the area of the region or length of the line segment defined by R .

1.5 Major Results

This section discusses the generic outline of the algorithms for solving inclusion and enclosure problems and summarizes the contributions of this thesis.

1.5.1 Verification of the Finiteness Criterion

At first, the nature of most geometric optimization problems like the ones considered in this thesis seem to be continuous as opposed to discrete or combinatorial. Thus, in order to solve such optimization problems, one must first find a *finiteness criterion*. This is comprised of the proof that the optimal solution to the problem is one of a finite number of well defined solutions. See [Chang and Yap, 1986] for a detailed treatment. The bulk of the efforts required focus on such proof, indeed, if one exists. Once the finiteness criterion has been established, the problem is broken down into smaller subproblems each of which can be solved using the familiar tools of Calculus, Linear Programming, etc.

For simpler polygons such as centrosymmetric, dome and semicircular polygons, this thesis gives exact algorithms for solving inclusion and enclosure problems for triangles, squares, rectangles and circles. However, for convex and arbitrary polygons, the algorithms presented are asymptotic in nature. In other words, we give algorithms that solve such problems guaranteeing that the respective ratio of optimality approaches unity, of course, at the expense of increasing computational costs (See Section 1.4, for a definition of *ratio of optimality*).

1.5.2 The Proposed Approach

There are a number of *similarity* parameters associated with geometric objects. A particular set of values for the respective parameters defines a whole class of similar geometric objects. For instance, all rectangles can be characterized by a single parameter, namely, the ratio of width to length. For each possible value of ratio, there is associated a whole class of similar rectangles. In addition to these, there are parameters that specify the orientation or location of an object in the plane. It is usually easier to solve the inclusion and enclosure problems for geometric objects with less similarity or orientation parameters. With this in mind, our approach concentrates on achieving the *best* solution from the solutions of the associated more simpler subproblems which are characterized by lesser similarity and orientation parameters. Such best solution, the *algorithmic* solution, may be the one with the greatest or least area, depending on the objective function of the problem. The final step is to find a relationship between the algorithmic and optimal solution with respect to the objective function that we are optimizing.

To illustrate this concept, consider the following example. Suppose we were to find the maximal area inscribed square in a convex polygon P . We know that the only geometric parameter involved is in connection to the orientation of the maximal area square in P . Thus, for an arbitrary integer k , we choose k distinct directions and for each direction d , we compute the maximal area inscribed square in P , whose side is along direction d . Indeed, the orientation of one of these squares computed may be close to the orientation of the optimal square. Equivalently, the algorithmic solution, the maximum of these k squares, say, S , approximates the optimal one whom we have no knowledge of, and as k tends to infinity, the area of S clearly approaches the area of the optimal square.

With geometric objects with at least one similarity parameter, it is imperative that we find a relationship between the geometry of the polygon and the approximating

object (the inscribing or circumscribing figure). This relationship is necessary for relating the objective function of algorithmic and optimal solutions (area, in our case). We introduce the concept of *ratio* of a polygon as the parameter that contains the geometric information of a polygon. This ratio depends exclusively on the shape, and not just on the size, of the given polygon.

The technique mentioned above will be utilized for finding maximal area squares, rectangles, equilateral and arbitrary triangles inscribed in convex as well as simple polygons. The same method will be used for solving the related enclosure problems as well.

1.5.3 Why Approximation?

The approach outlined in the previous section gives us an approximation to the optimal solution whose accuracy is a direct function of the number of the respective subproblems solved. The asymptotic nature of the algorithms, using this approach, gives us complete control over the accuracy of the desired solutions. Of course, for better and better approximations, increasingly higher number of subproblems need to be solved, and consequently, greater and greater computational costs are incurred. This makes sense because the applications of these problems tend to work in a discrete environment such as a computer screen or a robotic vision. In any case, there exists an underlying *noise factor*, which renders more accurate solutions unprofitable. If this noise factor is known for a specific application, the suitability of our asymptotic algorithms becomes more obvious.

1.5.4 The Model

The algorithms described produce *algorithmic* geometric solutions whose areas approximate the area of the optimal solutions. The degree of accuracy depends on the arbitrarily chosen parameter k , and the geometry of the input convex set. Our algorithms have costs that are proportional to, in addition to the size of the polygon, a polynomial in k . The degree of this polynomial depends entirely on the approximating object. The larger the value of k the better the approximation. In addition, the smaller the value of the ratio of the polygon (corresponding to more elongated polygons) the worse the approximation, and the larger the value of this ratio (corresponding to more circular polygons) the better the degree of accuracy. Equivalently, it will be more costly to obtain a desired degree of approximation for more *elongated* polygons than *circular* ones. We shall find this to be the general theme for asymptotic algorithms for approximating objects with at least one similarity parameter; such as rectangles and arbitrary triangles.

Our algorithms work best for applications where quicker approximate answers are preferred over slower exact ones. This is the case for discrete environments such as Computer screens and Robotic visions.

1.5.5 Major Contributions

The most significant contributions of this thesis are asymptotic algorithms for solving the inclusion and enclosure problems for squares, equilateral triangles, rectangles and triangles for convex and simple polygons. Table 1.1 compares the results in this thesis against the ones found in the literature, where k represents the number of times the respective subproblems need to be solved. In effect, k is proportional to the accuracy of the solution. This proportionality function, the *ratio of optimality*, is given in Tables 3.1.4.1 and 5.1. The complete summary of the results is given in Tables 1.2 and 1.3.

MAJOR RESULTS WITH RESPECT TO THE LITERATURE

Approximating Object	Current Best Algorithms	Results of This Thesis
Inscribed Square in Convex Polygon.	$O(n^2)$ [DePano et. al., 1987a]	$O(k n)$
Inscribed Equilateral Triangle in Convex Polygon.	$O(n^2)$ [DePano et. al., 1987a]	$O(k n)$
Inscribed Equilateral Triangle in Simple Polygon.	$O(n^3)$ [DePano et. al., 1987a]	$O(k n^2 \log n)$
Circumscribed Rectangle in Convex Polygon	$O(n)$ [Toussaint, 1980]	$O(k^2 \log n)$
Inscribed arbitrary Triangle in Simple Polygon	$O(n^4)$ [Melissaratos and Souvaine, 1992]	$O(k^3 n^2 \log n)$

Table 1.1: A comparison of the results of this thesis and those found in the literature.

INCLUSION ALGORITHMS

Regular Convex Polygon	CONVEX POLYGON	ARBITRARY POLYGON
SQUARE	$O(k n)$	$O(k n^2 \log n)$
EQUILATERAL TRIANGLE	$O(k n)$	$O(k n^2 \log n)$
RECTANGLE	$O(k^2 n)$	$O(k^2 n^2 \log n)$
ARBITRARY TRIANGLE	$O(k^3 n)$	$O(k^3 n^2 \log n)$

Table 1.2: Major results of this thesis regarding the inclusion problem

ENCLOSURE ALGORITHMS

APPROXIMATING POLYGON	CONVEX POLYGON
SQUARE	$O(k \log n)$
EQUILATERAL TRIANGLE	$O(k \log n)$
RECTANGLE	$O(k^2 \log n)$

Table 1.3: Major results of this thesis regarding the enclosure problem

1.6 Outline of the Thesis.

This thesis is mainly concerned with asymptotic algorithms for the inclusion and enclosure problems for simple and convex polygons. We focus on area as the objective function for these optimization problems. These problems are also solved, by computing exact solutions, for some special polygons such as semicircular, centrosymmetric and dome polygons. In almost every case, the output consists of the area of the optimal solution and not the solution itself. However, the algorithms presented can be easily modified to output the geometric object forming the optimal solution.

Chapter 2 gives exact algorithms for solving the inclusion problems for semicircular, centrosymmetric and dome polygons. Linear algorithms for computing maximal area right triangle, rectangle, arbitrary triangle and circle for semicircular, and square and circle for centrosymmetric polygons are included in this chapter. Chapter 3 gives asymptotic algorithms for computing maximal area squares, equilateral triangles, rectangles and arbitrary triangles in convex polygons. Chapter 4 extends all these results for the convex polygon to the case of the simple polygons. While Chapter 5 gives algorithms for solving the dual of the problems discussed in Chapters 3 and 4.

Last but not least, the entire thesis is summarized in Chapter 6. Also included are directions for future research and hopes in conjunction to this area.

Chapter 2

INCLUSION PROBLEMS FOR SPECIAL POLYGONS

This chapter presents algorithms for solving the inclusion problems for some special polygons. For the centrosymmetric convex polygon, linear algorithms for finding maximal area square and circle are presented. Linear algorithms are also discussed for solving the inclusion problems for right triangle, rectangle and circle for the semicircular polygon.

2.1 Centrosymmetric Convex Polygons

The *center* of a polygon is defined to be a point inside the polygon which bisects every chord passing through it. A convex polygon that has a center is known as a *centrosymmetric convex* polygon. Since the emphasis is only on convex centrosymmetric polygons, such polygons will be simply referred to as centrosymmetric polygons. This section presents linear time algorithms for finding maximal area circle and square inscribed in centrosymmetric polygons.

Special properties of Centrosymmetric polygons

Let P denote the given Centrosymmetric polygon. If we place P on the Cartesian coordinates, with the center coinciding with the origin, then the boundary of P can be described by two convex functions: say, f in the positive y and g in the negative y . For every point $q(x,y)$ on f , there exists a point $q'(-x,-y)$ on g , which is *symmetric* to it

with respect to the origin. For an edge e on P , we define e' as the edge *symmetric* to e if every point on e has its symmetric counter part on e' , and vice versa. Lemma 2.1.1 gives a useful result about centrosymmetric polygons.

Lemma 2.1.1

The longest chord in a centrosymmetric polygon P , parallel to some direction d , passes through the center of P .

Fig. 2.1.1: A centrosymmetric polygon on the Cartesian coordinates. The longest chord parallel to a given direction always passes through the center of the polygon.

Proof. Suppose that this were not the case. Assume that the chord AB in Fig 2.1.1, parallel to d , is the longest one, and assume $X'Y'$ is parallel to d and passes through the center. We will show that it is impossible for $|X'Y'| < |AB|$. Let A and B be symmetric to X' and B' , respectively. Now, $\angle AOB = \angle A'O'B'$, $|BO| = |B'O|$ and $|AO| = |A'O|$. Thus, $|AB| = |A'B'|$. Consider the parallelogram $ABA'B'$. Since $|X'Y'| < |AB|$ and $|X'Y'| < |A'B'|$, and all three of these are parallel to d , it must be that either X or Y lies inside $ABA'B'$. In

any case, P cannot be a convex polygon. Therefore, by contradiction, $|XY| \geq |AB|$.
Q.E.D.

The result of Lemma 2.1.1 provides useful techniques for proving the finiteness criterion discussed in the introduction of this thesis. The next section utilizes this result to give an algorithm for finding the maximal area inscribed square in a centrosymmetric polygon.

2.1.1 Inscribed Square

This section presents an algorithm that computes in time $O(n)$, the largest area square S_{max} , inside a given n vertex centrosymmetric polygon P . Lemma 2.1.2 given below proves the finiteness criterion for this inclusion problem.

Lemma 2.1.2

The diagonals of S_{max} intersect at the center of P .

Proof. Assume otherwise. See Fig. 2.1.2. Suppose that d_1 and d_2 are the diagonals of S_{max} . Let chords D_1 and D_2 , passing through the center, be parallel to d_1 and d_2 , respectively. Let S be the square whose diagonals have length $\min(|D_1|, |D_2|)$, and these diagonals intersect at O . Assume, without loss of generality that $|D_1| \geq |D_2|$. Due to the symmetry of P , either both corners of S lying on D_1 , lie inside P or both lie outside P . The latter cannot be the case since that will imply that the diagonal of S is longer than D_1 . Since the other pair of opposite corners of S are the end points of D_2 , we can conclude that S is contained in P . According to Lemma 2.1.1, $|D_1| \geq |d_1|$ and $|D_2| \geq |d_2|$, implying that S is at least as large as S_{max} in area. Thus, S_{max} could not have been maximal. By this contradiction it can be concluded that the center of S_{max} coincides with the center of P .
Q.E.D.

Fig. 2.1.2: A Square which does not have its center coinciding with the center O of P , cannot be maximal in area. In the above case, the square with diagonal $\min(D_1, D_2)$ is larger in area.

From Lemma 2.1.2, we know that the diagonals of S_{max} intersect at the center of P . If we knew the location of one corner of S_{max} , say corner A , then the corner opposite to A would be given by its symmetric counter part, A' . The other two corners, say, B & B' can be found by taking the end points of the chord passing through the center, orthogonal to AA' . The area of S_{max} would then be given by $2|OA|^2$. From this relation, it is clear that maximizing $|OA|$ is equivalent to maximizing the area of S_{max} . Consequently, it is enough to search for the location of A among the edges of P . In fact, due to the symmetry, search needs to be conducted in only one of the top or bottom convex functions defining the boundary of P . Thus, the algorithm involves the search for a corner of S_{max} among the edges of P . Let us refer to OA as a *half diagonal*.

Assume a counter clock wise (c.c.w) polar coordinate representation of P . Assume that the top chain of P , given by the radial function $R(\theta)$, is given as a c.c.w. cyclic order of its edges, vertices and angles, $e_1, e_2, e_3, \dots, e_k, v_0, v_1, v_2, \dots, v_k$ and

$\theta_0, \theta_1, \theta_2, \theta_3, \dots, \theta_k$ respectively. Where $e_i = v_{i-1}v_i$ is given by $R(\theta)$ for $\theta \in [\theta_{i-1}, \theta_i]$. The notation $|R(\theta)|$ will be used to denote the length of the portion of ray, that makes an angle of θ with the positive x -axis, from the origin to the boundary of P . Furthermore, define the *orthogonal chain* C_e , with respect to edge e as the chain of edges given by $C_e = R(\theta)$ for all $\theta \in [\theta_{i-1} + \pi/2, \theta_i + \pi/2]$. In other words, ray r , emanating from the origin, intersects e , if and only if, when r is rotated by an angle of $\pi/2$ it intersects some edge in C_e . Note that the orthogonal chain of an edge may be a proper subset of some edge.

Formal Description of the Algorithm

The algorithm searches for a corner of S_{max} on each edge of R . Specifically, for each edge e , it computes C_e , and for each edge h in C_e , it computes the angle interval for which corner A would fall on e and vertex B on h . The maximum half diagonal in this angle interval range provides us with a candidate for the half diagonal of S_{max} . This procedure is repeated for all edges in C_e and for each edge e in R . The maximum of these $O(n)$ half diagonals, say OA , gives the desired half diagonal of S_{max} .

Consider the following subproblem; *Square-Interval*: We are given two edges; $e_i = R(\theta)$ for $\theta_{i-1} \leq \theta \leq \theta_i$ and $e_j = R(\theta)$ for $\theta_{j-1} \leq \theta \leq \theta_j$, $j > i$, where intervals $[\theta_{i-1}, \theta_i]$ and $[\theta_{j-1}, \theta_j]$ are disjoint. The problem is to compute an interval $[\mu_1, \mu_2]$, subset of $[\theta_{i-1}, \theta_i]$, such that the following conditions be simultaneously satisfied.

for every μ in $[\mu_1, \mu_2]$,

i) $R(\mu)$ is on e_i if and only if $R(\mu + \pi/2)$ is on e_j .

ii) $|R(\mu)| \leq |R(\mu + \pi/2)|$

iii) Both, the angle that a ray from the origin to $R(\mu)$ makes with e_i and its supplementary, are no less than $\pi/4$.

The motivation for *Square-Interval* is that the maximum value of $R(\mu)$ for $\mu_1 \leq \mu \leq \mu_2$, provides us with a candidate location for corner A. The condition i) guarantees us a pair of diagonals for S_{max} , ii) makes sure that $R(\mu)$ is the corner A, while iii) assures that a corner of a square can be binding at $R(\mu)$.

How to solve Square-Interval :

The condition i) will be satisfied when we pick e_j among the edges in the orthogonal chain of e_i . If we rotate e_i by a c.c.w. angle of $\pi/2$, and find the intersection of e_i and e_j , the angle interval for which e_i has lesser radial distance than e_j can be easily computed. This angle interval rotated $\pi/2$ clock wise provides us with an interval, say, $[\partial_1, \partial_2]$, that satisfies the condition ii). The procedure given below computes the angle interval $[\partial_1, \partial_2] \supseteq [\mu_1, \mu_2]$, which satisfies condition iii).

Let $X = R(\partial_2)$, $Y = R(\partial_1)$

Let m be the slope of XY

Find points U and V on e_i , s.t. ,

$$|\text{slope of } OU| = |\text{slope of } OV| = (m-1)/(m+1)$$

/ OU & OV make 45 (or 135) degree angles with XY . */*

/ can be done in $O(1)$ time */*

Let OU make an angle of β_U with the positive x-axis (Initial Ray)

Let OV make an angle β_V with the initial ray

$$[\mu_1, \mu_2] \leftarrow [\partial_1, \partial_2] \cap [\beta_U, \beta_V]$$

Given two edge segments e_i and e_j , in constant time, we can find a point A on e_i , such that $|OA|$ is maximum, and qualifies as the binding corner of S_{max} , provided that S_{max} is to have a corner on e_i . Procedure *Find-Binding-diagonal-on-Edge* below finds such A, if it exists.

Procedure Find-Binding-diagonal-on-Edge (e_i, e_j)

From the angle interval for e_i , $[q_{i-1}, q_i]$, compute the interval

$[\mu_1, \mu_2]$ as in Square-Interval problem.

$L_{ij} \leftarrow \text{MAX} \{ R(\mu_1), R(\mu_2) \}$

End .

From the previous work, the complete algorithm to compute S_{max} follows naturally. For each edge e_i in the top chain of P , i.e. radial function R , we compute its orthogonal chain and for each edge e_j in this orthogonal chain, we compute L_{ij} . The maximum of these L_{ij} values provides us with the half diagonal for S_{max} . The computation of the orthogonal chains can be performed using a *scan* type technique as follows. In the cyclic order of edges in R , the orthogonal chain of e_i follows the orthogonal chain of e_{i-1} . Thus, all orthogonal chains can be computed by a scanning algorithm in time $O(n)$. Algorithm 2.2.1 gives the pseudo code description.

Algorithm 2.1.1

INPUT : P

OUTPUT : $|S_{max}|$

- 1) Place P on radial coordinates with its center at the origin and its main diameter on the main axis.
- 2) Let R denote the top convex function of P .
- 3) $C \leftarrow$ orthogonal chain of edge e_1
- 4) For each edge e_i in R do
 - 4.1) For each edge e_j in C do

Compute L_{ij} .

4.2) $C \leftarrow$ orthogonal chain of edge e_{i+1}

Computed by scanning along R , starting from the orthogonal chain of e_i

$|L| \leftarrow \text{Max } \{ L_{ij} : 1 \leq i \leq n, i \geq j \}$

$|S_{\max}| \leftarrow 2 |L|^2.$

Complexity

Steps 1 and 2 are clearly in $O(n)$. Computing orthogonal chains involves ray shooting, i.e., finding the edge intersected by a given ray, and finding this point of intersection. Since the orthogonal chains of consecutive edges are also consecutive in the same order, all orthogonal chains are computed by scanning along the edges. Therefore, each edge is visited a constant number of times, and the algorithm takes an aggregate time of $O(n)$. Thus, the overall time and space complexity is $O(n)$.

2.1.2 Inscribed Circle

This section presents an $O(n)$ time and space algorithm which, given a centrosymmetric polygon P , computes the largest Inscribed circle in P , say, $C_{\max}$. Since the area of a circle is proportional to the square of its perimeter, the algorithm presented effectively solves this inclusion problem by using both the area and the perimeter as the optimization criterion. Lemma 2.1.3 below verifies the discrete nature of the problem.

Lemma 2.1.3

The center of $C_{\max}$ coincides with the center O of the polygon P .

Proof. Assume otherwise. Suppose that the circle C' , of diameter d , shown in Fig. 2.1.3 is the optimal one. Now, according to Lemma 2.1.1, for any chord of C' that

passes through its center, there is at least as long a chord of P passing through O . Thus, there exists an inscribed circle with center at O , whose area is at least as much as that of C' . Hence, the optimal circle must have its center at O . Q.E.D.

Fig. 2.1.3: C' cannot be optimal if it does not have its center at O since, all chords that pass through the origin are greater in length than d .

Description of the Algorithm

Lemma 2.1.3 tells us that C_{max} is centered at O . In order to be certain that C_{max} is contained in P , the radius r_{max} , must be the minimum value of the distance of each edge of P from O . From this observation, the algorithm follows immediately.

Algorithm 2.1.2

INPUT: Centrosymmetric polygon P .

OUTPUT: C_{max}

For each edge e in the top chain of P

$r_e \leftarrow$ Perpendicular distance of e from O .

($r_e \leftarrow \infty$ if no ray from O is perpendicular to e)

$r_{\max} \leftarrow \text{Max} \{ r_e : e \text{ is an edge} \}$

$C_{\max}$ has center at O and radius $r_{\max}$.

Complexity

For each edge, we do $O(1)$ amount of work. There are $O(n)$ edges in the top chain of P , giving a total time and space cost of $O(n)$. ♦

2.2 Semicircular Polygon

The *diameter* of a polygon is the longest chord connecting two points of its perimeter, and a convex polygon with an edge as its diameter is called a *semicircular* polygon. Such an edge in a polygon will be referred to as the *main diameter* of the polygon. Part of the motivation for studying inclusion problems comes from the fact that if a convex polygon is partitioned into two parts by slicing along its diameter, then each of the two parts form semicircular polygons. As Chapter 3 will show, much of the work on inclusion problems for semicircular polygons can be utilized for developing approximation algorithms for the corresponding problems for convex polygons.

This section describes algorithms for solving the inclusion problems on semicircular polygons. Specifically, inclusion algorithms for maximal area right triangle, rectangle, square, and triangle are presented.

Let P be the given semicircular polygon. Throughout this section, it is assumed that OD is the main diameter of P , and that OD lies on the x -axis on the interval $[0, d]$, where d is the length of the main diameter of P . The boundary of P , except the main diameter, will be referred to as the *convex function* of P .

2.2.1 Inscribed Right Triangle

This section presents an $O(n)$ time and space algorithm for computing the maximal area right triangle, say, T_{max} , inscribed within P . Lemma 2.2.1 and Lemma 2.2.2 provide the proof for the finiteness criterion by providing a useful characterization of the solution. The claim is that the hypotenuse of T_{max} coincides with the main diameter.

Lemma 2.2.1

One side of T_{max} coincides with the main diameter of P .

Proof. The proof consists of the following. All possible configurations of T_{max} within the semicircular polygon will be considered separately. And for each configuration, it will be shown that there exists a right triangle whose area is at least as much as T_{max} and that one of its sides lies on the x -axis (main diameter). The proof is divided into three major distinct cases.

Let $T_{max} = \triangle ABC$. If Q is a point in the plane then denote its x and y coordinates as Q_x and Q_y , respectively.

Case 1: $A_y \leq B_y, C_y$.

Let AC make an angle of θ with the x -axis. Let $|BB'|=a$, $|CC'|=b$, $|B'A|=x$ and $|AC'|=y$. Let B' and C' be the orthogonal projections of B and C on the x -axis, respectively. See Fig. 2.2.1. We shall prove that $|\triangle B'CC'| \geq |\triangle ABC|$.

Fig. 2.2.1: $\triangle ABC$ is no more than $\triangle B'CC'$ in area.

Without loss of generality, assume that $a \leq b$. From Fig. 2.2.1,

$$\tan \theta = \frac{x}{a} = \frac{b}{y}$$

$$\therefore xy = ab \dots\dots\dots (*)$$

$$\begin{aligned} |\triangle ABC| &= \frac{1}{2} |AB| |AC| \\ &= \frac{1}{2} \left(\frac{x}{\sin \theta} \right) \left(\frac{b}{\sin \theta} \right) \end{aligned}$$

$$\begin{aligned} |\Delta B'CC'| &= \frac{1}{2} |C'C| |BC'| \\ &= \frac{1}{2} b(x+y) \end{aligned}$$

Thus, it is enough to show that:

$$\frac{x}{\sin^2 \theta} \leq (x+y)$$

Now, from (*)

$$\begin{aligned} xy &= ab, \quad b \geq a \\ \Leftrightarrow xy &\leq b^2 \\ \Leftrightarrow \frac{x}{y} &\leq \frac{b^2}{y^2} \\ \Leftrightarrow \frac{x}{y} &\leq \tan^2 \theta \\ \Leftrightarrow \frac{x}{y} &\leq \frac{\sin^2 \theta}{\cos^2 \theta} \\ \Leftrightarrow x \cos^2 \theta &\leq y \sin^2 \theta \\ \Leftrightarrow x - x \sin^2 \theta &\leq y \sin^2 \theta \\ \Leftrightarrow x &\leq (x+y) \sin^2 \theta \\ \Leftrightarrow \frac{x}{\sin^2 \theta} &\leq (x+y) \end{aligned}$$

Hence, $|\Delta B'CC'| \geq |\Delta ABC|$. Q.E.D.

Case 2: $A_y \geq B_y, C_y$.

T_{max} in this case may have two possible configurations.

Case 2(a) $C_x \leq A_x \leq B_x$ (C on the x -axis)

If $|\Delta ABC| = |T_{max}|$ then the following steps transform it into a triangle that has its hypotenuse on the main diameter, namely, $\Delta A'B'O$ in Fig. 2.2.2:

- i) From point A , rotate AC until it lies on AO .
- ii) Slide AC along AO until C coincides with point O .
- iii) From O , rotate OB until it lies on the main diameter.

Fig. 2.2.2: Case 2a. $|\Delta A'B'O|$ is equal to $|\Delta ABC|$ and has one of its sides on the main diameter.

Since $|OD| \geq |CB|$, $\Delta A'B'O$ will lie in the region defined by $OABD$ and therefore, will lie in P as well. Incidentally, if C coincided with O then we can ignore transformation Steps *i*) and *ii*).

Fig. 2.2.3: $|\Delta A'B'C| = |\Delta ABC|$, $\Delta A'B'C$ is contained in the polygon and has one side on the x -axis.

Case 2(b) $C_x \geq A_x \geq B_x$ (B on the x -axis):

It is easy to show that in this case only the following two configurations are possible.

(i) B coincides with the origin while C is binding.

Since $|BD| \geq |BC|$, then, from B , we rotate ΔABC towards the x -axis until BC lies on the x -axis. See Fig. 2.2.3. Clearly, this rotated version of ΔABC , i.e. $\Delta A'C'O$,

lies within the semicircular polygon defined by $BACD$. Thus, the above transformation places one side of T_{max} on the x -axis.

(ii) B lies to the right of O while A and C may both be binding.

Fig. 2.2.4: $\triangle A'B'C'$ is contained in the region $OACD$ and has area equal to $\triangle ABC$, and is obtained by rotating AB towards AO at A then translating down to the x -axis.

Consider the following transformation of $\triangle ABC$ into a triangle with the same area and whose hypotenuse lies on the x -axis. From A , we rotate AB towards AO until either AB coincides with AO or BC becomes parallel with the x -axis. See Fig. 2.2.4. If the latter happens first then we simply translate the triangle down to the x -axis. On the other hand, if AB coincides with AO first, we slide AB along AO until B falls at O , at which point we are dealing with the already proved case 2b(i).

Case 3: $B_y \leq A_y \leq C_y$

Three configurations for the triangle must be considered in this case.

Case 3(a) $C_x \geq B_x, A_x$

Let F be the intersection between a line passing through A_xA and a horizontal line passing through C . Furthermore, let $\angle ACF = \theta$, $|AA_x| = a$, $|CC_x| = b$, $|A_xB| = x$ and $|BC_x| = y$. See Fig. 2.2.5.

Fig. 2.2.5: Case 3a: $|\Delta A_x C C_x| \geq |\Delta ABC| = T_{max}$.

We will show that $|\Delta A_x C C_x| \geq |\Delta ABC|$. From elementary geometry, $\angle A_x A B = \theta$, implying that $\Delta A_x A B$ is similar to $\Delta A C F$. Thus,

$$\tan \theta = \frac{x}{a} = \frac{b-a}{x+y}$$

$$b-a = \frac{(x+y)x}{a} \dots\dots\dots (*)$$

Suppose that $|\Delta A_x C C_x| < |\Delta ABC|$, which implies,

$$\Leftrightarrow b(x+y) < |AB| \times |AC| = \sqrt{a^2 + x^2} \sqrt{(b-a)^2 + (x+y)^2}$$

and substituting (*) into this gives,

$$b(x+y) < \sqrt{a^2 + x^2} \sqrt{(x+y)^2 \left(\frac{x}{a}\right)^2 + (x+y)^2}$$

$$\Leftrightarrow b < \sqrt{a^2 + x^2} \sqrt{\frac{x^2}{a^2} + 1} = \frac{a^2 + x^2}{a}$$

$$\Leftrightarrow ab < a^2 + x^2 \Leftrightarrow a(a+b-a) < a^2 + x^2$$

$$\Leftrightarrow a(b-a) < x^2 \Leftrightarrow \frac{b-a}{x} < \frac{x}{a}$$

but from (*),

$$\Leftrightarrow \frac{x}{a} = \frac{b-a}{x+y}$$

$$\Leftrightarrow \frac{b-a}{x} < \frac{b-a}{x+y}$$

$$\Leftrightarrow x > x+y$$

which is clearly false since by our assumption $y > 0$. Thus, by contradiction, it must be true that $|\Delta A_1 C_1| \geq |\Delta ABC|$. Hence proved.

Case 3(b) $A_x \geq B_x, C_x$ (or $A_x \leq B_x, C_x$)

Fig 2.2.6: Case 3b: If C is at the origin, then $|\Delta A'B'C| = |\Delta ABC|$. If not, then we rotate BC until it falls on BO and slide C down to the origin.

Let $|\Delta ABC|$ be as shown in Fig. 2.2.6. This case is very similar to case (2b). We show that there exist simple transformations that can bring one side of ΔABC down to the x -axis.

(i) If C lies on the origin then reflecting ΔABC through CA , followed by, from C , a rotation until BC lies on the x -axis, gives us triangle $\Delta A'B'C$ that lies within the convex region OBD and has its hypotenuse on the x -axis.

(ii) If on the other hand C lies to the right of the origin, then as in case (2b), from B , we can rotate CB until either it coincides with the line BO , or AC becomes parallel to the x -axis. If the former happens first, we can simply translate the triangle vertically down to the x -axis. However, if CB coincides with BO first, then we can slide CB along BO until C lies on the origin, at which point we have situation label as (i) in this case.

Case 3(c) $C_X \leq B_X$.

(i) B lies at D .

Let $\angle ABC = \theta$ and $\angle OBA = \alpha$. See Fig. 2.2.7. Let $\triangle A'BC'$ be obtained from $\triangle ABC$ by reflecting it through a line that bisects $\angle C'BC$. Since $|OD| \geq |BC|$, it is clear that $\triangle A'BC'$ has one of its sides on the x -axis and is equal in area to $\triangle ABC$.

Fig. 2.2.7: Case 3c (i). $\triangle ABC$ can be transformed to $\triangle A'BC'$ by a simple reflection, such that $|\triangle ABC| = |\triangle A'BC'|$ and $\triangle A'BC'$ has one of its sides on the x -axis.

(ii) B lies to the left of D .

In this situation, from C , we can rotate CB towards CD until either CA becomes perpendicular to the x -axis or CB coincides with CD . If the former happens, then we can simply translate the whole triangle down to the x -axis. However, in the latter case, we can slide $\triangle ABC$ along CD until B lies on D . At which point case 3c(i) applies. See above. Q.E.D.

Lemma 2.2.2

T_{max} has its hypotenuse coinciding with the x -axis.

Proof: In Lemma 2.2.1, we showed that T_{max} has one of its sides on the x -axis. Thus, we will show that if T_{max} has sides AB or AC on the x -axis then there exists a triangle T' , with $|T'| \geq |T_{max}|$, and whose hypotenuse lies on the x -axis. Because both of these cases are very similar, we shall only prove one of these. See Fig. 2.2.8.

Fig. 2.2.8: If C is at point D , then $|\Delta A'B'C| = |\Delta ABC|$. Otherwise, a translation followed by reflection places ΔABC 's hypotenuse on the x -axis.

Let BD make an angle of α with the x -axis as shown in Fig. 2.2.8. If C coincides with D , then we can take T' as the reflection of ΔABC through a line L which bisects α . This T' is $\Delta A'B'C$ in Fig. 2.2.8. Since the length of the main diameter is larger than any chord in P , $|OD| \geq |BC|$. Thus, it is easy to see that T' lies below DBO and above the x -axis.

If C lies to the left of D , then before we perform the above reflection, from B , we rotate BC towards BD until it coincides with BD . Now, the reflection through L places ΔABC 's hypotenuse on the x -axis. Q.E.D.

Let F be the convex function of P . Armed with the results in Lemma 2.2.1 and Lemma 2.2.2, it is sufficient to search for T_{max} among the set of all right triangles that have their hypotenuse on the x -axis, one corner on F and one corner at the origin (or at D). Given an edge e of F , denote T_e as the largest area right triangle with one corner on e ,

hypotenuse on the x -axis and one corner at the origin. Let e be defined by : $y = mx + b$, $x_1 \leq x \leq x_2$, $b \geq 0$.

Consider $\triangle ABC$ in Fig. 2.2.9. For any arbitrary t , $0 \leq t \leq d$, let $T(t)$ be the triangle with its hypotenuse lying on the x -axis and the opposite corner adjacent to F . We can get the definition of the area of $T(t)$ as follows. Let A be at the point $(t, mt+b)$. From this, CA can be defined by $y = x(mt+b)/t$, $0 \leq x \leq t$.

Fig. 2.2.9 : Depiction of triangle $T(t)$ adjacent on edge e of P

If we knew $|CB|$ then we could calculate $|T(t)|$. From the fact that the product of the slopes of AB and CA equals -1 , and that $(t, mt+b)$ lies on A , we can obtain the equation of AB as follows:

$$y = -\frac{t}{mt+b}x + \frac{(mt+b)^2 + t}{(mt+b)}$$

The x -intercept of this equation, say, x_m , gives us the coordinates of B .

$$x_m = \frac{(mt+b)^2 + t^2}{t}$$

Thus, $T(t)$ has height $mt+b$ and base x_m .

$$|T(t)| = \frac{1}{2}(mt+b)\left(\frac{(mt+b)^2 + t^2}{t}\right)$$

However, since $T(t)$ must lie within the convex set, we require that $x_m \leq d$, i.e.,

$$d \geq \frac{(mt+b)^2 + t^2}{t}$$

$$\Leftrightarrow (mt+b)^2 + t^2 - td \leq 0$$

$$\Leftrightarrow (m^2 + 1)t^2 + (2mb - d)t + b^2 \leq 0$$

completing the square,

$$\Leftrightarrow t \leq \frac{d - 2mb \pm \sqrt{d^2 - 4mbd - 4b^2}}{2(m^2 + 1)}$$

We desire the maximum values of $T(t)$ within the domains of each edge of F . Using techniques from calculus,

$$\frac{dT(t)}{dt} = \frac{1}{2} \left\{ \frac{3mt(mt + b)^2 - (mt + b)^3}{t^2} + 2mt + b \right\}$$

$$\frac{dT(t)}{dt} = 0$$

$$\Leftrightarrow 3mt(mt + b)^2 - (mt + b)^3 + 2mt^3 + bt^2 = 0$$

$$\Leftrightarrow 2m(m^2 + 1)t^3 + b(1 + 3m^2)t^2 - b^3 = 0$$

The solutions of this equation provide possible candidates for t such that $T_e = T(t)$. The solutions of such a polynomial can be calculated in $O(1)$. Given below is the pseudo code description of the procedure for finding $|T_e|$.

Algorithm

INF : Edge e defined by $y=mx+b$, $x_1 \leq x \leq x_2$.

OUTPUT: $|T_e|$

$$[u_1, u_2] \leftarrow [x_1, x_2] \cap \left[\frac{d - 2md + \sqrt{d^2 - 4mbd - 4b^2}}{2(m^2 + 1)}, \frac{d - 2md - \sqrt{d^2 - 4mbd - 4b^2}}{2(m^2 + 1)} \right]$$

If $m \geq 0$ Then /* rising edge */

```

 $|T_e| \leftarrow |T(u_2)|$ 
Else          /* Falling edge. */
 $|T_e| \leftarrow \text{Max } \{ |T(u_1)|, |T(u_2)|, |T(t)| : \text{for all } t \text{ in } [u_1, u_2], \text{ such that}$ 

$$2m(m+1)t^3 + b(1+3m^2)t^2 - b^3 = 0 \}$$

Endif

```

The complete algorithm for computing T_{max} is as follows:

Algorithm 2.2.1

INPUT: F , the boundary of Semicircular polygon P .

OUTPUT: $|T_{max}|$.

```

1- For each edge  $e$  in  $F$ 
    Compute  $T_e$ .
2- Reflect  $F$  by the line  $x = d/2$  and repeat Step 1
 $|T_{max}| \leftarrow \text{MAX } \{ |T_e| : e \text{ is an edge of } P \}$ 

```

Complexity

The procedure for finding a single T_e can be done in constant time, and the algorithm repeats such procedure a total of at most $2n$ times. Thus, the overall algorithm is in $O(n)$.

2.2.2 Inscribed Rectangle

In this section, we present an $O(n)$ algorithm for finding the maximal area inscribed rectangle, say, R_{max} . The usual proof for the problem's finiteness criterion is given below.

Lemma 2.2.3

R_{max} has one of its sides coinciding with the main diameter of P .

Proof:

It will be proved that if R_{max} does not have one of its sides on the main diameter, then we can always find a rectangle equal in area with R_{max} that does have this property. Assume that rectangle $ABCD$ in Fig. 2.2.10 is the maximal area inscribed rectangle in P .

Fig. 2.2.10: Rectangle $ABCD$ has the same area as rectangle $AA'F'F$.

See Fig. 2.2.10. Let F be the horizontal projection of A on BC (if this projection falls on DC , then we would take F as the projection of C on AB , and consider the corresponding rectangle). Let the projection of A and F on the main diameter be A' and F' , respectively. We shall show that area of $AFF'A'$ is exactly same as the area of $ABCD$. Since angle $A'AD$ is equal to angle FAB , and $\angle AA'D = \angle ABF = 90$ degrees, it is easy to see that $\triangle AA'D$ is similar to $\triangle ABF$. From this similarity,

$$\frac{|AB|}{|AA'|} = \frac{|AF|}{|AD|}$$

$$|AB||AD| = |AA'||AF|$$

Thus, $|ABCD| = |AA'F'F|$.

Q.E.D.

From the result in Lemma 2.2.3, we now know that R_{max} is among the set of rectangles that have one side lying on the x -axis. If the convex function of P is denoted by H , then it can be easily shown that the top two corners of R_{max} are adjacent to H . The arguments are as follows. Assume this were not the case. If neither of the top two corners of R_{max} lie on H , then a vertical magnification of R_{max} increases its area. Next, suppose that only one corner of R_{max} is adjacent to H . Well, now we can move R_{max} horizontally and obtain a rectangle neither of whose corners touch H . Thus, by contradiction, R_{max} must have both of its top two corners on H . Therefore, the search for R_{max} must be conducted among all rectangles inside P which have two corners on the x -axis and two on H .

Consider the following subproblem of finding the maximal area rectangle R_0 , within a trapezoid T . Assume that T lies on the x -axis and has the non-parallel sides defined by $f(x)=m_1x+b_1, u_1 \leq x \leq v_1$ and $g(x)=m_2x+b_2, u_2 \leq x \leq v_2$. Let $R(x)$ be a rectangle lying on the x -axis with its other two corners on H . Let $R(x)$ have width t and height $f(x)$ for some x in $[u_1, v_1]$. The corners of $R(x)$ would then be,

$$\{ x, x+t, g(x+t), f(x) \}, \text{ where } g(x+t) = f(x).$$

Thus, we can find t and x for R_0 by solving the optimization problem given by:

$$\begin{aligned} & \text{MAX } t f(x) \\ & \text{s.t.} \\ & f(x) = g(x+t), \quad 0 \leq t \leq d, \quad u_1 \leq x \leq v_1 \end{aligned}$$

From the above condition, we can get t in terms of x :

$$m_1x + b_1 = m_2(x+t) + b_2$$

$$t = \frac{(m_1 - m_2)x + b_1 - b_2}{m_2}$$

we need to find the maximum value of $|R(x)|$ for x in $[v_1, u_2]$

$$|R(x)| = (m_1 x + b_1) \left\{ \frac{(m_1 - m_2)x + b_1 - b_2}{m_2} \right\}$$

Using calculus,

$$\frac{d|R(x)|}{dx} = \frac{1}{m_2} \{ 2m_1(m_1 - m_2)x + (2m_1b_2 - m_1b_2 - m_2b_1) \}$$

and we are interested in the values of x for which the above quantity disappears.

$$\begin{aligned} \frac{d|R(x)|}{dx} &= 0 \\ \Leftrightarrow x_m = x &= \frac{m_1b_2 + m_2b_1 - 2m_1b_1}{2m_1(m_1 - m_2)} \end{aligned}$$

Thus, $|R(x)|$ attains its maximum value at either x_m or at the end points v_1 or u_2 .

$$|R_0| = \text{MAX} \{ |R(x_m)|, |R(u_1)|, |R(u_2)| \}$$

Description of the algorithm

The algorithm consists of partitioning P into a set of trapezoids. For each trapezoid in P , the procedure for solving the previous subproblem is used to find the maximal area rectangle whose top corners lie on the edge of the trapezoid. The maximum of the resulting rectangles is taken as R_{max} .

Partitioning of P into trapezoids:

Consider the following partition. Draw horizontal lines through each vertex in H . This will create at most $n-3$ trapezoids and one triangle at the top. Denote by T_{ij} the

trapezoid bounded by edge e_i on the left, e_j on the right and by the horizontal lines directly above and below. And denote by R_{ij} the rectangle with maximal area that has one side on the x -axis, left vertex on e_i and right vertex on e_j .

Algorithm: 2.2.2

INPUT: H , the Convex function of Semicircular polygon P .

OUTPUT: R_{max} , the maximal area rectangle that can be inscribed in P .

1) Partition P into the $O(n)$ trapezoids.

2) For each trapezoid T_{ij}

compute R_{ij} .

$|R_{max}| = \text{MAX} \{ |R_{ij}| : 1 \leq i, j \leq n \}$.

Complexity

The solution of the optimization problem for finding R_{ij} takes $O(1)$ time. This process is repeated for $O(n)$ trapezoids, giving us an overall complexity of $O(n)$ time and space for the algorithm.

2.2.3 Inscribed Triangle

This section describes an $O(\log n)$ algorithm for finding the maximal area inscribed triangle, say, T_{max} , in P . The assumption is that the vertices of the polygon have already been input in a cyclic order. As before, let H be the convex function of P .

Lemma 2.2.4

One side of T_{max} is the main diameter of P and its opposite corner coincides with the vertex of H that has maximum vertical distance from the main diameter .

Proof: The proof will be presented by way of contradiction. Suppose T_{max} does not lie on the main diameter. Assume that $\triangle ABC$ in Fig. 2.2.11 is T_{max} . Observe that a line parallel to AB is farthest from AB when it is a tangent to H . As a consequence of this fact, $\triangle ABD$ is greater or equal in height than $\triangle ABC$, and since they both share the same base, $\triangle ABD$ has area at least as much as $\triangle ABC$. Similarly, $\triangle OBD$ has area at least as much as $\triangle ABD$. Therefore, T_{max} has the main diameter as its one side.

By the same argument, any inscribed triangle with base at the main diameter has maximum area when its third corner attains greatest height. A point, other than the extremes, of an edge of H cannot have the greatest height since we can obtain greater heights by shifting the point towards one of the vertices to left or right. Thus, the opposite corner of T_{max} must coincide with the highest vertex of H , and has its base as the main diameter. Q.E.D.

Fig. 2.2.11: The largest area triangle $\triangle OBD$, must have the main diameter as one its sides.

According to Lemma 2.2.4, the only work needed to do is to find a vertex w of H which has the largest y coordinate. Starting from O (see Fig. 2.2.11), height of the

points on the boundary of H monotonically increase until w , after which it monotonically decreases. Since we are given the vertices of H in a cyclic order, binary search over the heights of these vertices will return w in time $O(\log n)$. Thus, the entire algorithm runs in time $O(\log n)$.

2.2.4 Inscribed Circle

Denote by C_{max} , the maximal area circle that can be inscribed within our given semicircular polygon P . This section presents an algorithm that computes C_{max} in time $O(n)$. Lemma 2.2.5 below, gives the proof for the finiteness criterion for this problem.

Lemma: 2.2.5

C_{max} is adjacent on at least three points on the boundary of P , one of which must be on the main diameter (or the x -axis).

Fig. 2.2.12: Figures depicting why a maximal area inscribed circle must be adjacent to at least three points on boundary of P , one of which must be on the x -axis. In each of the situations shown in (a), (b) and (c), simple transformations exist which increase the area of C_{max} . C_{max} must have the configuration shown in (d).

Proof. Suppose C_{max} is not adjacent to the x -axis. From the convexity of P , we can simply translate C_{max} vertically down until it becomes tangent to the x -axis. Now, if C_{max} is not tangent to the convex function of P , then we simply increase its radius until it does

so. Finally, with C_{max} touching P at the above two points, we can translate it horizontally left or right while increasing its radius, until it becomes adjacent on a third point of P . Thus, C_{max} lies on the x -axis and is adjacent to at least two points on boundary of P . See Fig. 2.2.12. Q.E.D.

Given an edge e , define the middle segment of e w.r.t. the x -axis, denoted by $m(e)$, as the locus of points p , such that the perpendicular distance of p from e is the same as its distance from the x -axis, and that p lies below e . It can be easily shown that if a circle is to be tangent to both the x -axis and e , then it must have its center lie on $m(e)$ and its radius will be the height of this center. As a consequence of Lemma 2.2.5 and the above fact, C_{max} has its center on the middle segment of some edge of P .

How to compute middle segments

Given an edge e , consider the line containing e , say, L . The line M , which bisects the angle that L makes with the x -axis, is the set of points equidistant from L and the x -axis. Thus, $m(e)$ lies on M . Now, take two parallel lines perpendicular to L and passing through the end points of e . The intersections of these two parallel lines and M form the end points of $m(e)$. Therefore, in $O(1)$ time we can compute the middle segment of an edge w.r.t. the x -axis.

Lemma 2.2.6

Let H be the semicircular polygon defined by the lower envelope L , of the set of middle segments $\{m(e) : e \in P\}$, and the main diameter of P . The center of C_{max} coincides with the vertex of H that has the greatest distance from the x -axis.

Proof. Let c be the center of C_{max} . First, we will show that c lies on L . Suppose otherwise. That is, assume that c lies above L . From the definition of L , it can be easily

seen that C_{max} cannot be tangent to x -axis. According to Lemma 2.2.6, this implies that C_{max} cannot possibly be maximal. Similarly, it is impossible for c to lie below L .

Thus, from every point q , on the chain L , one can construct a circle with radius as the perpendicular distance of q from the x -axis, and that is tangent to the x -axis and the convex function of P . It is clear from this that finding C_{max} is equivalent to finding the vertex of L with the greatest height above the x -axis. Q.E.D

Formal Description

Using the results in Lemma 2.2.5 and 2.2.6, the general form of the algorithm is as follows. We compute the set M of middle segments of all edges of P , compute the lower envelope L of these segments and find the vertex v on this envelope with the greatest distance from the x -axis. C_{max} has its center on v .

The major operation in the algorithm is that of finding the lower envelope of the middle segments. Hart and Sharir [16] have shown that this can be done in $O(n A^{-1}(n))$, where $A^{-1}(n)$ is the extremely slow growing inverse of

Ackermann's function. Their algorithm handles the case of arbitrary configuration of the given lines. However, in our case, the middle segments of the edges are not quite arbitrary. Indeed, it is worth noting that the middle segments of two consecutive edges do in fact intersect. Of course, the intersection may not be on the lower envelope. Based on this observation, the following algorithm computes the middle segments of the edges of P and their lower envelope L . A stack S is used to store the vertices of L . Denote by S_k as the k^{th} element, from the top, in the stack S (S_0 will denote the top element).

Algorithm: 2.2.3

Compute lower envelope of the middle segments of the edges of P .

INPUT: P

OUTPUT: L

Compute m_1

Push first end point of m_1 on S

Push second end point of m_1 on S

$i \leftarrow 2$

Repeat

Compute m_i

If m_i intersects with line segment $S_{i-1}S_0$ Then

$x \leftarrow$ intersection of m_i and $S_{i-1}S_0$

Push x on S

increment i

Else

Pop S

Until $i \geq n$

$L \leftarrow$ chain of vertices from S

The complete algorithm for computing C_{max} is as follows:

Algorithm: 2.2.4

Compute C_{max}

INPUT: P

OUTPUT: C_{max}

1- $L \leftarrow$ Lower envelope of the middle segments of the edges of P .

2- $v \leftarrow$ vertex of L with maximum y-coordinate.

center of $C_{max} \leftarrow v$

radius of $C_{max} \leftarrow y\text{-coordinate of } v.$

Complexity

For Step 1 (see Algorithm 2.2.3), during each execution of the *repeat* loop, either the index i is incremented or an end point of some middle segment is popped off the stack never to be considered again. In other words, if an amortized cost of 2 or 3 is assumed for each edge of the polygon, the overall cost of finding the lower envelope is in $O(n)$.

Step 2 can be done in $O(\log n)$ time by employing binary search over the heights. As a result, the overall time complexity is linear.

2.3 Dome Polygons

If the projection of every point of a polygon on a line passing through an edge e , lies in e , then e is called a *containing edge*. A convex polygon that contains a containing edge is known as a *dome convex* polygon, or for our purposes, just *dome* polygon. Observe that a dome polygon with the restriction that the longest chord in it be no longer than its containing edge is none other than the semicircular polygon. Inclusion algorithms on semicircular polygons presented in Section 2.2 that did not utilize this constraint in a semicircular polygon can be effectively extended to dome polygons. For instance, the inclusion algorithm for finding the maximal area circle can be equally well applied to the dome polygons as well. Similarly, the results for inscribed square, rectangle and triangle can also be extended to dome polygons.

Chapter Summary

This chapter presented algorithms for maximal area inscribed squares, circles, rectangles and triangles in some special polygons such as centrosymmetric, semicircular and dome polygons. The general techniques used for these algorithms can be utilized for solving other related inclusion as well as enclosure problems as well. For instance, the techniques used for the algorithm for finding maximal inscribed rectangles and squares can be easily extended to the case of finding minimal area circumscribed rectangles and squares on semicircular and dome polygons. We hope that such techniques can be employed for producing algorithms for solving such types of problems on other types of special polygons as well. Table 2.1 below summarizes the results of this chapter .

INCLUSION ALGORITHMS FOR SPECIAL POLYGONS

Regular Convex	Centrosymmetric	Semicircular	Dome
RECTANGLE	---	$O(n)$	$O(n)$
SQUARE	$O(n)$	---	---
TRIANGLE	---	$O(\log n)$	$O(\log n)$
RIGHT TRIANGLE	---	$O(n)$	---
CIRCLE	$O(n)$	$O(n)$	$O(n)$

Table 2.1: Summary of the results for the inclusion problem for special polygons.

Chapter 3

INCLUSION PROBLEMS FOR CONVEX POLYGONS

This chapter presents asymptotic algorithms for the inclusion and enclosure problems on Convex polygons. We give algorithms that produce inscribed Square, Rectangle, Equilateral Triangle and arbitrary Triangle within the given convex polygon, such that the ratio of optimality of these regular convex polygons asymptotically approach unity. For the case of the squares and equilateral triangles, the best known algorithm of DePano *et al.* [DePano et. al., 1987a] is improved from $O(n^2)$ to $O(kn)$ where k is an arbitrary parameter which effects the ratio of optimality of the respective algorithms.

The chapter begins by discussing some relatively simple techniques for solving the inclusion problems for triangles and 4-gons in convex polygons, by giving algorithms with fixed ratio of optimality. The finiteness criterion follows from the following theorem that given a convex polygon P , the vertex set of maximal area inscribing k -gon Q , is a subset of the vertex set of P . For a proof of this result, see Dobkin and Snyder [Dobkin and Snyder, 1979] and Boyce *et al.* [Boyce et. al., 1985]. Throughout this chapter, we shall denote the given convex polygon by P .

3.1 Suboptimal Inscribed Triangle

This section presents two algorithms for finding suboptimal area triangles in a convex polygon, with ratio of optimality $1/2$ and $1/3$ at a cost of $O(n)$ and $O(n \log n)$, respectively. Even though, Dobkin and Snyder [Dobkin and Snyder, 1979] have a linear

time algorithm for finding the maximal area triangle in a convex polygon, the emphasis here is on the simplicity of ideas discussed. The philosophy is that simple techniques such as the ones presented in this section may possibly be generalized for developing more efficient approximation algorithms for the inclusion and enclosure problems. The algorithms presented utilize the corresponding work done on semicircular polygons (Section 2.2). Let the maximal area inscribed triangle in P be denoted by T_{max} .

3.1.1 Algorithm with Optimality Ratio of $1/3$.

Consider the following simple $O(n)$ algorithm that computes triangle T_a which has area at least $1/3$ of the area of T_{max} .

Algorithm 3.1

INPUT : P

OUTPUT : T_a .

- 1- Compute diameter D of P
 - 2- Slice P along D into two semicircular polygons; H_1 and H_2 .
 - 3- Compute maximal area triangle T_1 inscribed in H_1
 - 4- Compute maximal area triangle T_2 inscribed in H_2 .
- $|T_a| \leftarrow \text{Max } \{|T_1|, |T_2|\}$

Lemma 3.1.1

If T_a is the outcome of Algorithm 3.1 then,

$$|T_a| \geq 1/3 |T_{max}|$$

Proof. If all three corners of T_{max} lie on H_1 or H_2 , then Algorithm 3.1 will return a T_a which would have the same area as T_{max} , completing the proof for us. Thus, we will

consider the case where T_{max} has one corner in one and the other two corners in the second semicircular polygon. See Fig. 3.1, where ΔABC represents T_{max} , and without loss of generality, assume that the top half of P is H_1 and the bottom half is H_2 . Let X and Y be the intersections of the diameter with the boundary of ΔABC .

From Fig. 3.1,

$$|T_{max}| = |\Delta AXB| + |\Delta XYB| + |\Delta XYC|$$

but, $|\Delta AXB| \leq |T_1|$, $|\Delta XYB| \leq |T_1|$ and $|\Delta XYC| \leq |T_2|$,

$$|T_{max}| \leq |T_1| + |T_1| + |T_2|$$

from the algorithm, $|T_1|, |T_2| \leq |T_a|$, implying that

$$|T_{max}| \leq 3 |T_a| \text{ or}$$

$$|T_a| \geq 1/3 |T_{max}|$$

Q. E. D.

Fig. 3.1: Possible configuration of T_{max} with relation to the diameter of the convex polygon.

Complexity

The diameter of a convex polygon can be found in linear time (see [Preparata and Shamos, 1990], Section 4.2). Step 2 is clearly $O(n)$. Steps 3 and 4 have costs $O(\log n)$ using the work in Section 2.2.4. Thus, the above algorithm has optimality ratio of $1/3$ and space and time costs of $O(n)$. ♦

3.1.2 Algorithm with Optimality Ratio of $1/2$.

The optimality ratio of the previous algorithm of Section 3.1.1 can be improved to $1/2$ by incurring an increase in cost by a factor of $O(\log n)$. This section will show that there exists an algorithm which computes a triangle T_a , which has area at least $1/2$ of the area of T_{max} . As a motivation for this problem, first, we discuss the solution to the subproblem, *binding maximal area triangle*.

Binding maximal area triangle Problem

Given a convex polygon P and one of its vertices v , find the maximal area inscribed triangle T_v , one of whose corners coincides with v .

It has been demonstrated that the corners of any maximal area inscribed triangle must be a subset of the vertices of the convex polygon it is inscribed in [Dobkin and Snyder, 1979]; Boyce *et. al.*, 1985]. Thus, a naive algorithm for this problem is the one that considers all possible triangles with one corner as v . Essentially, each distinct pair of vertices among vertices other than v , together with the vertex v , produces a triangle. Therefore, since there are $O(n^2)$ pairs of vertices, this algorithm has apparently a running time of $O(n^2)$. However, there is a way to improve this complexity. The important observation is that, if we were given two corners A and B of a triangle, we can maximize the area by choosing, among the vertices of P , the vertex which has maximum distance from chord AB . Such idea has been used for finding T_{max} for a semicircular polygon in Section 2.2.4. The reader will recall that the third corner of the maximal area triangle can be found using binary search on the distances of the vertices of P from chord AB . Equivalently, for each corner B of T_v , we do $O(\log n)$ work to find the third corner. Thus, a binding maximal area triangle can be found in $O(n \log n)$ time. We can now formalize the solution to this subproblem

Algorithm 3.2**INPUT :** P, v **OUTPUT:** T_v For each vertex $w, w \neq v$, do1- $u \leftarrow$ vertex that has maximum distance from chord vw .2- Let $T_{vw} \leftarrow \Delta uvw$ $|T_v| \leftarrow \text{MAX } \{|T_{vw}| : w \text{ is a vertex } \neq v\}$ **Complexity**

The *FOR* loop is executed $n-1$ times and each of its executions do $O(\log n)$ amount of work for binary searching, giving a total cost of $O(n \log n)$.

Now, going back to the problem at hand, the result of the *binding-maximal-area triangle* problem will be utilized as follows.

Algorithm: 3.3**INPUT :** P **OUTPUT :** T_a .1- Compute diameter AB of P 2- Slice P along AB into two semicircular polygons; H_1 and H_2 .3- Compute maximal area triangle T_1 inscribed in H_1 4- Compute maximal area triangle T_2 inscribed in H_2 .

5- Solve "binding maximal area triangle" problem and compute

 T_A and T_B . $|T_a| \leftarrow \text{Max } \{|T_1|, |T_2|, |T_A|, |T_B|\}$

Complexity

As demonstrated above, finding binding vertex maximal area triangles T_A and T_B can be accomplished in $O(n \log n)$ time which is the dominating cost of the whole algorithm. All other operations are $O(n)$, as has been shown previously. Hence, Algorithm 3.2 has time cost in $O(n \log n)$ and space cost in $O(n)$.

The next lemma proves the claim on the ratio of optimality for this algorithm.

Lemma 3.1.2

If T_a is the outcome of Algorithm 3.3 then,

$$|T_a| \geq 1/2 |T_{max}|$$

Proof As demonstrated in Lemma 3.1.1, we need only to consider the case where T_{max} has one corner in one and two corners in the second semicircular polygon. See Fig. 3.2, where ΔXYZ represents T_{max} , and without loss of generality, assume that the top half of P is H_1 and the bottom is H_2 . Also assume that B and X lie in different half planes defined by a line passing through ZY .

Fig. 3.2: Possible configuration of $T_{max} = \Delta XYZ$, in relation to the diameter AB , of the convex polygon.

From Fig. 3.2,

$$|T_{max}| = |\Delta YX| = |\Delta XYQ| + |\Delta XZQ|$$

$$\text{but, } |\Delta XYQ| \leq |T_B|, |\Delta XZQ| \leq |T_B|$$

$$\text{therefore, } |T_{max}| \leq |T_B| + |T_B|$$

from the algorithm, $|T_B| \leq |T_a|$, implying that

$$|T_{max}| \leq 2|T_a| \text{ or}$$

$$|T_a| \geq 1/2 |T_{max}|$$

Q. E. D.

3.2 Suboptimal Quadrilateral

Given a convex polygon P , denote by Q_{max} as the maximal area quadrilateral inscribed in P . This section presents a simple $O(n)$ algorithm which produces a quadrilateral Q_a , inscribed in P , with the area at least half as much as the area of Q_{max} . Consider the following algorithm:

Algorithm 3.4

INPUT: P

OUTPUT: Q_a

1- Compute diameter AB of P

2- Slice P along AB into two semicircular polygons; H_1 and H_2

3- Compute maximal area triangle T_1 inscribed in H_1

4- Compute maximal area triangle T_2 inscribed in H_2

$Q_a \leftarrow T_1 \cup T_2$

/ This union is possible since T_1 and T_2 share the diameter AB as one of their edges. */*

Complexity

Diameter finding and partitioning in Steps 1 and 2 take $O(n)$ time [Preparata and Shamos, 1990], while the rest of the steps are all in $O(\log n)$. Thus, the overall time and space complexity is in $O(n)$.

The proof of the ratio of optimality of the algorithm is provided in the next lemma.

Lemma 3.1.3

If diameter AB partitions P into two non-empty semicircular polygons, H_1 and H_2 , then at most two corners of Q_{max} can lie exclusively in either semicircular polygon.

Proof: Assume otherwise. Without loss of generality, suppose that all four corners X, Y, Z and W lie in H_1 . Now, since H_2 is non-empty, we can increase the area of Q_{max} by extending a vertex into H_2 . Similarly, assume that three vertices lie in H_1 (See Fig. 3.3).

Fig. 3.3: Q_{max} cannot have more than two vertices in either half of the convex polygon. If it has, then we can always find a bigger Q_{max} .

Let line L be parallel to XZ , and be tangent to P at Y' . It is easy to see that if angle $\mu = \pi/2$ then Y' coincides with B . If, however, $\mu < \pi/2$ then Y' lies in H_2 , implying that $|\Delta XZY'| \geq |\Delta XYZ|$ and that quadrilateral $XY'ZW$ is larger in area than quadrilateral $XYZW$. If $\mu > \pi/2$ then we would consider these arguments for a line parallel to XZ and tangent to P on the left side. In either case, our quadrilateral would have at most two vertices in either of the two semicircular halves.

Lemma: 3.1.4

If Algorithm 3.4 returns quadrilateral Q_a , then

$$|Q_a| \geq 1/2 |Q_{max}|$$

Proof:

Lemma 3.1.3 proved that Q_{max} can have only two possible configurations in P . These are depicted in figures 3.4a and 3.4b.

Case 1:

Depicted in Fig. 3.4a, algorithm 3.4 returns exactly Q_{max} .

Fig. 3.4a: Q_{max} may have two of its vertices at A and B , and the other two at highest points on H_1 and H_2 .

Case 2:

See Fig. 3.4b. Let M and N be the intersections of WX and ZY with the diameter AB , respectively.

$$|Q_{max}| = |WXYZ| = |\Delta WMZ| + |\Delta MZN| + |\Delta MXN| + |\Delta XNY|$$

$$|\Delta WMZ|, |\Delta MZN| \leq |T_1|$$

$$|\Delta MXN|, |\Delta XNY| \leq |T_2|$$

$$|Q_{max}| \leq |T_1| + |T_1| + |T_2| + |T_2|$$

from the definition of Q_a ,

$$|Q_{max}| \leq |Q_a| + |Q_a|$$

$$|Q_a| \geq 1/2 |Q_{max}|$$

Q.E.D.

Fig. 3.4b: Two vertices of Q_{max} may lie on one half and two on the other

3.3 Asymptotically Optimal Inscribed Square

In the previous two sections, approximation algorithms were given for the inclusion problem for triangles and quadrilaterals in convex polygons. These algorithms had fixed ratios of optimality. In contrast, the algorithms developed in the rest of this thesis will have ratio of optimalities that asymptotically tend to unity at the expense of computational costs. These algorithms follow the proposed approach mentioned in the introduction of this thesis. See Section 1.5.2.

In this section, the first of these algorithms is presented. The algorithm described computes asymptotically maximal area inscribed square in the given convex polygon. Let k be an arbitrary parameter and let S_{max} be the maximal area inscribed square in P . Our algorithm, in time $O(kn)$, produces an inscribed square S_{alg} such that,

$$\frac{|S_{alg}|}{|S_{max}|} \geq \frac{1}{\left(1 + \tan\left(\frac{\pi}{2k}\right)\right)^2}$$

This last inequality shows that better and better approximations of S_{max} can be achieved at the expense of increasing costs proportional to k .

The type of machinery used in this section and Section 3.5 will set precedence for solving inclusion and enclosure problems in the rest of this thesis. We shall soon see that extraordinary results can be achieved by following the ideas presented here. The main idea is to somehow construct an *algorithmic* square, S_{alg} , whose area asymptotically approaches the area of the optimal square.

Lemma : 3.3.1

Given the square S in Fig. 3.3.1, and an angle θ , if S' is the maximal area inscribed square in S one of whose sides makes an angle of θ with the vertical side of S , then,

$$\frac{|S'|}{|S|} = \frac{1 + \tan^2 \theta}{(1 + \tan \theta)^2}$$

Proof:

Let S have area equal to unit squared, and assume that the binding corner of S' divides a side of S into portions of lengths x and $1-x$. See Fig. 3.3.1.

Fig. 3.3.1: The maximal area inscribed axial square within a square

From Fig. 3.3.1,

$$\begin{aligned} \frac{|S'|}{|S|} &= x^2 + (1-x)^2 \\ &= (1-x)^2 \left\{ \left(\frac{x}{1-x} \right)^2 + 1 \right\} \\ &= \frac{1 + \left(\frac{x}{1-x} \right)^2}{\left(\frac{1}{1-x} \right)^2} \end{aligned}$$

$$= \frac{1 + \left(\frac{x}{1-x}\right)^2}{\left(1 + \frac{x}{1-x}\right)^2}$$

but,

$$\tan \theta = \frac{x}{1-x}$$

therefore,

$$\frac{|S'|}{|S|} = \frac{1 + \tan^2 \theta}{(1 + \tan \theta)^2}$$

Q.E.D.

Outline of the Algorithm

Consider the following simpler subproblem:

Axial Square Problem:

Given a convex polygon P and a direction d , find the maximal area square inscribed in P one of whose sides is along the direction d .

Suppose that we already have an algorithm that solves the axial square problem. The idea of the algorithm for the inscribed square is as follows. We choose k directions uniformly from $[0, \pi/2]$, and using the, as yet unknown, algorithm for the axial square problem, compute k axial maximal area inscribed squares $S_1, S_2, S_3, \dots, S_k$ such that one side of S_{i-1} makes an angle of $\pi/2k$ with some side of S_i . It should be clear that the maximum of all these S_i squares approach S_{max} as k tends to infinity. Indeed, Lemma 3.3.2 will show that if S_{alg} is taken as the maximum of all S_i , then it has the claimed lower bound on the ratio of optimality.

Given the parameter k , convex polygon P , the following algorithm outputs the sub optimal maximal area inscribed square S_{alg} .

Algorithm 3.3.1

INPUT: Convex polygon P , parameter k .

OUTPUT: S_{alg} .

1- Consider k directions $d_1, d_2, d_3, \dots, d_k$. $d_1 = \pi/2k$

2- For $i = 1$ to k

$S_i \leftarrow$ axial maximal area square with one side parallel to d_i .

$|S_{alg}| \leftarrow \text{MAX} \{ |S_i| : 1 \leq i \leq k \}$

Complexity

Steps 1 and 3 are clearly linear in n . In the following section, a linear algorithm for solving the axial square problem is given. Step 2 repeats this axial square algorithm k times. Therefore, the overall complexity of our algorithm is $O(kn)$.

Lemma : 3.3.2

If S_{max} is the maximal area inscribed square in convex polygon P and S_{alg} is the output of algorithm 3.3.1 then,

$$\frac{|S_{alg}|}{|S_{max}|} \geq \frac{1}{\left(1 + \tan\left(\frac{\pi}{2k}\right)\right)^2}$$

Proof. See Algorithm 3.4.1. Since the k squares $S_1, S_2, S_3, \dots, S_k$ are such that any two consecutive squares have sides that make an angle $\mu \leq \pi/2k$, there exists an i , such that one side of S_i makes an angle μ with one side of S_{max} . Suppose S' is the maximal area square inscribed in S_{max} one of whose sides makes an angle of μ with one side of S_{max} . Clearly, by construction of S_i , $|S_i| \geq |S'|$. And from Lemma 3.3.1 we know that

$$\frac{|S'|}{|S_{max}|} = \frac{1 + \tan^2 \mu}{(1 + \tan \mu)^2}$$

Now, for $\mu \leq \pi/2k$,

$$\frac{|S'|}{|S_{\max}|} \geq \frac{1}{\left(1 + \tan\left(\frac{\pi}{2k}\right)\right)^2}$$

Since $|S_i| \geq |S'|$ and $|S_{alg}| \geq |S_i|$ for any $1 \leq i \leq k$, we can conclude that

$$\frac{|S_{alg}|}{|S_{\max}|} \geq \frac{1}{\left(1 + \tan\left(\frac{\pi}{2k}\right)\right)^2}$$

Q.E.D

It remains to give the algorithm for the axial square problem.

3.3.1 Algorithm for the Axial Square Problem

This section gives a linear algorithm for solving the axial square problem used in the algorithm for finding maximal area inscribed squares in a convex polygon. The problem statement is as follows. Given a convex polygon P , and a direction d , give an algorithm that computes the maximal area inscribed square in P , one of whose sides is parallel to the direction d . Because we now have no freedom for the direction of the square we are searching for, it should be obvious that this problem is simpler than the problem of finding the maximal area inscribed squares. Since this is our first axial problem, we describe the algorithm in considerable details.

Without loss of generality, assume that the given convex polygon P is located in the positive quadrant and the direction d is along the positive x -axis. Denote the desired maximal area inscribed axial square inside P as S . It is not very difficult to see that S is adjacent to at least two points on P . For if S had no binding corners then a simple magnification gives a larger square. And also, if S had only one binding corner, then a vertical translation followed by a magnification produces a bigger square. Furthermore, it

is clear that the binding corners of S are opposite to one another because a single adjacent pair of binding corners allows magnification of the square.

Let A and B (D and C) be the top (respectively, bottom) left and right corners of S , respectively. We shall assume that A and C are the binding corners (This also handles the case for which D and B are binding, if P is rotated by an angle of $\pi/2$).

Notation

Let P be in the positive y and positive x quadrant in the Cartesian coordinates. Define the *left end* and *right end* of P as the points on P with the minimum x coordinate and maximum x coordinate, respectively. Denote the upper and lower sequence of edges from the left to the right end of P as the chains C_U and C_L , respectively.

Short Description

The idea of the algorithm is as follows. For each pair of points for A and C along the upper and lower chains, respectively, the Loci for the D and B corners of S are traced out. From these loci of D and B , those set of points for which the associated square lies within P are chosen. And finally, from these sets of corners for D and B , the pair of corners most distant from one another are picked by solving a simple optimization problem. Such an optimal pair of corners for D and B together with their associated A and C corners determine the desired square S .

Formal Description

Consider the following partitioning of P into trapezoidal slabs. Draw 135 degree lines passing through the vertices of P . Define the trapezoidal region, formed by the intersection of the region between two consecutive slant lines and P , as a slab. Denote by s_{ij} the slab which is bounded by edge e_i from C_U and e_j from C_L . This partitioning gives us at most $n-1$ slabs and at most $n-3$ of which are trapezoids. See Fig. 3.3.1.

Fig. 3.3.1: Partitioning of the polygon and a candidate maximal area square.

Consider a typical slab s_{ij} . If corner A has coordinates $(t, m_i t + b_i)$ for some arbitrary value of t on the x -axis within an interval for which s_{ij} is defined for, then we can compute the loci of points for the other corners in terms of this parameter t as follows:

$A(t) :$

$$\{t, m_i t + b_i\}$$

Equation for $AC :$

$$y = -x + (m_i + 1)t + b_i$$

and intersection of AC and e_j gives us the coordinates for $C(t)$.

$$C(t): \left\{ \frac{(m_i + 1)t + b_i - b_j}{m_i + 1}, m_j \frac{(m_i + 1)t + b_i - b_j}{m_i + 1} + b_j \right\}$$

From the geometry of the situation, the loci for D and B can be derived.

$$D(t): \left\{ t, m_j \frac{(m_i + 1)t + b_i - b_j}{m_i + 1} + b_j \right\}$$

$$B(t): \left\{ \frac{(m_i + 1)t + b_i - b_j}{m_j + 1}, m_i t + b_i \right\}$$

In the formulas above t is an arbitrary parameter lying in the domain of the portion of e_i that bounds slab s_{ij} from above. It is worth noting that $D(t)$ and $B(t)$ are chains of $O(n)$ line segments, each segment derived from some slab in the partitioning of P . Therefore, it is easy to see that by performing $O(1)$ amount of work within each slab, the entire loci of $D(t)$ and $B(t)$ can be computed in $O(n)$ time and storage.

Remark::

$B(t)$, $D(t)$, C_L and C_U all intersect at the *left* and *right ends* of P . These intersection points correspond to the trivial point square of 0 area at the left and right ends of P .

Lemma 3.3.3

Other than the intersections at the left and right ends of P , $D(t)$ may only intersect with C_L , and at most once. Similarly, $B(t)$ may only intersect with C_U , and at most once.

Proof. The two cases are similar, so only the case for $D(t)$ is considered. If $D(t)$ intersected with C_U , then the A corner does not lie inside P , which is a contradiction. Secondly, $D(t)$ always lies to the left of $C(t)$, which in turn lies on C_L , both with equal y-coordinates. Thus, from the convexity of P , it is clear that if $D(t)$ lies in P and above C_L , it will always lie above and to the left of C_L . And if it lies outside P , it may enter it once and then as before always lie to the left and above C_L . Q.E.D.

For each value of the parameter t , there exists a square with binding corners $A(t)$ and $C(t)$. However, since $B(t)$ and $D(t)$ may lie outside P , some of these squares may not be contained in P . Therefore, the next obvious step is to find that subset of values of t , for which the associated squares do in fact lie within P . Aiding in this idea, Lemma 3.3.3 indicates that either $B(t)$ and $D(t)$ completely lie inside P , or they leave or enter the interior of P only once. Thus, by taking the non-trivial chain intersection of $D(t)$ and C_L the portion of $D(t)$ lying inside P can be found. Similarly, the subchain of $B(t)$ lying within P can also be found. The intersection of the intervals of the t values for which these subchains of $B(t)$ and $D(t)$ are defined can then be taken. This gives us the set of those values for t , for which the corresponding envelope of squares are all inscribing and in addition to this, each one has a pair of binding opposite vertices.

The main work, then, involves the finding of the intersection between the $B(t)$ and C_U , and between $D(t)$ and C_L . It is not difficult to see that each of these chains are monotone with respect to the direction d . Thus, the intersections can be easily found by the following simple $O(n)$ technique:

- Partition the chains into vertical slabs by taking vertical lines emanating from the vertices of the chains.
- Within each slab, check to see if the only edges inside it intersect.

- Report yes if in some slab the contained edges intersect, otherwise the chains do not intersect.

Finding the maximal area axial Square

Once the interval of t for which the corresponding square is inscribing and has a pair of opposite binding corners has been found, the original problem of finding the maximal such square boils down to solving $O(n)$ simple optimization problems. Specifically, the distance between $A(t)$ and $C(t)$ gives us the magnitude of the diagonal, denoted by $\partial(t)$, of the associated square. More formally, within slab s_{ij} ,

$$|\partial(t)| = \sqrt{2} \left\{ \frac{(m_i - m_j)}{m_j + 1} t + \frac{b_i - b_j}{m_j + 1} \right\}$$

Since $\partial(t)$ is a linear function of t , it attains maximum value when the value of t coincides with either of the end points of the interval of the domain of the part of e_i that bounds s_{ij} from above. In other words, maximum diagonal in a slab is one of the left or right slant lines that bound the slab. The formal description of the algorithm is given below:

Algorithm 3.3.2

INPUT: Convex P and direction d .

OUTPUT: Maximal area inscribed square S which has one side parallel to d .

1- Place P on the Cartesian coordinates so that d is parallel to the x -axis.

2- Partition P into slabs s_{ij} .

3- For each slab s_{ij} , do

- compute next segments of chains $B(t)$ and $D(t)$.

4- $I \leftarrow$ Interval of t for which chains $B(t)$ and $D(t)$ both lie in P .

5- For t in I , consider those slabs which contain $A(t)$, and for all such slabs,

-find maximum value of

$$|\alpha(t)| = \sqrt{2} \left\{ \frac{(m_i - m_j)}{m_j + 1} t + \frac{b_i - b_j}{m_j + 1} \right\}$$

- This gives us the maximum diagonal in each slab.

Let $D \leftarrow \text{MAX} \{ \text{diagonals from Step 5} \}$

$|S| \leftarrow 1/2 D^2$

6- Rotate P by $\pi/2$ and repeat Steps 1-6, and take the maximum of the two squares.

Complexity

Steps 1 and 2 are clearly $O(n)$. As shown above, computing the chains $B(t)$ and $D(t)$ and finding their intersections with the upper and lower chains of P , respectively, can be done in $O(n)$ time and space. Thus, Steps 3 and 4 are in $O(n)$. In Step 5, $O(n)$ optimization problems are solved each taking constant time. Rotating P and repeating the algorithm in Step 6 adds an extra $O(n)$ amount of cost. Therefore, the overall algorithm has linear time and space complexity.

3.4 Equilateral Triangles

This section deals with the problem of finding the maximal area inscribed equilateral triangle in a convex polygon. The best known algorithm for such problem is due to DePano *et. al.* [DePano et. al., 1987a] and has a complexity of $O(n^2)$. Using the type of machinery used for computing asymptotically optimal squares in the previous section, this section will show that there exists an $O(kn)$ algorithm that computes an equilateral triangle T_{alg} , inscribed in P such that

$$\frac{|T_{alg}|}{|T_{max}|} \geq \frac{\tan^2\left(\frac{k-1}{k} \cdot \frac{\pi}{3}\right) + 1}{4}$$

where, T_{max} is the unknown maximal area equilateral triangle in P .

General Outline

As in the case for squares, We first compute, given an equilateral triangle T and angle θ , the maximal area equilateral triangle T' , inscribed in T such that one side of T' makes an angle of θ with one side of T , and compute the ratio of the area of T' to T . Next, we consider the following subproblem:

Axial Equilateral Triangle Problem:

Given P and a direction d , what is the maximal area equilateral triangle in P , one of whose sides is along d ?

This section is completed by showing that the maximum of all axial equilateral triangles in P , along some specific k directions, qualifies as T_{alg} .

Lemma 3.4.1

Let T be an equilateral triangle of side 1, and θ be an angle less than $\pi/2$. If T' is the maximal area triangle in T , one of whose sides makes an angle of θ with one side of T , then

$$\frac{|T'|}{|T|} = \frac{\tan^2(\theta - \frac{\pi}{4}) + 1}{4}$$

Proof: It can be easily shown that T' must have all three of its corners adjacent to the sides of T . For if only two were adjacent, then translation of T' along a line bisecting the interior angle of the non-binding corner of T' will render T' bound by one corner (cases for 0 or 1 corner binding are even more obvious).

Fig. 3.4.1 Computing the ratio of the area of T' to T .

Suppose that one corner of T' divides the corresponding side into segments of length x and $1-x$, and let s be the length of one side of T' . See Fig. 3.4.1. From the similarity of the three triangles in T other than T' , and the law of cosines, we obtain the value of s ,

$$s^2 = (1-x)^2 + x^2 - 2x(1-x)\cos \frac{\pi}{3}$$

which simplifies to

$$s^2 = 3x^2 - 3x + 1.$$

Since the area of an equilateral triangle is $\sqrt{3}/4$ of the square of its side,

$$|T'| = \frac{\sqrt{3}}{4} s^2 = \frac{\sqrt{3}}{4} (3x^2 - 3x + 1)$$

$$|T| = \frac{\sqrt{3}}{4}$$

and the desired ratio is given by

$$\frac{|T'|}{|T|} = 3x^2 - 3x + 1 = 3\left(x - \frac{1}{2}\right)^2 + \frac{1}{4}$$

In Fig. 3.4.1, let $\triangle ABC$ be the triangle in T below T' , re-drawn in Fig. 3.4.2.

Fig. 3.4.2: $\triangle ABC$ is the triangle in T below T' , from Fig. 3.4.1

Using elementary trigonometry, from $\triangle ABC$, it is clear that $|CD| = \sqrt{3}/2 (1-x)$

and $|AD| = 1/2(1-x)$, while $|DB| = x - 1/2(1-x) = 1/2(3x-1)$. From these we obtain,

$$\tan\left(\theta - \frac{\pi}{6}\right) = \frac{\frac{1}{2}(3x-1)}{\frac{\sqrt{3}}{2}(1-x)}$$

From the identity:

$$\tan\left(\theta - \frac{\pi}{6}\right) = \frac{\tan \theta - \tan \frac{\pi}{6}}{1 + \tan \theta \tan \frac{\pi}{6}}$$

we get,

$$\begin{aligned} \frac{\tan \theta - \frac{1}{\sqrt{3}}}{1 + \tan \theta \frac{1}{\sqrt{3}}} &= \frac{3x-1}{\sqrt{3}(1-x)} \\ \Leftrightarrow (1-x)(\sqrt{3} \tan \theta - 1) &= (3x-1)(1 + \frac{1}{\sqrt{3}} \tan \theta) \end{aligned}$$

With a bit of algebraic manipulations, we get

$$\tan \theta = \frac{\sqrt{3} x}{2 - 3x}$$

and from this equation we can express x in terms of $\tan \theta$ as follows,

$$x = \frac{2 \tan \theta}{\sqrt{3} + 3 \tan \theta}$$

substituting this value of x into the ratio of $|T'|$ to $|T|$ we get,

$$\begin{aligned} \frac{|T'|}{|T|} &= 3 \left(\frac{\tan \theta - \sqrt{3}}{2\sqrt{3} + 6 \tan \theta} \right)^2 + \frac{1}{4} \\ &= \frac{1}{4} \left(\frac{\tan \theta - \sqrt{3}}{1 + \sqrt{3} \tan \theta} \right)^2 + \frac{1}{4} \end{aligned}$$

The quantity inside the bracket can be recognized as the tangent of $\theta - \pi/3$, which completes the proof.

$$\frac{|T'|}{|T|} = \frac{\tan^2(\theta - \frac{\pi}{3}) + 1}{4}$$

Q.E.D.

Assume that there is an $O(n)$ algorithm for solving the axial equilateral triangle problem (This algorithm is presented in Section 3.4.1). Following is a formal description of the algorithm that produces the equilateral triangle T_{alg} satisfying the ratio of optimality given in the introduction to this section.

Algorithm 3.4.1

INPUT: Convex polygon P , parameter k .

OUTPUT: T_{alg} .

1- Consider k directions $d_1, d_2, d_3 \dots d_k$, $d_i = \pi/3k$

2- For $i = 1$ to k

$T_i \leftarrow$ axial maximal area equilateral triangle with one side parallel to d_i .

(See Section. 3.4.1)

$|T_{alg}| \leftarrow \text{MAX} \{ |T_i| : 1 \leq i \leq k. \}$

Complexity

The linear algorithm of Step (2) is repeated exactly k times giving us a total time and space complexity of $O(kn)$ and $O(n)$, respectively.

The proof of the result is given in the next lemma.

Lemma 3.4.2

If T_{alg} is the output of Algorithm 3.4.1 and T_{max} is the maximal area equilateral triangle in P , then

$$\frac{|T_{alg}|}{|T_{max}|} \geq \frac{\tan^2\left(\frac{k-1}{k} \cdot \frac{\pi}{3}\right) + 1}{4}$$

Proof. The proof is similar to the proof for the inscribed square problem. See Lemma 3.3.2. Among all the k triangles $T_1, T_2, T_3, \dots, T_k$, there must exist an i , $1 \leq i \leq k$, such that one side of T_i makes an angle $\theta \leq \pi/3k$ with some side of the unknown T_{max} . Now, suppose T' is the maximal area equilateral triangle inscribed in T_{max} , one of whose sides makes an angle of θ with one side of T_{max} . Lemma 3.4.1 tells us that

$$\frac{|T'|}{|T_{max}|} = \frac{\tan^2\left(\theta - \frac{\pi}{3}\right) + 1}{4}$$

From Algorithm 3.4.1, $|T_{alg}| \geq |T_i| \geq |T'|$, thus

$$\frac{|T_{alg}|}{|T_{max}|} = \frac{\tan^2\left(\theta - \frac{\pi}{3}\right) + 1}{4}$$

And since $\theta \leq \pi/3k$, for a sufficiently large k ,

$$\tan^2\left(\theta - \frac{\pi}{3}\right) \geq \tan^2\left(\frac{\pi}{3k} - \frac{\pi}{3}\right) \geq \tan^2\left(\frac{k-1}{k} \cdot \frac{\pi}{3}\right)$$

Hence,

$$\frac{|T_{alg}|}{|T_{max}|} \geq \frac{\tan^2\left(\frac{k-1}{k} \cdot \frac{\pi}{3}\right) + 1}{4}$$

Q.E.D

The algorithm for the axial equilateral problem is given in the next section.

3.4.1 Axial Equilateral Triangles

This section gives the solution to the *axial equilateral triangle* problem as promised above. We are given a convex polygon P and a direction d , and we are to find the maximal area equilateral triangle T_d , in P , one of whose sides is along direction d .

As it turns out that this algorithm is very similar to the algorithm for computing maximal area axial squares of Section 3.3. Please see Section 3.3. Let A , B and C be the corners of T_d . It should be obvious that at least two corners of T_d lie on the boundary of P .

Briefly, as in the case of axial squares (Section 3.3.1), we partition P into $O(n)$ slabs by drawing lines parallel to d , through each vertex of P . Within each slab s , we place the A and B corners of T_d on the two edges of P in s , and trace the position for the C corner on either side of AB . It has been shown before that such a loci of points, say, C_s and C_s' , form straight line segments. Let C_s be the corner in the left half plane defined by a line passing through AB , and C_s' be its reflection through this line. Since the C_s and C_s' are similar, we shall consider the case of C_s only. We are interested in the end points of C_s , for it is these points, at which the associated equilateral triangle attains the maximum area among all other equilateral triangles with their A and B corners in S . Recall that it might be the case that C_s has a non-trivial intersection with the boundary of P . In such situations,

this point of intersection simply trims the length of C_s and gives us a different pair of end points to maximize the area of T_d on.

Repeating the above procedure for each slab produces a chain R , of $O(n)$ line segments C_s . Once we know the possible intersections of R and the boundary of P , we can proceed to maximize the area of T over each slab.

In Lemma 3.3.1, due to the convexity of P , we showed that loci such as C_s and C_s' have the property that these intersect the boundary of P at most once. By going through each slab, we can determine the possible intersection of C_s and the boundary of P , by looking for slabs where the vertices of C_s shift from outside to inside P , in $O(n)$ time. Finally, we simply compare the triangles associated with the end points of all C_s and C_s' . The complete algorithm is given below.

Algorithm 3.4.2

1) *Partition P into Slabs.*

2) *For each Slab s*

Compute C_s' and C_s .

Let R be the ordered chain formed by joining consecutive C_s .

Let R' be the ordered chain formed by joining consecutive C_s' .

3) *By going through each slab, find the possible intersection of R and the boundary of P .*

$R \leftarrow$ Portion of R inside P .

Repeat Step 3) for R'

4) *For each edge in R , consider the area of the associated triangle, take the maximum.*

Repeat 4) for R'

$T_d \leftarrow$ The triangle with maximum area among triangles from Steps 4) and 5)

Complexity

It is easy to see that the entire algorithm for finding the maximal area axial equilateral triangle runs in $O(n)$ time and space.

3.5 Asymptotically Optimal Inscribed Rectangle

The previous two sections showed how the proposed approach can be used to find the asymptotically optimal inscribed square and equilateral triangles in convex polygons. This section uses this same machinery to find asymptotically optimal inscribed rectangle in a convex polygon. In contrast to the case of the squares and equilateral triangles where we approximated only one parameter, namely, the direction of the maximal square, two parameters need to be approximated here. In addition to the direction of the maximal area rectangle, its ratio is also unknown.

This section presents an algorithm which produces an approximation to the maximal area inscribed rectangle R_{max} in the convex set, to a given arbitrary degree of accuracy. It will be shown that given a convex polygon P , and an arbitrary accuracy parameter k , our algorithm will produce a rectangle R_{alg} , in time $O(k^2 n)$, with a ratio of optimality that tends to unity against increasing values of k . Equivalently,

$$\frac{|R_{alg}|}{|R_{max}|} \geq \left(\frac{k\rho(P)}{k\rho(P) + 8\pi} \right)^2 \left(1 - \frac{8}{k\rho(P)} \right)$$

Where the constant $\rho(P)$ denotes the *ratio* of P , defined later, and depends exclusively on the geometry of the convex set and not its size.

3.5.1 Ratio of a Convex Polygon

The maximal area inscribed rectangle inside a given polygon obviously depends only on the geometry of the convex set. Equivalently, there must exist some relationship between the geometry of the rectangle and the given polygon. One parameter that contains such geometric information of a convex set is its so called *ratio*, defined below.

Given a vector $\underline{v} \in \mathcal{R}^2$, let $c(\underline{v})$ be the length of the longest chord in the given convex polygon C , along the direction of $\underline{v}$. Denote the *length* and *width* of C as $L(C)$ and $W(C)$, respectively, and define these as follows:

$$W(C) = \text{MIN} \{ c(\underline{v}) \mid \underline{v} \in \mathcal{R}^2 \}$$

$$L(C) = \text{MAX} \{ c(\underline{v}) \mid \underline{v} \in \mathcal{R}^2 \}$$

Then the *ratio* of C , $\rho(C)$, is defined as the ratio of its width to its length.

$$\rho(C) = \frac{W(C)}{L(C)}$$

Note that this is different from the usual notion of a ratio, say, that of a rectangle which is simply the ratio of the lengths of its sides. Further note that the *length* of a convex polygon is simply its *diameter*. This concept of *ratio* of a polygon will be useful when we attempt to compute ratio of optimality for R_{alg} . ♦

The lemma below gives one relationship between the ratio of a convex set and the ratio of the maximal area rectangle that can be inscribed in it.

Lemma 3.5.1

If σ denotes the ratio of the maximal area rectangle R_{max} , inscribed in convex P , then,

$$\sigma \geq \frac{\rho(P)}{8}$$

Proof. Let AB be the diameter of P (so $|AB| = L(P)$). Let C and D be points in P above and below AB , respectively, that have maximum distances from AB . Let X and Y be the midpoints of AC and BC , respectively. Furthermore, let X' and Y' be the projections of X and Y on AB , respectively. See Fig. 3.5.1.

Fig. 3.5.1: AB is the diameter of P , and C and D are points in P with maximum distances from AB , above and below AB , respectively. Ratio of $XX'Y'Y$ is at least $1/8$ of the ratio of P

Without loss of generality assume that ΔACB is larger than ΔADB . Let R be the rectangle $XX'Y'Y$. Let H_C and H_D be the heights of triangles ΔABC and ΔABD , respectively. Note that ΔCXY is similar to ΔABC , it follows from elementary geometry that,

$$|XY| = |X'Y'| = \frac{1}{2}|AB|$$

and,

$$|XX'| = \frac{1}{2}H_C.$$

Area of R can be now be given in terms ΔABC

$$|R| = |XX'| \times |X'Y'| = \frac{1}{2}H_C \times \frac{1}{2}|AB| = \frac{1}{2}|\Delta ABC|$$

but by assumption,

$$|\Delta ABC| \geq \frac{1}{2}|ACBD|$$

Thus,

$$\begin{aligned} |R| &\geq \frac{1}{4} |ACBD| = \frac{1}{4} \{ |\Delta ABC| + |\Delta ABD| \} \\ &= \frac{1}{4} \left\{ \frac{1}{2} |AB| (H_D + H_C) \right\} \end{aligned}$$

Thus, by the definition of the width of P ,

$$|R| \geq \frac{1}{8} L(P) W(P) \dots\dots\dots (*)$$

Let rectangle R_{max} have width a and length b , and ratio σ . Suppose that,

$$\sigma = \frac{1}{t} \rho(P)$$

Thus, we have to show that $t \leq 8$.

$$\begin{aligned} |R_{max}| &= ab \\ &= \frac{a}{b} b^2 \\ &= \sigma b^2 \\ &= \frac{1}{t} \rho(P) b^2 \end{aligned}$$

b is at most the length of P ,

$$\begin{aligned} &\leq \frac{1}{t} \frac{W(P)}{L(P)} L(P)^2 \\ &\leq \frac{1}{t} W(P) L(P) \end{aligned}$$

Now, from $|R_{max}| \geq |R|$, (*) and the previous equation we obtain,

$$\frac{1}{8} W(P) L(P) \leq |R| \leq |R_{max}| \leq \frac{1}{t} W(P) L(P)$$

allowing us to deduce that $t \leq 8$. Q.E.D.

The lower bound for the ratio of optimality follows from the result in Lemma 3.5.2.

Lemma 3.5.2

Let R be a rectangle with width a and length b and ratio $\sigma = a/b$. Given a ratio r , $0 < r < 1$, and an angle $\theta \leq \pi/2$. If R' is the maximal area rectangle with ratio r inscribed in R and whose longer side makes an angle of θ with the longer side of R , then it satisfies the following:

$$\frac{|R'|}{|R|} \geq \frac{\sigma}{r(1 + \frac{r}{r})^2}$$

Proof: See Fig. 3.5.2. Let a corner of R' partition the side of R it is adjacent to, into segments of length x and $b-x$. Observe that the triangles bounded above by the lower sides of R' and from below by the base of R are similar. And from the similarity of these triangles and some elementary geometry, the area of R' can be computed:

Fig. 3.5.2: Maximal area *axial* inscribed rectangle within a rectangle.

$$|R'| = \left(\frac{x}{\cos \theta} \right) \left(\frac{rx}{\cos \theta} \right) = \left(\frac{x}{\cos \theta} \right)^2 r$$

From Fig. 3.5.2,

$$\tan \theta = \frac{a - rx}{x}$$

$$\Leftrightarrow x \tan \theta = a - rx$$

$$\Leftrightarrow x = \frac{a}{r + \tan \theta}$$

Substituting this value of x into the formula for R' gives us,

$$|R'| = \frac{a^2 r}{\cos^2 \theta (r + \tan \theta)^2} = \left(\frac{a}{r \cos \theta + \sin \theta} \right)^2 r$$

For $\theta \leq \pi/2$, $\sin \theta \leq \theta$ and $\cos \theta \leq 1$, giving us the lower bound on the area of R .

$$|R'| \geq \left(\frac{a}{\theta + r} \right)^2 r$$

Now, $|R| = ab$, so

$$\frac{|R'|}{|R|} \geq \frac{\frac{a}{\theta + r}}{\frac{a}{\theta + r}} r = \frac{\sigma}{r(1 + \frac{\theta}{r})^2}$$

Q.E.D.

In addition to the orientation of the optimal solution, we must also approximate its ratio. The subproblems now involve the computation of maximal area rectangles whose direction and ratio are fixed before hand. This subproblem is formally described below.

Axial Fixed Ratio Rectangle problem:

Given a convex polygon P , a ratio r , $0 < r < 1$, and a direction d , what is the maximal area rectangle that can be inscribed in P and which has ratio r and whose length is along the direction d .

Section 3.5.2 gives an $O(n)$ algorithm for solving this problem. From the previous two lemmas, we know that σ , the ratio of R_{max} , lies somewhere in the interval $[\rho(P)/8, 1]$. Following the technique of the corresponding algorithm for squares (See Section 3.3), given the accuracy parameter of k , we shall choose k appropriate ratios $r_1, r_2, r_3, \dots$ and r_k , uniformly from the interval $[0, 1]$, and k angles: $\theta_1, \theta_2, \theta_3, \dots$ and θ_k uniformly from $[0, \pi]$ (from the symmetry in a fixed ratio rectangle). Next, the *Axial Fixed Ratio Rectangle* problem is solved and the following k^2 axial fixed ratio rectangles are computed. Denote by R_{ij} the maximal area inscribed rectangle in P , whose length makes an angle of θ_i with the horizontal and whose ratio is r_j . The claim is that if R_{alg} is taken as the maximum of R_{ij} , for $1 \leq i, j \leq k$, then such an R_{alg} satisfies the following,

$$\frac{|R_{alg}|}{|R_{max}|} \geq \left(\frac{k\rho(P)}{k\rho(P) + 8\pi} \right)^2 \left(1 - \frac{8}{k\rho(P)} \right)$$

This result is proved in Lemma 3.5.3. But first, the formal description of the algorithm is given.

Algorithm 3.5.1

Compute R_{alg}

INPUT : P, k

OUTPUT: R_{alg}

1) For $i \leftarrow 1$ to k

$$\theta_i \leftarrow \frac{\pi}{k} i$$

$$r_i \leftarrow \frac{1}{k} i$$

2) For $i \leftarrow 1$ to k

For $j \leftarrow 1$ to k

Compute R_{ij}

$$|R_{alg}| \leftarrow \text{MAX } \{ |R_{ij}| : 1 \leq i, j \leq k \}$$

Complexity

Step (1) is in $O(k)$. In Step 2, k^2 calls are made to the algorithm in Section 3.5.2. Each of these calls take $O(n)$ time and space. Therefore, the over all time complexity is in $O(k^2 n)$ and space complexity is in $O(n)$.

Lemma 3.5.3

If R_{alg} is the output of Algorithm 3.5.1 then

$$\frac{|R_{alg}|}{|R_{max}|} \geq \left(\frac{k\rho(P)}{k\rho(P) + 8\pi} \right)^2 \left(1 - \frac{8}{k\rho(P)} \right)$$

Proof. From the distribution of the r_i 's from the interval $[0, 1]$, there must exist a t , $1 \leq t \leq k$ such that

$$r_i \geq \sigma$$

and

$$\begin{aligned} r_i - \sigma &\leq \frac{1}{k} \\ \Rightarrow 1 - \frac{\sigma}{r_i} &\leq \frac{1}{kr_i} \end{aligned}$$

and from Lemma 3.5.1,

$$\begin{aligned} r_i &\geq \sigma \geq \frac{\rho(P)}{8} \\ \Rightarrow \frac{\sigma}{r_i} &\geq 1 - \frac{8}{k\rho(P)} \dots\dots\dots (*) \end{aligned}$$

As reasoned in the analogous case of the maximal squares (Section 3.3), there must exist some s , $1 \leq s \leq k$, such that the angle β , between the corresponding sides of R_{max} and R_{st} is less than π/k . Let R be the maximal area inscribed rectangle in R_{max} whose length makes an angle of β with the length of R and whose ratio is r_t . By construction and definition,

$$|R_{alg}| \geq |R_{st}| \geq |R|$$

Let σ be the ratio of R_{max} , then from Lemma 3.5.2,

$$\begin{aligned} \frac{|R|}{|R_{max}|} &\geq \frac{\sigma}{r_t \left(1 + \frac{\beta}{r_t}\right)^2} \\ \Rightarrow \frac{|R_{alg}|}{|R_{max}|} &\geq \frac{\sigma}{r_t \left(1 + \frac{\beta}{r_t}\right)^2} \end{aligned}$$

Using $\beta \leq \pi/k$ and $r_t \geq \rho(P)/8$ we get,

$$\frac{|R_{alg}|}{|R_{max}|} \geq \frac{1}{\left(1 + \frac{\pi\tau}{k\rho(P)}\right)^2} \frac{\sigma}{r_t}$$

and substituting (*) into this, together with a bit of algebra, gives us the desired result

$$\frac{|R_{alg}|}{|R_{max}|} \geq \left(\frac{k\rho(P)}{k\rho(P) + 8\pi} \right)^2 \left(1 - \frac{8}{k\rho(P)} \right)$$

Q.E.D.

2 OF/DE 2

PM-1 3½"x4" PHOTOGRAPHIC MICROCOPY TARGET
NBS 1010a ANSI/ISO #2 EQUIVALENT

Performance of the Algorithm

From the lower bound on the ratio of optimality for this algorithm, it is apparent how the ratio of the polygon affects the rate of convergence of agreement between R_{alg} and R_{max} . The closer $\rho(P)$ is to unity the better the approximation, and the closer it is to zero the worse the agreement between R_{alg} and R_{max} . This fact becomes clear with the special polygon depicted in Fig. 3.5.3.

Fig. 3.5.3. A special polygon for which the rectangle algorithm has poor performance.

This special polygon consists of two regions joined by a relatively long rectangular slab. Consequently, its ratio is *close* to 0. Clearly, if this slab is relatively long, the optimal rectangle is the shaded region R in Fig. 3.5.3. However, in order for R_{alg} to have a "good" agreement with R_{max} , its longer side must have direction sufficiently close to horizontal. In other words, the parameter k must be sufficiently large.

The performance dependence on the geometry of the given polygon exists for most asymptotic algorithms given in this thesis. Theoretically, there exist polygons for which even infinitely large k may not be sufficient to achieve a desired degree of accuracy. However, for practical applications, extreme polygons such as the one shown in Fig. 3.5.3 may be unlikely. In fact, for most practical situations, there exists a non-zero lower bound on the ratio of the given polygon. For instance, for a standard computer screen, the ratio of given polygon can be no less than $1/(\text{length of the diagonal of the screen})$.

3.5.2 Algorithm for Axial Fixed Ratio Rectangle problem

Given a ratio ρ , $0 < \rho < 1$, and a direction d , this section gives the algorithm that computes in time $O(n)$ the maximal area rectangle that can be inscribed in P and which has ratio ρ and whose length is along the direction d . Let us denote such a rectangle by R .

It is not difficult to see that such an algorithm is exactly like the algorithm for finding maximal area axial squares (see Section 3.3.1). The only differences are mechanical in nature, arising from the fact a diagonal of R makes an angle of $\pi - 2\tan^{-1}(\rho)$ with a side of R instead of $\pi/4$ as for a square. Nevertheless, such an angle is fixed and minor changes to the axial square algorithm could transform it to one that solves the axial fixed ratio rectangle problem. As a matter of formality, these necessary changes are described in some detail while the reader is referred to the axial square algorithm of Section 3.3.1.

Alterations to Section 3.3.1

Please see Section 3.3.1. The reader can quite easily be convinced that the essential lemmas of Section 3.3.1 hold true in here as well. Most of the alterations center around the change in the angle between the crossing diagonals, from 90° for the square to $\pi - 2\tan^{-1}(\rho)$ for the fixed ratio rectangle.

Because of this added parameter ρ , the slant slabs are now formed using lines at an angle of $\pi - \tan^{-1}(\rho)$, instead of 135 . The following slight changes in the parametric equations of $A(t)$, $B(t)$, $C(t)$ and $D(t)$ are required.

$A(t)$:

$$\{t, m, t + b, \}$$

$B(t)$:

$$\left\{ \frac{(m_i + \rho)t + b_i - b_j}{m_i + \rho}, m_i t + b_j \right\}$$

$C(t)$:

$$\left\{ \frac{(m_i + \rho)t + b_i - b_j}{m_i + \rho}, m_i \frac{(m_i + \rho)t + b_i - b_j}{m_i + \rho} + b_j \right\}$$

$D(t)$:

$$\left\{ t, m_j \frac{(m_i + \rho)t + b_i - b_j}{m_i + \rho} + b_j \right\}$$

After some tedious algebraic calculations the $\tilde{\alpha}(t)$ function of diagonals is computed to be,

$$|\tilde{\alpha}(t)| = \sqrt{\rho^2 + 1} \left\{ \frac{(m_i - m_j)}{m_j + 1} t + \frac{b_i - b_j}{m_j + 1} \right\}$$

See Section 3.3 for the complete algorithm.

Remarks

It can be easily verified that for $\rho = 1$ this algorithm is identical to Algorithm 3.3.1 which computes the maximal area axial square. Furthermore, none of the above alterations increase the $O(n)$ complexity of Algorithm 3.3.1.

3.6 Asymptotically Optimal Inscribed Arbitrary Triangle

This section presents an $O(k^3 n)$ algorithm for computing an inscribed arbitrary triangle T_{alg} , such that the ratio of optimality of this triangle asymptotically approaches unity and is given by the following.

$$\frac{|T_{alg}|}{|T_{max}|} \geq \left(\frac{k\rho(P)}{k\rho(P) + 8\pi} \right)^2 \left(1 - \frac{2}{k\rho(P)} \right)$$

where T_{max} is the maximal area inscribed arbitrary triangle in the given convex polygon P and $\rho(P)$ is the ratio of P (See Section 3.5.1 for a definition of the ratio of a convex set.).

The best known algorithm for computing maximal area triangles in convex sets is due to Dobkin Synder [Dobkin and Snyder, 1979]. We acknowledge that their $O(n)$ algorithm is more efficient than ours. However, in this thesis, the emphasis is on the machinery proposed and its motivations towards other related problems. Indeed, we will show that the techniques presented in our algorithm when extended to simple polygons will compare to the best known corresponding algorithm for simple polygons. See Section 4.5.

The outline of the algorithm is similar to all the other asymptotic algorithms presented in this thesis. We first need to discuss some relevant attributes of arbitrary triangles.

3.6.1 Fixed Similarity Triangles

Consider the following canonical form of an arbitrary triangle. Denote its longest, second longest and smallest side by a , b and c , respectively. Let h_a , h_b and h_c be the height of the triangle taking a , b and c as bases, respectively. And let α and β be the

angles opposite b and c , respectively. Given a triangle with side a and angles α and β in the above canonical form, denote it by the canonical notation $T(a, \alpha, \beta)$, and the class of all triangles similar to $T(a, \alpha, \beta)$ by the *similarity class* $\Gamma(\alpha, \beta)$. All triangles in some similarity class are known as *fixed similarity triangles*. For example, an equilateral triangle belongs to the similarity class $\Gamma(\pi/3, \pi/3)$.

Consider the following problem.

Axial Fixed Similarity Triangle:

Given a convex polygon P , a direction d , and a similarity class $\Gamma(\alpha, \beta)$, what is the maximal area member of $\Gamma(\alpha, \beta)$ which can be contained in P whose longest side is along direction d .

It should be obvious to the reader that the $O(n)$ algorithm for the equilateral triangle problem of the previous section can be easily extended to the present case, without incurring any additional cost of a constant factor of n . Previously, we have done this sort of generalization by carrying the case of the square to that of a fixed ratio rectangle (Section 3.5). Thus, we refer the reader to the associated algorithm for computing maximal area axial equilateral triangles in Section 3.4.

Ratio of a Fixed Similarity Triangle.

We define the ratio r , of a similarity triangle $T(a, \alpha, \beta)$ as follows,

$$r = \text{MIN} \left\{ \frac{h_a}{a}, \frac{h_b}{b}, \frac{h_c}{c} \right\}$$

The lemma below shows that such a ratio is exactly h_a/a .

Lemma 3.6.1

The ratio r of triangle is given by

$$r = \frac{h_a}{a}$$

Proof. Each of the quantities given by

$$\frac{1}{2}bh_b, \frac{1}{2}ah_a, \text{ and } \frac{1}{2}ch_c$$

give the area of the triangle. Thus,

$$ah_a = bh_b$$

Multiplying throughout this equation by b/a^2b gives us,

$$\frac{h_a}{a} = \left(\frac{b}{a}\right)^2 \frac{h_b}{b}$$

Since $b \leq a$, and so $(b/a)^2 \leq 1$, we get,

$$\frac{h_a}{a} \leq \frac{h_b}{b}$$

Similarly,

$$\frac{h_a}{a} \leq \frac{h_c}{c}$$

Since the ratio of a triangle is given by

$$r = \text{MIN} \left\{ \frac{h_a}{a}, \frac{h_b}{b}, \frac{h_c}{c} \right\}$$

The proof is completed.

$$r = \frac{h_a}{a}$$

Q.E.D

Lemma 3.6.2

If σ is the ratio of triangle $T(a, \alpha, \beta)$ then its area can be expressed as

$$|T(a, \alpha, \beta)| = \frac{1}{2}a^2\sigma$$

Proof. Let $T(a, \alpha, \beta)$ be $\triangle ABC$ shown in Fig. 3.6.1. If A is placed on the origin and AB along the x -axis on the Cartesian coordinates, then the intersections of the equations of AC and BC gives us the height of the triangle $T(a, \alpha, \beta)$.

Let the intersection of AC and BC be at (x_m, y_m) . Thus, the area of $T(a, \alpha, \beta)$ can be computed as follows,

$$\begin{aligned} |T(a, \alpha, \beta)| &= \frac{1}{2} a y_m \\ &= \frac{1}{2} a^2 \frac{y_m}{a} \end{aligned}$$

From Lemma 3.6.1, y_m/a is exactly the ratio of $T(a, \alpha, \beta)$, which completes the proof.

$$|T(a, \alpha, \beta)| = \frac{1}{2} a^2 \sigma$$

Q.E.D

Fig 3.6.1: Canonical representation of a fixed similarity triangle $T(a, \alpha, \beta)$.

The next lemma provides the first step in the machinery for this algorithm.

Lemma 3.6.3

If T_{max} is the maximal area triangle contained in convex set P , and σ is its ratio then,

$$\sigma \geq \frac{\rho(P)}{2}$$

Proof. Assume otherwise. Suppose that

$$\sigma < \frac{\rho(P)}{2}$$

Let AB be the diameter of P . Consider the triangle formed by taking AB as the base and the third corner C as a point on the boundary of P farthest from AB . See Fig. 3.6.2. Let

CD be perpendicular to AB and let these line segments intersect at X . Since the width $W(P)$, of a convex polygon is the minimum of the maximum of the length of chords of P along any direction, $|CD|$ is at least as long as $W(P)$. Thus,

$$|CX| + |DX| \geq W(P)$$

Without loss of generality, assume that $|CX| \geq |DX|$, implying that

$$|CX| \geq \frac{1}{2}W(P)$$

The area of ΔABC can be bounded by below as follows.

$$\begin{aligned} |\Delta ABC| &= \frac{1}{2}|AB||CX| \\ &\geq \frac{1}{2}|AB|\frac{1}{2}W(P) \\ |\Delta ABC| &\geq \frac{1}{4}L(P)W(P) \end{aligned}$$

Since $|\Delta ABC| \leq |T_{max}|$, we get,

$$|T_{max}| \geq \frac{1}{4}L(P)W(P).....(I)$$

Fig 3.6.2: ΔABC is contained in P .

Let T_{max} have base b and height h . From the definition of the ratio of a triangle,

$$\sigma \leq \frac{h}{b}$$

and from our assumption at the beginning of this proof, we get

$$\frac{h}{b} < \frac{\rho(P)}{2}$$

$$\Leftrightarrow h < \frac{W(P)}{2L(P)} b$$

It follows that,

$$|T_{\max}| = \frac{1}{2} hb < \frac{W(P)}{4L(P)} b^2$$

But b is at most the length of the polygon, which gives us the following upper bound on the area of $T_{\max}$, and a straight contradiction to (I).

$$|T_{\max}| < \frac{1}{4} W(P)L(P)$$

Hence, because of the above contradiction we can conclude that:

$$\sigma \geq \frac{\rho(P)}{2}$$

Q.E.D.

We now need a result which forms the backbone of the machinery, provided by Lemma 3.6.4.

Lemma 3.6.4

Let T be an arbitrary triangle and let σ be its ratio. Given angles θ , α_I and β_I , let T_I be the maximal area triangle inscribed in T of similarity type $\Pi(\alpha_I, \beta_I)$ whose longest side makes an angle of θ with the longest side of T . If σ_I denotes the ratio of T_I then

$$\frac{|T_I|}{|T|} \geq \left(\frac{1}{1 + \frac{\tan \theta}{\sigma}} \cdot \frac{1}{\cos \theta} \right)^2 \frac{\sigma_I}{\sigma}$$

Proof: Let T have canonical form $T(a, \alpha, \beta)$. It is not too difficult to see that T_I must have the configuration shown in Fig. 3.6.3 (See Lemma 3.4.1 for the reason).

From Lemma 3.6.1,

$$|T_I| = \frac{1}{2} a_I^2 \sigma_I$$

$$|T| = \frac{1}{2} a^2 \sigma$$

which gives us,

$$\frac{|T_I|}{|T|} = \left(\frac{a_I}{a} \right)^2 \frac{\sigma_I}{\sigma}$$

Now, the longest side (its base) of T_I can be described by the equation

$$y = x \tan \theta$$

and the base of T by

$$y = -(x - a) \tan \beta$$

Fig 3.6.3: Maximal area axial fixed similarity triangle inscribed in an arbitrary triangle.

Using elementary algebra, the intersection of these two equations can be computed as the point:

$$\left\{ \frac{a \tan \beta}{\tan \theta + \tan \beta}, \frac{a \tan \theta \tan \beta}{\tan \theta + \tan \beta} \right\}$$

The magnitude of this point gives the value of a_I . See Fig. 3.6.3.

$$a_I = \frac{a \tan \beta}{\tan \theta + \tan \beta} \sqrt{1 + \tan^2 \theta}$$

Using the identity $\sec^2 \theta = 1 + \tan^2 \theta$ we get

$$\frac{a_I}{a} = \frac{\tan \beta}{\tan \theta + \tan \beta} \cdot \frac{1}{\cos \theta}$$

Thus,

$$\frac{|T_1|}{|T|} = \left(\frac{1}{1 + \frac{\tan \theta}{\tan \beta}} \cdot \frac{1}{\cos \theta} \right)^2 \frac{\sigma_1}{\sigma}$$

From Fig. 3.6.3,

$$\tan \beta \geq \frac{\text{height of } T}{a} = \sigma$$

Hence,

$$\frac{|T_1|}{|T|} \geq \left(\frac{1}{1 + \frac{\tan \theta}{\sigma}} \cdot \frac{1}{\cos \theta} \right)^2 \frac{\sigma_1}{\sigma}$$

Q.E.D.

The previous lemma gives us an idea for the lower bound on the ratio of optimality of this algorithm. The complete algorithm is described below.

Formal Description

Consider k directions: $d_1, d_2, d_3, \dots, d_k$ uniformly chosen from $[0, 2\pi]$, k ratios: $r_1, r_2, r_3, \dots, r_k$ chosen from $[0, 1]$, and k angles: $\alpha_1, \alpha_2, \alpha_3, \dots, \alpha_k$ chosen from $[0, \pi/2]$. Observe that we can get a fix on the similarity type of a triangle given one of its angles and its ratio. Furthermore, note that in its canonical form, a triangle cannot have its α angle more than 90° , for if it does so then the base cannot possibly be its longest side. Denote by T_{ijh} as the maximal area axial fixed similarity triangle inscribed in P , whose base is along direction d_i , ratio is r_j , and the angle opposite to its second longest side is α_h . From Sections 3.6.1 and 3.4.1 we know of an $O(n)$ algorithm that can compute T_{ijh} . The claim is that the maximum of T_{ijh} for all $1 \leq i, j, h \leq k$ satisfies the desired triangle T_{alg} .

The proof of this is given in Lemma 3.6.5. But before this, the formal algorithm is in order.

Algorithm: 3.6

INPUT: k, P

OUTPUT: T_{alg} .

```

1) For  $i \leftarrow 1$  to  $k$ 
     $r_i \leftarrow \frac{1}{k} i$ 
     $\alpha_i \leftarrow \frac{\pi}{2k} i$ 
     $\theta_i \leftarrow \frac{2\pi}{k} i$ 

2) For  $i \leftarrow 1$  to  $k$ 
    For  $j \leftarrow 1$  to  $k$ 
        For  $h \leftarrow 1$  to  $k$ 
            Compute  $T_{ijh}$ 

 $|T_{alg}| \leftarrow \text{MAX } \{ |T_{ijh}| : 1 \leq i, j, h \leq k \}$ 

```

Complexity

Steps (1) is in $O(k)$. In the dominating Step (2), we execute the $O(n)$ algorithm for computing fixed similarity axial triangles a total of k^3 times. Therefore, the overall algorithm has time complexity in $O(k^3n)$ and space requirements in $O(n)$.

The next lemma provides a proof for the lower bound on the ratio of optimality for such a T_{alg} .

Lemma 3.6.5

If T_{alg} is the output of algorithm 3.6 and T_{max} is the maximal area triangle in a given convex polygon P , then

$$\frac{|T_{alg}|}{|T_{max}|} \geq \left(\frac{k\rho(P)}{k\rho(P) + 8\pi} \right)^2 \left(1 - \frac{2}{k\rho(P)} \right)$$

Proof. Suppose that T_{max} makes an angle of θ with the horizontal line, has ratio σ , and in the canonical form has α as the angle opposite to its second longest side. From the algorithm, we know that there must exist a triple (x,y,z) , $1 \leq x,y,z \leq k$, such that,

$$|\theta - \theta_x| \leq \frac{2\pi}{k},$$

$$|\alpha - \alpha_x| \leq \frac{\pi}{k},$$

$$r_z \leq \sigma,$$

and

$$\sigma - r_z \leq \frac{1}{k}.$$

In other words, among all triangles T_{ijh} , $1 \leq i,j,h \leq k$, T_{xyz} is closest to T_{max} in similarity and orientation.

Now, let T be the maximal area triangle inscribed in T_{max} whose base makes an angle of $\beta = |\theta - \theta_x| \leq 2\pi/k$ with the base of T_{max} , has ratio r_z and whose second largest angle is α_y . Lemma 3.6.4 gives us the following bound on the area of T relative to the area of T_{max} .

$$\frac{|T|}{|T_{max}|} = \left(\frac{1}{1 + \frac{\tan \beta}{\sigma}} \cdot \frac{1}{\cos \beta} \right)^2 \frac{r_z}{\sigma}$$

From the algorithm, $|T| \leq |T_{xyz}| \leq |T_{alg}|$, giving us the respective result for T_{alg} .

$$\frac{|T_{alg}|}{|T_{max}|} \geq \frac{|T|}{|T_{max}|} \geq \left(\frac{1}{1 + \frac{\tan \beta}{\sigma}} \right)^2 \frac{1}{\cos^2 \beta} \cdot \frac{r_z}{\sigma}$$

Now, in Lemma 3.3.2, we showed that for $\beta \leq 2\pi/k$ and $k \geq 8$, $\tan \beta \leq 4\pi/k$. Furthermore, since $\cos^2 \beta \leq 1$, then $1/\cos^2 \beta \geq 1$. Taking these two inequalities into consideration, we get the following bound on ratio of optimality for T_{alg} .

$$\frac{|T_{alg}|}{|T_{max}|} \geq \left(\frac{1}{1 + \frac{8\pi}{k\sigma}} \right)^2 \frac{r_i}{\sigma} \dots\dots\dots (*)$$

Now, since

$$\begin{aligned} \sigma - r_i &\leq \frac{1}{k} \\ \Leftrightarrow 1 - \frac{r_i}{\sigma} &\leq \frac{1}{\sigma k} \\ \Leftrightarrow \frac{r_i}{\sigma} &\geq 1 - \frac{1}{k\sigma} \end{aligned}$$

Substituting this information into (*) gives us the inequality,

$$\frac{|T_{alg}|}{|T_{max}|} \geq \left(\frac{1}{1 + \frac{8\pi}{k\sigma}} \right)^2 \left(1 - \frac{1}{k\sigma} \right) \dots\dots\dots (**)$$

From Lemma 3.6.3,

$$\sigma \geq \frac{\rho(P)}{2}$$

Finally, this last inequality and (**) complete the proof of this lemma.

$$\begin{aligned} \frac{|T_{alg}|}{|T_{max}|} &\geq \left(\frac{1}{1 + \frac{8\pi}{k\rho(P)}} \right)^2 \left(1 - \frac{1}{k \frac{\rho(P)}{2}} \right) \\ \Leftrightarrow \frac{|T_{alg}|}{|T_{max}|} &\geq \left(\frac{k\rho(P)}{k\rho(P) + 8\pi} \right)^2 \left(1 - \frac{2}{k\rho(P)} \right) \end{aligned}$$

Please see similar proof in Lemma 3.5.3. Q.E.D

In Section 4.4, with a slight amount of modification, we show how the above machinery can be used to develop an $O(k^3 n^2 \log n)$ algorithm for computing an asymptotically optimal inscribed triangle in an arbitrary polygon.

Chapter Summary

This chapter presented some asymptotic algorithms for solving the inclusion and enclosure problems for convex polygons. Major results included the algorithms for finding asymptotically optimally inscribed square, rectangle, equilateral triangles, and arbitrary triangle. See Table 3.1 for a summary of these results.

INCLUSION ALGORITHMS FOR CONVEX POLYGONS

APPROXIMATING CONVEX	COMPLEXITY	RATIO of OPTIMALITY
QUADRILATERAL	$O(n)$	$1/2$
EQUILATERAL TRIANGLE	$O(k n)$	At least: $\frac{\tan^2\left(\frac{k-1}{k} \cdot \frac{\pi}{3}\right) + 1}{4}$
TRIANGLE	$O(n)$	$1/3$
TRIANGLE	$O(n \log n)$	$1/2$
TRIANGLE	$O(k^3 n)$	At least $\left(\frac{k\rho(P)}{k\rho(P) + 8\pi}\right)^2 \left(1 - \frac{2}{k\rho(P)}\right)$
SQUARE	$O(k n)$	At least $\frac{1}{\left(1 + \tan\left(\frac{\pi}{2k}\right)\right)^2}$
RECTANGLE	$O(k^2 n)$	At least: $\left(\frac{k\rho(P)}{k\rho(P) + 8\pi}\right)^2 \left(1 - \frac{8}{k\rho(P)}\right)$

Table 3.1. Summary of results for inclusion algorithms presented in Chapter 3.

Chapter 4

INCLUSION PROBLEMS FOR SIMPLE POLYGONS

This chapter presents the most significant results of this thesis. We will see how the work on Convex Polygons can be naturally extended to the case of Simple Polygons. Given a simple polygon P and a regular convex polygon R , this chapter will give asymptotic algorithms that compute the maximal area inscribing object of similarity type R . Specifically, we give algorithms for computing maximal area inscribed rectangles, squares, arbitrary triangles and equilateral triangles in arbitrary polygons.

Throughout this chapter, we shall denote the given simple polygon by P . The machinery used for these algorithms follows directly from the analogous work on convex polygons in Chapter 3. We start this chapter by extending the concept of *ratio* of a convex polygon to simple polygons.

4.0.1 Ratio of a Simple Polygon

Denote by $\rho(P)$ the *ratio* of a simple polygon P . Similar to the ratio of a convex set, this ratio is still the ratio of the *width* of P to its *length*. However, a bit different definition for width and length of simple polygons is necessary. Let Γ be the set of all maximal length chords in P . For every $L \in \Gamma$, take two lines A and B , above and below L , respectively, parallel to L and with maximum distance from L such that A and B have at least one point in common with P and no reflex vertex of P lies in the region formed by taking the intersection of P and the region between A and B . See Fig. 4.1.

Let W_L be the distance between A and B . Define the *width* of P , denoted by $W(P)$, as the minimum of W_L , for all L in Γ . Also, define the length $L(P)$, as the length of a L in Γ (all have same length).

$$W(P) = \text{MIN} \{ |W_L| : L \in \Gamma \}$$

$$L(P) = |L|, \quad L \in \Gamma$$

Thus,

$$\rho(P) = \frac{W(P)}{L(P)}$$

Fig. 4.1: Computing the width of P w.r.t to the longest chord L .

4.1 Asymptotically Optimal Inscribed Square

The inscribed square problem is to find the maximal area square that can be contained in P . Suppose that $S_{\max}$ is the maximal area inscribed square in P , and k is an arbitrary parameter that effects the accuracy of the result. The algorithm presented in this section produces, in time $O(kn^2 \log n)$, an inscribing square S_{alg} such that,

$$\frac{|S_{\text{alg}}|}{|S_{\max}|} \geq \frac{1}{(1 + \tan(\frac{\pi}{2k}))^2}$$

DePano *et. al.* [DePano *et. al.*, 1987a] express some difficulties attempting to solve this problem in time $O(n^3)$.

The machinery used in the corresponding problem for the convex set, as in Section 3.3, is also valid here. The difference lies in the fact that we are now dealing with simple polygons. Consequently, the algorithm for computing axial squares in convex polygon needs to be replaced by the respective algorithm for computing axial squares in simple polygons. That is, we need an algorithm that computes the maximal area inscribed square in P , one of whose sides is along some given direction d . See Section 4.2 for an algorithm that solves this problem in time $O(n^2 \log n)$.

The outline of the complete algorithm is given below.

Algorithm 4.1

INPUT: Simple polygon P , parameter k .

OUTPUT: S_{alg} .

1- Consider k directions $d_1, d_2, d_3 \dots d_k$. $d_i = \pi/2k$

2- For $i = 1$ to k

$S_i \leftarrow$ axial maximal area square in P with one side parallel to d_i .

(Using algorithm from Section 4.2)

$|S_{alg}| \leftarrow \text{MAX} \{ |S_i| : 1 \leq i \leq k \}$

Complexity

The complexity of $O(kn^2 \log n)$ is obtained by executing the $O(n^2 \log n)$ algorithm of Section 4.2, a total of k times.

Lemma 4.1.1

If S_{max} is the maximal area inscribed square in convex polygon P and S_{alg} is the output of Algorithm 4.1 then,

$$\frac{|S_{alg}|}{|S_{max}|} \geq \frac{1}{(1 + \tan(\frac{\pi}{2k}))^2}$$

Proof. See proof of Lemma 3.3.2.

4.2 Algorithm for Inscribed Axial Square

This section gives an $O(n^2 \log n)$ algorithm for computing the maximal area inscribed square in a given simple polygon P , one of whose sides is along a given direction d . Analogous to the case of maximal area squares in a convex set, here too, the computation of the maximal area axial square, acts as a major subroutine for the problem of finding the largest square and rectangle in a simple polygon. It has been demonstrated previously that algorithms for squares can be effortlessly extended to the case of fixed ratio rectangles. Therefore, only the case of squares is considered, while the case of the largest fixed ratio axial rectangles is assumed automatically.

Denote by S_d our desired maximal area axial inscribed square in P , one of whose sides is parallel to d . The next lemma provides the finiteness criterion for this problem.

Lemma 4.2.1

The square S_d assumes one of the five configurations shown in Fig. 4.2.1.

Proof: Loosely speaking, either opposite sides or opposite corners, or a combination of both, must be bound by edges or reflex vertices of P . Using brute force, we could go through all the possible combinations of edges and reflex vertices binding S_d and convince ourselves that except for the above five cases, all other cases allow S_d to expand in area.

One obtains these configurations by considering all configurations of squares bound by no reflex vertex, one reflex vertex, two reflex vertices, and so on.

Q.E.D.

Fig. 4.2.1: The five possible configurations of S_d in P .

We shall separately deal with each of the five cases in Lemma 4.2.1. Denote by S_1 , S_2 , S_3 , S_4 and S_5 the maximal area squares of the type described in case (1), (2),

(3),(4), and (5), respectively. Without loss of generality, assume that direction d is along the x -axis.

A bit of notation is in order first. Two points in P are *visible* from each other if the line segment joining them does not intersect any edge of P . An important result from the papers by Chazelle and Guibas [Chazelle and Guibas, 1989] and Guibas *et. al.* [Guibas *et. al.*, 1987], that we shall use repeatedly, is as follows. Given a simple polygon P , after a preprocessing of $O(n \log n)$, there is an algorithm which answers queries of the following type, known as *ray shooting*, in logarithmic time:

Given a point x in P and a direction d , what is the first point on the boundary of P intersected by a ray originating from x and directed along d ?

Each of the five cases of this algorithm have complexities more than $O(n \log n)$, thus in the complexity analysis, we will ignore this $O(n \log n)$ steps of preprocessing required for ray shooting.

Case 1: Three Corners adjacent on edges.

The algorithm to handle this case is based on our algorithm for finding the maximal area axial square in a convex set (See Section 3.3.1). First of all, P is partitioned, in much the same way as the convex set was partitioned, into slabs formed by taking consecutive lines among lines making an angle of 45° with d and passing through the vertices of P . Instead of the slabs being bounded by two edges as in the convex case, this time the slabs may be further partitioned by the numerous edges of the polygon. Lets refer to these regions as *subslabs*. It is not difficult to distinguish the subslab regions that are internal to the polygon from the ones that are external. Fig 4.2.2 shows a typical slab and the respective subslabs. Note that the sequence of *outs* and *ins* alternates and it always

begins and ends with *out*. Thus, a linear scan can easily distinguish the *in* subslabs from the *out* subslabs.

Starting from the top left corner, in clockwise direction, let A , B , C and D be the four corners of S_j . In the case of the convex set, within each slab, the loci of points for one pair of opposite corners of S_{max} were computed by tracing its other pair of opposite corners along the two edges defining the slab. However, in the present situation, this procedure for the convex set will be performed for each *in* subslab. That is, the loci for B and D will be computed by tracing the A and C corners on the two edges defining a particular *in* subslab.

Fig 4.2.2: A typical slab and the trapezoidal subslabs within it.

Formal Description

Assume e_i and e_j are the two edges bounding an *in* subslab S_{ij} , from above and below, respectively. Let u_i and u_j be the portions of edges e_i and e_j , respectively, that define S_{ij} . For a point t on u_i , define the square $S(t)$ with corners $A(t)$, $B(t)$, $C(t)$ and $D(t)$, such that $A(t)$ and $C(t)$ lie on u_i and u_j , respectively, representing the end points of the diagonal of $S(t)$. By moving t along u_i , $B(t)$ and $D(t)$ trace out straight line segments. The

parametric definitions of these curves are given in Section 3.3.1. Recall that the area of $S(t)$ was maximized for t at the end points of $D(t)$ or $B(t)$. In our current situation, we are no longer guaranteed that all squares associated with corners on $B(t)$ and $D(t)$ are contained within P . Consequently, we need to determine the interval of t for which $S(t)$ is indeed contained in P . The end points of this interval provides us with values of t for which $S(t)$ has maximum area, and has its diagonal in S_{ij} . The problem of computing this interval boils down to the problem of finding the farthest points on $B(t)$ and $D(t)$ that lie in P . See Fig. 4.2.3.

Fig 4.2.3: The trapezoidal region represents a typical *in* subslab for which the $D(t)=D_aD_b$ and $B(t)=B_aB_b$ line segments have been computed. X (resp. Y) is the intersection of the boundary of P and D_aD_b (resp. B_aB_b) closest to D_b (resp. B_a).

$S(t)$ is not contained in P if $B(t)$ (resp. $D(t)$) corner extends beyond Y (resp. X). Thus, it is imperative that we determine these two points. This is where *ray shooting* comes into play. Ray shooting from B_a and D_b , along B_aB_b and D_bD_a , respectively, gives us points Y and X , respectively. Suppose t_1 and t_2 are such that $Y = B(t_1)$ and $X = D(t_2)$. If Y is visible from $C(t_1)$ (this can be decided by ray shooting from one to the other, in $\log(n)$ time), and from $A(t_1)$, then $S(t_1)$ is contained in P and is of type (1). Similarly, we

can determine if $S(t_2)$ lies in P . S_{ij} is simply the maximum of $S(t_1)$ and $S(t_2)$. Since $B(t)$ and $D(t)$ are linear functions of t , computing t_1 and t_2 from X and Y , respectively, takes $O(1)$ time. Note that, if B_a (or D_b) does not lie in P then $S(t_1)$ (resp. $S(t_2)$) cannot have its B (resp. D) corner visible from both of its A and C corners.

The outline of the algorithm is as follows.

Fig 4.2.4: Example of a polygon with $O(n^2)$ subslabs.

Algorithm 4.2.1:

INPUT: P

OUTPUT: S_I .

Partition P into slabs and subslabs.

For each "in" subslab, compute square S as follows,

Compute $B(t)$ and $D(t)$

Compute X and Y

Let t_1 and t_2 be such that

$Y = B(t_1)$ and $X = D(t_2)$,

$S \leftarrow \text{MAX} \{ S(t_i) \mid S(t_i) \text{ is contained in } P, i=1,2 \}$

$|S_I| \leftarrow \text{MAX} \{ |S| : \text{for all } S \text{ computed above} \}$

Rotate P by $\pi/2$ and repeat the algorithm again.

Complexity

For each subslab, as shown in Section 3.3.1, all operations except ray shooting can be done in constant time. Therefore, with the constant number of ray shootings, each taking $O(\log n)$ time, we perform $O(\log n)$ work for each subslab. Now, the number of subslabs in a simple polygon is in $\Omega(n^2)$. See Fig. 4.2.4. Thus, the overall complexity of handling case 1 is in $O(n^2 \log n)$.

Case 2: Adjacent edges adjacent to reflex vertices and an opposite corner on an edge

See Fig. 4.2.1. Let r_i and r_j be any pair of reflex vertices. For such a pair, following is a method for computing the largest square of the type in case 2, say, S_{ij} . Let L_i and L_j be two lines perpendicular to each other, passing through r_i and r_j , respectively. Let L_i be along d and intersect with L_j at X . By ray shooting towards X , from r_i and r_j , along L_i and L_j , respectively, we can determine if X lies in P or not. If X does not lie in P then no S_{ij} exists. So, assume that X lies in P . It is clear that X qualifies for the C corner of S_{ij} . See Fig. 4.2.5. From X , we ray shoot along a line making an angle of 45 with the d , towards the interior of P and get a point, say, Y . Next, from Y , we ray shoot along L_i and L_j , towards the interior of P , to determine if Y can see its projections on L_j and L_i , respectively. If so, then S_{ij} has diagonal XY and has maximal area among the squares of the type in case 2 that happen to be adjacent to r_i and r_j . Otherwise, S_{ij} does not exist.

Hence, the algorithm simply consists of finding the S_{ij} squares for each pair of reflex vertices, and the maximum of these squares gives us the desired square S_2 .

There are $O(n^2)$ pairs of reflex vertices and for each pair we perform three ray shootings in $O(\log n)$ time. The overall complexity for finding S_2 is in $O(n^2 \log n)$.

We shall use a somewhat different approach this time. Again assume that the direction d is the horizontal line. Let r be a reflex vertex, and L be a horizontal line adjacent on r . Assume, without loss of generality, that the edges adjacent on r lie below L and that L is adjacent on the x -axis. Loosely speaking then, for each reflex vertex that is visible from L , say, r_i , we will compute the square, if one exists, whose opposite edges are adjacent on r and r_i . This process repeated for all the other reflex vertices that are visible from L , will give us some $O(n)$ squares, the maximum of which is clearly our desired square S_3 .

114

can see their vertical projections on AB . More precisely, R is a monotone polygon, with respect to L , whose boundary consists of AB , and a monotone chain M , with respect to L . We shall refer to M as the *vertical visibility* of AB in P . The points of M are all those points from the boundary of P that can see their projections on AB . See Fig. 4.2.6.

Fig 4.2.6: S_3 has all of its corners in R . R is a monotone polygon, *w.r.t.* to L , and its boundary is composed of AB and the vertical visibility, M , of AB in P .

Let r_i be a reflex vertex on the boundary of M . Let s_{r_i} be the square, if it exists, whose bottom and top sides are adjacent to r and r_i , respectively. Clearly the maximum of all s_{r_i} , for all reflex vertices r_i on M , gives us s_r . And furthermore, the maximum of all s_r , for all reflex vertices r in P , gives us the desired square S_3 . Therefore, our next task is to concentrate on computing s_{r_i} .

Let X and Y be the results of ray shooting towards M , from r_i , along a line parallel to L , to the left and right, respectively. And let X_I and Y_I be the projections of X and Y on AB , respectively. Note that by the definition of M , these projections are visible

from X and Y . The reader will agree that if s_{ri} exists, it lies completely inside the rectangular region $XY Y_1 X_1$. In $O(1)$ time we can easily determine, assuming it exists, the maximal area square inside $XY Y_1 X_1$ whose sides are parallel to L and are adjacent to both the reflex vertices r and r_i . Fig. 4.2.6 shows the successful computation of s_{ri} .

Fig 4.2.6: Computation of s_{ri} .

The formal description of the algorithm is as follows:

Algorithm 4.2.2

INPUT: P

OUTPUT: S_3 .

For each reflex vertex r in P ,

Construct AB

$M \leftarrow$ vertical visibility of AB in P (See Algorithm 4.2.3 for this subroutine)

For each reflex vertex r_i on M ,

Compute Rectangular Region $XY Y_1 X_1$.

Compute s_{ri} .

$$s_r \leftarrow \text{MAX} \{ s_{ri} : \forall i \}$$

$$|S_3| \leftarrow \text{MAX} \{ |s_r| : r \text{ is a reflex vertex of } P \}$$

Complexity

For each reflex vertex, we compute vertical visibility chains and find the rectangular region, XY_iX_j as above, for each reflex vertex in these chains. As the sequel will show, we can find vertical visibility in $O(n)$ time, given that P is cyclically sorted which is true in our case. Every vertex of a vertical visibility is from the boundary of the polygon, thus it has $O(n)$ reflex vertices, and for each reflex vertex from this vertical visibility, we do $O(\log n)$ amount of work for the $O(1)$ number of ray shootings. Thus, $O(n \log n)$ amount of work is done for each of the $O(n)$ reflex vertices in P , giving us a total time complexity of $O(n^2 \log n)$ for finding S_3 .

Computing Vertical Visibility

We now describe the method for finding the vertical visibility chains defined in the above algorithm. Let $s = AB$, and let chain C be that part of the boundary of P that lies above s . The problem of finding the vertical visibility chain is essentially that of finding a subset of C , say, M , such that every point in M can see its vertical projection on s . Described below is a linear time algorithm for this problem, which is really a simplified version of the *point visibility* algorithm of D.T. Lee [Lee, 1983].

Denote by $\text{Proj}(p, s)$ as the projection of point p on s , and without any ambiguity we shall refer to this as the projection of p . Since M is monotone w.r.t to L , the projections of the vertices of M are ordered in an ascending order. Thus, if p is a point on C , as well as on M , then all those points x on C , following p , such that $\text{Proj}(x, s) < \text{Proj}(p, s)$, either block, or are themselves blocked from, s by some previously considered

points of M . The algorithm scans the vertices of C from A to B , and continues to add suitable ones on M , or remove the now discovered unsuitable ones from M . There are three possible scenarios to consider. These are depicted in Fig. 4.2.7(a),(b) and (c).

Fig 4.2.7 (a): Vertices whose projections follow the projections of the previous vertices are simply added to M .

Scanning along the vertices of C , if the projection of the next vertex follows that of the previous one, then it is simply appended on to the current chain of M . See Fig 4.2.7(a).

If on the other hand, the projection of the next vertex of C is less than the projection of the current vertex, then depending on whether it is a right turn or a left turn in C , we would either ignore subsequent vertices of C or remove the now hidden vertices of M until the next vertex is visible from its vertical projection from x . See Fig. 4.2.7 (b) and Fig. 4.2.7 (c).

Fig 4.2.7 (b): If projection of v_i is less than that of v_{i-1} , and we encounter a left turn, then we skip all subsequent vertices of C until we get to an edge ab which is partly vertically visible from s . v_i , x and b are added to M .

Fig 4.2.8 (c): If the vertical projection of v_i is to the left of the vertical projection of v_{i-1} , and we encounter a right turn, then we remove from all previously added vertices until we get to an edge ab which is partly vertically visible from s . x becomes the current vertex of M .

We now give the formal description of the above algorithm. A stack S is used, which is initially empty and at the termination of the algorithm contains the vertices of M . We shall use the standard *Push* and *Pop* procedures in addition to a procedure, say, *TOP*,

which references the element last pushed on S . Denote by $V(x)$ as the vertical line that passes through the point x .

Algorithm 4.2.3

Compute vertical visibility

INPUT: C .

OUTPUT: M

$x \leftarrow$ Intersection of $V(A)$ and C .

Push A .

Push x .

Repeat

 If $\text{Proj}(v_i, s) < \text{Proj}(TOP, s)$ Then

 If edge $v_{i-1}v_i$ makes a left turn Then /* See Fig. 4.1.8(b) */

 advance on C until we find edge ab which intersects $V(TOP)$

$x \leftarrow$ Intersection of $V(TOP)$ and ab .

 Push x

 Push v_i

 Else /* edge $v_{i-1}v_i$ makes a right turn. See Fig. 4.1.8(c) */

 Pop all edges from S until edge ab which intersects $V(v_i)$

$x \leftarrow$ Intersection of $V(v_i)$ and ab .

 Push x . /* Top two elements on S are now a and x */

 Else /* see Fig. 4.1.8(a) */

 Push v_i

Until ($i = n$).

Push B .

$M \leftarrow$ vertices in S , from bottom to top.

Complexity

Observe that at each step of the algorithm, we do one of three things. Either we push a vertex on S , or *Pop* a vertex from S never to consider such a vertex again, or simply advance our index i . Thus, a cost of 3 or 2 is sufficient as the amortized cost of each edge. Therefore, the complexity of computing the vertical visibility is in $O(n)$.

Case 4: Adjacent corners bound by edges and opposite side by a reflex vertex

See case (4) of Fig. 4.2.1. We will show that there is an algorithm which can find S_4 in time $O(n^2 \log n)$. As in case (3), let r be a reflex vertex, and L be a horizontal line adjacent to r and let A and B be the results of ray shooting from r , along L , to the left and right, respectively. Assume, without loss of generality, that the edges of P adjacent to r lie below L and that L coincides with the x -axis. Also, let s_r be the maximal area square of type (4) whose bottom edge is adjacent on r .

The general idea is as follows. Since all four corners of s_r must be mutually visible, its top two corners must lie on the boundary of the vertical visibility of AB in P . See case (3) for a definition of the vertical visibility. Let R be the monotone polygon as in case (3). Simply stated, we search for the top two corners of s_r among the edges of M .

Partition of the region R , into a set of trapezoids is known as a trapezoidation of R . Such *trapezoidation* of R is obtained as follows. For each vertex v_i of M , let X_i and Y_i be the results of ray shooting towards M , from v_i , along L , to the right and left, respectively. Notice that in some cases, X_i or Y_i may coincide with v_i . Denote by T_i the trapezoid formed by the region which is bounded from above by $X_i Y_i$ and from below by a similar line segment directly below $X_i Y_i$, and the respective edges of M . See Fig. 4.2.8 for a depiction of a complete trapezoidation.

Fig 4.2.8: Region R and its trapezoidation.

It can be easily shown that if s_r exists, it must have its top two corners on some edge of M , in some trapezoid. Thus, once we have constructed these $O(n)$ trapezoids in R , we can use Algorithm 2.2.2 from Chapter 2 to search for the top two corners of s_r in each trapezoid separately. Algorithm 2.2.2 will spend $O(1)$ time in each trapezoid and return the unique maximal area square in R , if it exists, whose bottom side coincides with AB and whose top two corners lie on M . If such a square is also adjacent to r , then it is identical to s_r , otherwise s_r does not exist.

Algorithm : 4.2.4

INPUT: P

OUTPUT: S_4 .

For each reflex vertex r

Find L

Find M and R

Perform Trapezoidation of R

For each trapezoid T_i

Let $s_i \leftarrow$ Maximal area square of type (4) with corners in T_i .

```

/* See Algorithm 2.2.2 */
|s| ← MAX { |si| : Ti is a trapezoid }
If bottom side of s is adjacent to r then
    sr ← s
Else
    sr ← an empty square
|S4| ← MAX { |sr| : r is a reflex vertex }

```

Complexity

From Algorithm 4.2.3, computing R and M take $O(n)$ time. The partitioning of R into trapezoids require $O(n)$ ray shootings, one for each vertex in M , giving a total cost $O(n \log n)$. A constant amount of work is done in each trapezoid. Therefore, for each of the $O(n)$ reflex vertices we do $O(n \log n)$ amount of work, giving us a total time complexity in $O(n^2 \log n)$.

Case 5: Opposite corners adjacent to edges and one side on reflex vertex

See Fig. 4.2.1. As in case (3), for each reflex vertex r , we find the line segment AB . See Case (3). Now, from A , we ray shoot towards a direction that makes an angle of 45 degrees with the d , and get point, say, X . Clearly, if square of type (5), with an edge adjacent to r , exists, then X and A are one of two pairs of its opposite corners. The other pair of corners can be obtained by ray shooting from A and X , up and down, respectively. We can determine, by ray shooting, if such a square lies in P and if all of its corners are mutually visible. The same procedure is repeated for the point B . The larger of these two squares gives us the maximal area square of type (5) with an edge adjacent to r . This process repeated for each reflex vertex gives a set of candidate squares for S_5 . Finally,

S_5 is the square with the maximum area among all such candidate squares computed above.

Complexity

We perform a constant number of ray shootings for each of the $O(n)$ reflex vertices, giving us a total time complexity in $O(n \log n)$.

Equipped with all the five algorithms for handling the five separate cases, the algorithm for computing the largest axial square now follows naturally.

Algorithm 4.2.5:

Find S_1

Find S_2

Find S_3

Find S_4

Find S_5

$|S_{max}| \leftarrow \text{MAX } (|S_1|, |S_2|, |S_3|, |S_4|, |S_5|)$

Complexity

Each of $S_1, S_2, S_3, S_4,$ and S_5 are computed using less than $O(n^2 \log n)$ steps. Therefore, the algorithm for finding the maximal area axial square has a time complexity of $O(n^2 \log n)$.

4.2.1 Extension to the Case of Fixed Ratio Axial Rectangles.

Our algorithm for finding maximal area axial squares can be easily modified to the algorithm for finding maximal area fixed ratio rectangle, as we have done so for the case of the convex sets. Thus, it is worth noting that without much change in the outline of the previous algorithm one can also find the maximal area fixed ratio rectangle in time complexity $O(n^2 \log n)$.

4.3 Asymptotically Optimal Inscribed Rectangles

Unlike the case of the square in Section 4.1, the case of the rectangles requires a bit of additional work in extending the algorithm for the convex set to the simple polygon. Let R_{max} be the maximal area inscribed rectangle in the given simple polygon P . This section will provide the necessary machinery for computing a rectangle R_{alg} , whose ratio of optimality asymptotically approaches unity. Given an arbitrary parameter k , representing the usual desired degree of agreement between R_{alg} and R_{max} , the following algorithm, at a cost of $O(k^2 n^2 \log n)$, outputs R_{alg} such that

$$\frac{|R_{alg}|}{|R_{max}|} \geq \left(\frac{k\rho(P)}{k\rho(P) + 8\pi} \right)^2 \left(1 - \frac{8}{k\rho(P)} \right)$$

First, we show that Lemma 3.5.1 also holds for simple polygons.

Lemma 4.3.1

If σ denotes the ratio of the maximal area rectangle R_{max} , inscribed in simple polygon P , then,

$$\sigma \geq \frac{\rho(P)}{8}$$

Proof: Let AB be the longest chord in P . Let L_1 and L_2 be lines parallel to AB , above and below, respectively, with maximum distance from AB , such that these lines have at least one point in common with P and no reflex vertex of P lies in the region formed by taking the intersection of P and the region between L_1 and L_2 . See Fig. 4.3.1. We know that the vertical distance between L_1 and L_2 is at least the width of P . Which implies that one of L_1 or L_2 must have at least a distance of $W(P)/2$ from AB . Without loss of generality, suppose it is L_1 .

Let C be any point lying in the portion of L_I that is contained in P . From the definition of L_I , we know that C will always exist. Clearly, C is visible from A and B . Let M and N be the midpoints of lines AC and BC , respectively. Let m and n be the orthogonal projections of M and N on AB , respectively. It is easy to see that rectangle $MNnm$ lies within P . We have previously proved this case for the convex set. In Lemma 3.5.1, if we replace rectangle $XX'Y'Y$ by rectangle $MNnm$, the proof for this lemma is completed. See Lemma 3.5.1, where we prove that,

$$\sigma \geq \frac{\rho(P)}{8}$$

Fig 4.3.1: Rectangle $MNnm$ is contained in P .

Q.E.D

The complete algorithm for finding the suboptimal maximal area is borrowed from the analogous case for the convex gon. As before, denote by R_{ij} , the maximal area fixed ratio rectangle inscribed in simple polygon P , whose length is along a line that makes an angle of θ_i with the horizontal and whose ratio is given by r_i , where

$$\theta_i \leftarrow \frac{\pi i}{k},$$

$$r_i \leftarrow \frac{1}{k} i$$

for all i , $1 \leq i \leq k$.

The algorithm given below computes R_{alg} .

Algorithm: 4.3

INPUT : P, k

OUTPUT: R_{alg}

1- For $i \leftarrow 1$ to k

$$\theta_i \leftarrow \frac{\pi i}{k}$$

$$r_i \leftarrow \frac{1}{k} i$$

2- For $i \leftarrow 1$ to k

For $j \leftarrow 1$ to k

Compute R_{ij} using algorithm in Section 4.2

$$|R_{alg}| \leftarrow \text{MAX } \{|R_{ij}| : 1 \leq i, j \leq k\}$$

Complexity

Algorithm 4.3 executes the maximal area axial fixed ratio rectangle of Algorithm 4.2 a total of k^2 times. Each execution takes time in $O(n^2 \log n)$, giving an aggregate time complexity in $O(k^2 n^2 \log n)$.

The proof in Lemma 3.5.3 gives the proof of the lower bound for the ratio of optimality for this algorithm. See Lemma 3.5.3. Namely, if R_{alg} is the result of algorithm 4.3 then

$$\frac{|R_{alg}|}{|R_{max}|} \geq \left(\frac{k\rho(P)}{k\rho(P) + 8\pi} \right)^2 \left(1 - \frac{8}{k\rho(P)} \right)$$

4.4 Inscribed Equilateral Triangle

As in the case for squares and rectangles, the work on equilateral triangles for convex polygons can also be carried on to simple polygons. This section will give an algorithm for computing asymptotically optimal inscribed equilateral triangle in a simple polygon. We are given a simple polygon P , and we are to find the maximal area equilateral triangle T_{max} inscribed in P . Our algorithm produces an equilateral triangle T_{alg} , in time $O(kn^2 \log n)$, whose area asymptotically approaches the area of T_{max} , such that

$$\frac{|T_{alg}|}{|T_{max}|} \geq \frac{\tan^2\left(\frac{k-1}{k} \cdot \frac{\pi}{3}\right) + 1}{4}$$

Recall the $O(n^3)$ result by DePano *et. al.* [DePano et. al., 1987a] for this problem.

The work follows from the analogous case of the equilateral triangle in convex polygon (Section 3.4). As usual, the difference lies in the fact that we are now dealing with simple polygons instead of convex. Consequently, there is need for an algorithm for finding the maximal area inscribed axial equilateral triangle in a simple polygon. Section 4.5 does deals with this problem. With this algorithm in place, the analogous work for the convex polygon is valid here as well.

As a matter of formality, the complete algorithm for finding T_{alg} is as follows:

Algorithm : 4.4.1

INPUT: Convex polygon P , parameter k .

OUTPUT: T_{alg} .

1- Consider k directions $d_1, d_2, d_3 \dots d_k$, $d_i = \pi/3k$

2- For $i = 1$ to k

$T_i \leftarrow$ axial maximal area equilateral triangle with one side parallel to d_i .

(Using algorithm from Section 4.5)

$$|T_{alg}| \leftarrow \text{MAX} \{ |T_i| : 1 \leq i \leq k \}$$

Complexity

The complexity consists of running the $O(n^2 \log n)$ algorithm in Step (2) (See Section 4.5) k times. Thus, such T_{alg} can be computed in time $O(k n^2 \log n)$.

Lemma 4.4.1

If T_{alg} is the output of Algorithm 4.4.1 and T_{max} is the maximal area equilateral triangle in P , then

$$\frac{|T_{alg}|}{|T_{max}|} \geq \frac{\tan^2\left(\frac{k-1}{k} \cdot \frac{\pi}{3}\right) + 1}{4}$$

Proof. See Lemma 3.4.2.

4.5 Inscribed Axial Equilateral Triangle

The algorithm presented in this section is the major subroutine for the algorithms for computing maximal area equilateral and arbitrary triangles of Sections 4.4 and 4.6, respectively. Given a direction d , the algorithm below, in time $O(n^2 \log n)$, finds the maximal area inscribed equilateral triangle T one of whose sides is parallel to d .

We state without proof that T can only have one of the five possible configurations shown in Fig. 4.5.1 (See the analogous case of the square, Lemma 4.2.1, for a brief reasoning). We shall refer to the triangles in the five cases as of type 1, 2, 3, 4, and 5, respectively.

The algorithm would, separately, search maximal area triangles of the five types T_1, T_2, T_3, T_4 and T_5 , and take the maximum of these five triangles as T . The general techniques for finding such triangles is similar to one for the squares in a simple polygon. Without loss of generality, assume the direction d is horizontal.

In each of the five cases below, we shall assume that the given polygon has already been preprocessed for ray shooting, at a cost of $O(n \log n)$ [19,20]. Since each of the five cases take more time than $O(n \log n)$, we shall ignore this additional cost, of preprocessing, in the complexity analysis.

Fig 4.5.1: The five possible configurations that T can have in P .

Case 1: All corners adjacent on edges.

See Fig. 4.5.1. Quite simply, we partition the simple polygon into *trapezoidal subslabs* as in the case of the squares in a simple polygon (See Section 4.2, Fig 4.2.2). This time, the slabs are parallel to d . Our algorithm searches for T_I in each of the subslab lying in P . Each subslab is a trapezoidal region and we can search for T_I using the analogous algorithm for the convex polygon (See Section 3.8.1). Briefly, for each subslab, we place two corners of T_I on the respective edges and trace out the locus of the third corner of T_I . This locus is a straight line segment, say L . By ray shooting from the internal end point of L , along L , we find the farthest points on L that lies in P . From this point, by ray shooting along the sides of the corresponding triangle, we can determine if it is visible from the other two corners. We do the same thing for the locus constructed by rotating the triangle by 180 degrees. One of the two triangles associated with these two points is T_I . It is clear that such T_I is maximal and is of type 1. We refer the reader to Section 4.2 for a look at similar ideas.

For each subslab, the work involved is a constant number of ray shootings. Considering that there can be as many as $O(n^2)$ such subslabs, and the logarithmic cost of ray shooting, the entire cost for computing T_I is in $O(n^2 \log n)$.

Case 2: Two corners adjacent on edges and the opposite side on a reflex vertex.

See Fig. 4.5.1. Let r be a reflex vertex of P and let L be a horizontal line that passes through r , with the edges adjacent to r lying below L . Let T_r be the maximal area equilateral triangle of type (2). We know that one side of T_r is adjacent to L , and we can search for the corners of T_r as follows. Ray shoot from r , along L , in either direction, and let the resulting points be X and Y . Next, from X , ray shoot towards the interior of P ,

along a line that makes an angle of 60 degrees with the horizontal line. Let Z be the result of this last ray shooting. See Fig. 4.5.2.

Fig 4.5.2: Finding maximal area triangle of type (2) that is adjacent to r .

Let W be the intersection of L and the line that makes an angle of 120 degrees with L and passes through Z . Now if W is visible from Z (This can be decided by ray shooting along the line passing through ZW) and lies between r and Y , then we have just found a triangle of type 2 with the X and Z corners binding on edges. We could repeat this process by ray shooting from Y this time instead of from X and get another triangle of type (2). The larger of these two triangles is T_r . Of course, it could happen that Z is not visible from W in which case the respective triangle does not exist.

Thus, the process of finding T_2 consists of taking the maximum of T_r for all reflex vertices r . And we have seen that this process involves a constant number of ray shootings. Therefore, if T_2 exists then we can find it in $O(n \log n)$ steps.

Case 3: Two sides adjacent on reflex vertices and the opposite corner on an edge.

See Fig. 4.5.1. Let r_i and r_j be two reflex vertices. Let L_i and L_j be two lines that pass through r_i and r_j , respectively, make an angle of 60 degrees with each other and intersect at X . See Fig. 4.5.3. The following simple procedure will determine the maximal

Let W be the intersection of L_i and a line that makes an angle of 60 degrees with L_i and passes through Z . If W lies between Y and r_i , and is visible from Z then we have a candidate for T_{ij} , namely, ΔXZW , otherwise such a triangle does not exist. It is also important to consider the other candidate for T_{ij} whose one corner is adjacent on the edge of P that contains Y . We can find such a triangle by the same process as before. If at least one of these triangles exist, then the maximum of these satisfy the conditions for T_i .

134

Case 4: All three sides adjacent to reflex vertices.

See Fig. 4.5.1. As usual, let r be a reflex vertex, and L be a horizontal line adjacent to r . Assume, without loss of generality, that the edges of P , adjacent to r , lie below L and that L coincides with the x -axis. Denote by T_r the maximal area equilateral triangle of type (4) whose bottom side is simultaneously adjacent to L and r . The algorithm will search for T_r for each reflex vertex and take the maximum of these triangles as T_d .

The general idea of the algorithm follows the algorithm in case 4 of the maximal area axial square problem from Section 4.2. The basic observation is as follows. Each of the three corners of T_r must be mutually visible. Thus, it makes sense to look for T_r among those points of P that can see their projections on L along 60 and 120 degree lines. This point will become clear after the following notation.

α -visibility

For a point x , and a line L , denote by $\alpha\text{-Proj}(x, L)$ the point on L , such that the line segment joining x and $\alpha\text{-Proj}(x, L)$ makes an angle of α with L . We shall refer to $\alpha\text{-Proj}(x, L)$ as the α -projection of x on L . Note that if L is a line segment, $\alpha\text{-Proj}(x, L)$ may not exist. Further note that for $\alpha = 90^\circ$, $\alpha\text{-Proj}(x, L)$ is the usual vertical projection of x on L . Similarly, we define the notion of α -visibility in a polygon. Given a line segment L in P , the α -visibility of L in P is a region in P , say, R , such that every point in R can see its α -projection on L . Note that the vertical visibility discussed in Section 4.2 is none other than a 90-visibility. Consequently, with minor modifications to the vertical visibility algorithm of Section 4.2 (Case 4), α -visibility of L in P can also be computed in $O(n)$ number of steps.

In the present case, we are concerned with 60-visibility and 120-visibility. Let A and B be the results of ray shooting from r , along L , to the left and right, respectively.

Let R_{60} be the region in P , obtained by constructing the 60-visibility of AB in the region of P lying above L . Next, let R be the 120-visibility of AB in R_{60} . In other words, R is the intersection of the 60-visibility and 120-visibility of AB in the region of P lying in the upper half plane defined by L . It is clear that every point of R can now see its (A)-projection and 120-projection on AB . Let M be the boundary of R not including AB . Figure 4.5.4 below illustrates the R region of a simple polygon.

Fig. 4.5.4. The R region is the intersection of the 60-visibility and 120-visibility of AB in P . R is obtained by taking the 60-visibility of the 120-visibility of AB , or *vice versa*, in the region of P lying above P .

Our claim is as follows.

Lemma 4.5.1

For a reflex vertex r of P , if T_r exists, then both of its top sides must be adjacent to two consecutive edges of M .

Proof: Let X, Y and Z be the three corners of T_r . Since T_r is of type (4), let the top two sides of T_r be adjacent to reflex vertices u and v of P . Let u and v be on XZ and YZ , respectively. See Fig. 4.5.5.

Let I be the region in P defined by the simple polygon formed by taking its boundary as the clock wise chain from u to v , followed by edges vZ and Zu . The claim is that no point in I lies in R . This is because the computation of the 60-visibility throws away all those points from I that lie to the left of a line passing through XZ , keeping only those that lie to the right of a line passing through ZY , and the computation of the 120-visibility throws away all these remaining points. Thus, R has no points from I . However, every point from T_r is in R , thus it must be the case that edges uZ and Zv are edges on the boundary of R , which is M . Since uZ and Zv are consecutive edges, our proof is completed. Q. E. D

Fig. 4.5.5. $\triangle XYZ$ is T_r . No point in region I lies in R , while every point in $\triangle XYZ$ does. Thus, uZ and Zv are consecutive edges in M .

With the result in Lemma 4.5.1, the method for searching T_r is clear. After we have computed M , for all those consecutive edges in M , that make an interior angle of 60 degrees, we find the obvious equilateral triangle that can be placed there. Specifically, let

$e_i = v_{i-1}v_i$ and $e_{i+1} = v_iv_{i+1}$ be two consecutive edges of M such that e_i and e_{i+1} make an interior angle of exactly 60 degrees. Denote by T_{ri} the largest area equilateral triangle, with its base adjacent to AB and r , whose top sides are adjacent to e_i and e_{i+1} . Note that if the triangle with corners at $60\text{-Proj}(v_i, AB)$, v_i and $120\text{-Proj}(v_i, AB)$, has its base passing through r as well, then it qualifies as T_{ri} . The bottom line is that in constant time we can find T_{ri} , and the maximum of these triangles, for all possible i , gives us T_r . The complete algorithm is given below.

Algorithm : 4.5.1

INPUT: P

OUTPUT: T_4 .

For each reflex vertex r in P

Compute R and M / Algorithm in Section 4.2, case 4. */*

For each edge e_i in M

Compute T_{ri} .

$|T_r| \leftarrow \text{MAX} \{ |T_{ri}| : e_i \text{ is an edge of } M \text{ and } r \text{ is a reflex vertex} \}$

$|T_4| \leftarrow \text{MAX} \{ |T_r| : r \text{ is a reflex vertex of } P \}$

Complexity

For each of the $O(n)$ reflex vertices of P , the vertical visibility algorithm of Section 4.2 can be easily modified to compute R and M in time $O(n)$. Also, a constant number of ray shootings are performed and an additional $O(n)$ time is spent looking for T_r . Thus, the overall time complexity is $O(n^2)$.

Case 5: One side adjacent to two edges and a reflex vertex.

See Fig. 4.5.1. This last case is relatively simple. As in case (2), for each reflex vertex r , we find the chord XY that is parallel to d and is adjacent to r . See Fig. 4.5.3. From points X and Y , we take two lines at angles 60 and 120 degrees from XY , respectively. Let Z be the intersection of these lines. Clearly, if triangle of type (5), with a side adjacent to r , exists, then it must have corners X, Y and Z . By using a constant number of ray shootings, we can determine if Z is visible from X and Y , and if so, the triangle so formed is a candidate for T_5 . Such a procedure, repeated for all reflex vertices of P , gives us T_5 .

Complexity

We perform a constant number of ray shootings for each of the $O(n)$ reflex vertices, giving us a total time complexity in $O(n \log n)$.

With the algorithms for the above four cases in place, finally, the algorithm for computing maximal area axial equilateral triangle is as follows.

Algorithm : 4.5.2:

INPUT : P, d .

OUTPUT: T .

Find T_1 .

Find T_2 .

Find T_3 .

Find T_4 .

Find T_5 .

$|T| \leftarrow \text{MAX } \{|T_1|, |T_2|, |T_3|, |T_4|, |T_5|\}$

Complexity

The five cases in the above algorithm have costs in $O(n^2 \log n)$, $O(n \log n)$, $O(n^2 \log n)$, $O(n^2)$ and $O(n \log n)$, respectively. Therefore, the overall cost of finding the maximal area axial equilateral triangle is $O(n^2 \log n)$.

4.6 Asymptotically Optimal Inscribed Arbitrary Triangle

The problem discussed in this section is that of finding the maximal area arbitrary triangle, say, T_{max} , inscribed in a simple polygon P . Hitherto, the best known algorithm for solving this problem is due to Melissaratos and Souvaine [Melissaratos and Souvaine, 1992]. Their algorithm has time complexity in $O(n^4)$.

In this section, we present an asymptotic algorithm, which outputs an inscribed triangle T_{alg} , in time $O(k^3 n^2 \log n)$, inscribed in the given polygon P such that

$$\frac{|T_{alg}|}{|T_{max}|} \geq \left(\frac{k\rho(P)}{k\rho(P) + 8\pi} \right)^2 \left(1 - \frac{2}{k\rho(P)} \right)$$

where k is an arbitrary parameter and $\rho(P)$ is the familiar ratio of the simple polygon P . The outline of the algorithm is identical to its corresponding algorithm for the convex set. See Section 3.6. The difference is due to the fact that we must now compute axial fixed similarity inscribed triangles in simple polygons, instead of in convex polygons as in Section 3.6.

Axial Fixed Similarity Inscribed Triangle

The problem is as follows:

Given a simple polygon P , a direction d , and a similarity class $\Gamma(\alpha, \beta)$, what is the maximal area member of $\Gamma(\alpha, \beta)$ which can be contained in P whose longest side is along direction d

In Section 4.5, we gave an algorithm for solving a special case of the above problem, namely, when $\alpha = \pi/3$ and $\beta = \pi/3$, or the problem of computing maximal axial equilateral triangles in simple polygons. We remark that such algorithm for equilateral

triangles can be mechanically generalized to the algorithm for fixed similarity triangles. See algorithms in Section 4.5.

In general, the sides of a fixed similarity triangle are mutually distinguishable as opposed to indistinguishable as it is the case for an equilateral triangle. Therefore, for each of the four cases presented in Section 4.5, we must now consider all permutations of the respective configurations. However many, there are a constant number of permutations for each case (at most 12). It follows that the additional amount of work required to extend the algorithm for the equilateral triangle case to the algorithm for the fixed similarity triangle case is definitely in $O(n^2 \log n)$. Hence, we conclude that we can find the maximal area inscribed axial fixed similarity triangle in a simple polygon in time $O(n^2 \log n)$.

♦

We now need a lemma similar to Lemma 3.6.3.

Lemma: 4.6.1

If T_{max} is the maximal area triangle contained in simple polygon P , and σ is its ratio, then

$$\sigma \geq \frac{\rho(P)}{2}$$

Proof. Assume otherwise. Suppose that $\sigma < \rho(P)/2$. Let AB be the longest chord in P . So $|AB| = L(P)$. Let L_1 and L_2 be lines parallel to AB , above and below, respectively, with maximum distance from AB , such that L_1 and L_2 have at least one point in common with P and no reflex vertex of P lies in the region formed by taking the intersection of P and the region between L_1 and L_2 . Let W be the distance between L_1 and L_2 , and H the distance between L_1 and AB . Note that from the definition of the ratio of a simple polygon (See beginning of this chapter for a definition of the ratio of a simple polygon), W is at least the width of P . Without loss of generality, assume that AB is farther from L_1 than L_2 . Let C be a point on L_1 inside P . Clearly $\triangle ABC$ is inscribed in P . See Fig. 4.6.1.

Also,

$$H \geq \frac{1}{2}W \geq \frac{1}{2}W(P)$$

Area of $\triangle ABC$ is bounded below by

$$|\triangle ABC| = \frac{1}{2}|AB|H \geq \frac{1}{4}L(P)W(P)$$

From now on the proof is very similar to the proof in Lemma 3.6.2. Since

$|\triangle ABC| \leq |T_{\max}|$, we get,

$$|T_{\max}| \geq \frac{1}{4}L(P)W(P) \dots \dots \dots (I)$$

Fig. 4.6.1: An inscribed triangle in a simple polygon whose ratio is at least 1/2 of the polygon's ratio.

Let $T_{\max}$ have base b and height h . From the definition of the ratio of a triangle,

$$\sigma \leq \frac{h}{b},$$

and our assumption in the beginning of this proof, we get

$$\begin{aligned} \frac{h}{b} &< \frac{\rho(P)}{2} \\ \Leftrightarrow h &< \frac{W(P)}{2L(P)}b \end{aligned}$$

It follows that,

$$|T_{\max}| = \frac{1}{2} hb < \frac{W(P)}{4L(P)} b^2$$

But b is at most the length of the polygon, which gives us the following upper bound on the area of $T_{\max}$, and a straight contradiction to (I).

$$|T_{\max}| < \frac{1}{4} W(P)L(P)$$

Hence, because of the above contradiction we can conclude that

$$\sigma \geq \frac{\rho(P)}{2}$$

Q.E.D.

Lemma: 4.6.2

Let T be an arbitrary triangle and let σ be its ratio. Given angles θ , α_I and β_I , let T_I be the maximal area triangle inscribed in T of similarity type $\Gamma(\alpha_I, \beta_I)$ whose longest side makes an angle of θ with the longest side of T . If σ_I denotes the ratio of T_I then it is claimed that

$$\frac{|T_I|}{|T|} \geq \left(\frac{1}{1 + \frac{\tan \theta}{\sigma}} \cdot \frac{1}{\cos \theta} \right)^2 \frac{\sigma_I}{\sigma}$$

Proof. See proof of Lemma 3.6.4. Q. E. D.

The algorithm for computing T_{alg} is the same as in the corresponding case of the convex polygon. See Section 3.6. Denote by T_{ijh} the maximal area axial fixed similarity triangle inscribed in P , whose base is along direction d_i , ratio is r_j , and the angle opposite to its second longest side is α_h . From Section 4.4.1, we know of an $O(n^2 \log n)$ algorithm that can compute T_{ijh} . As a matter of formality, the pseudo code description of the algorithm for computing T_{alg} is given below.

Algorithm: 4.6

INPUT: k, P

OUTPUT: T_{alg} .

```

1- For  $i \leftarrow 1$  to  $k$ 
     $r_i \leftarrow \frac{1}{k} i$ 
     $\alpha_i \leftarrow \frac{\pi}{2k} i$ 
     $\theta_i \leftarrow \frac{2\pi}{k} i$ 

2- For  $i \leftarrow 1$  to  $k$ 
    For  $j \leftarrow 1$  to  $k$ 
        For  $h \leftarrow 1$  to  $k$ 
            Compute  $T_{ijh}$ 

 $|T_{alg}| \leftarrow \text{MAX } \{ |T_{ijh}| : 1 \leq i, j, h \leq k \}$ 

```

Complexity

In the dominating Step (2), we execute the $O(n^2 \log n)$ algorithm for computing fixed similarity axial triangles a total of k^3 times. Therefore, the algorithm has an overall time complexity in $O(k^3 n^2 \log n)$ and space requirements in $O(n)$.

The next lemma provides the proof for the lower bound on the ratio of optimality for such a T_{alg} .

Lemma: 4.6.3

If T_{alg} is the output of algorithm 3.6 and T_{max} is the maximal area triangle in a given simple polygon P , then

$$\frac{|T_{alg}|}{|T_{max}|} \geq \left(\frac{k\rho(P)}{k\rho(P) + 8\pi} \right)^2 \left(1 - \frac{2}{k\rho(P)} \right)$$

Proof. See proof of Lemma 3.6.5.

Chapter Summary

This chapter presented asymptotic algorithms for solving the inclusion problems for squares, rectangles, equilateral triangles and arbitrary triangles in simple polygons. Our results compare with the best results from the literature. For instance, for the problem of finding the maximal area inscribed triangle in a simple polygon, we improved on the $O(n^4)$ algorithm of Melissaratos and Souvaine [Melissaratos and Souvaine, 1992] by giving an algorithm for this problem with a complexity of $O(k^3 n^2 \log n)$. See Table 4.1 for a complete summary of the results of this chapter.

The type of machinery used can be applied to several other types of regular convex polygons as well. For instance, algorithms for finding maximal area inscribed right triangle, quadrilateral and other regular convex polygons with greater number of sides, can also be developed using the proposed techniques.

INCLUSION ALGORITHMS FOR SIMPLE POLYGONS

APPROXIMATING CONVEX	COMPLEXITY	RATIO of OPTIMALITY
SQUARE	$O(k n^2 \log n)$	At least $\frac{1}{\left(1 + \tan\left(\frac{\pi}{2k}\right)\right)^2}$
EQUILATERAL TRIANGLE	$O(k n^2 \log n)$	At least $\frac{\tan^2\left(\frac{k-1}{k} \cdot \frac{\pi}{3}\right) + 1}{4}$
RECTANGLE	$O(k^2 n^2 \log n)$	At least $\left(\frac{k\rho(P)}{k\rho(P) + 8\pi}\right)^2 \left(1 - \frac{8}{k\rho(P)}\right)$
ARBITRARY TRIANGLE	$O(k^3 n^2 \log n)$	At least $\left(\frac{k\rho(P)}{k\rho(P) + 8\pi}\right)^2 \left(1 - \frac{2}{k\rho(P)}\right)$

Table 4.1: Results of Chapter 4.

Chapter 5

ENCLOSURE ALGORITHMS

This chapter discusses the algorithms for computing asymptotically optimal circumscribed square, equilateral triangle and rectangle on convex and simple polygons. Chang and Yap [Chang and Yap, 1986] mention that there is no general technique for converting an algorithm for an inclusion problem to its dual enclosure problem. However, at least for asymptotic algorithms, it is quite possible to transform the ideas for one algorithm to its associated dual one. Indeed, this chapter gives algorithms for the enclosure problems which have been transformed from the corresponding algorithms for the inclusion problems. The machinery of the algorithms presented in this chapter has been taken directly from the inclusion problems from Chapters 3 and 4.

It is easy to see that the minimal area convex polygon covering a simple polygon P , has the same area as the one that covers the convex hull of P . This indicates that we only need to consider enclosure problems for convex polygons. For simple polygons, we simply compute their convex hulls and use the corresponding algorithm from the convex polygons. D. McCallum and D. Avis [McCallum and Avis, 1979] have a linear algorithm for finding the convex hull of a simple polygon. We remark that our real contribution is in the sublinear algorithms for convex polygons. For simple polygons, these algorithms can also be used, however, at an additional cost of $O(n)$.

5.1 Circumscribed Square

The enclosure problem for the square is as follows. Given a convex polygon P , give an algorithm that finds the minimal area square S_{min} , circumscribing P . In simple terms, find the smallest area square that completely covers P . The result in this section is as follows. Given an arbitrary parameter k , representing the desired degree of accuracy, there is an algorithm which computes a square S_{alg} , covering P ,

$$\frac{|S_{alg}|}{|S_{min}|} \leq \left(1 + \tan\left(\frac{\pi}{2k}\right)\right)^2$$

The outline of the algorithm is identical to the algorithm for the inscribed square in Section 3.3. First consider the usual subproblem for finding the minimal area axial square covering P .

5.1.1 Minimal Area Axial Covering Square

Consider the following simpler problem. Given a convex polygon P and a direction d , find the minimal area square S_d that covers P and has one of its sides along d . Here, an $O(\log n)$ algorithm that solves this problem is presented.

First, consider the following operation on convex polygons.

Tangent finding:

Given a convex polygon P , and a direction L , give a point on P that is tangent to a line perpendicular to L .

Note that tangent finding in P gives us the farthest extent of P in some given direction. Described below is a logarithmic method for finding tangents.

Method for Tangent Finding:

It is easy to show that the farthest extent of P in any given direction is some vertex of P . The reason is that if a point interior to an edge is the farthest point, then by shifting the point right or left along the edge, one can always get a more distant point. Thus, we can use binary search over the cyclically ordered vertices of P and search for that vertex of P both of whose adjacent edges lie in the same half plane defined by a line perpendicular to L . Therefore, this can be accomplished in $O(\log n)$ time.

With the above operation available, the method for computing minimal area axial covering square follows immediately. We use the tangent finding operations to find vertices A and B , that are the farthest extends of P along and opposite to d , respectively. Similarly, we compute the extends in the direction perpendicular to d , say vertices C and D . Next, it is quite obvious that S_d has one side equal to the maximum of $|AB|$ and $|CD|$. Hence, S_d can be computed in $O(\log n)$ time. ♦

Lemma : 5.1.1

Given a square S and an angle θ , if S' is the minimal area square covering S , one of whose sides makes an angle of θ with the vertical side of S , then,

$$\frac{|S'|}{|S|} = \frac{(1 + \tan^2 \theta)^2}{1 + \tan^2 \theta}$$

Proof:

Note that if S' is the minimal area square covering S , then S is the maximal area square inscribed in S' . Therefore, interchanging the roles of S and S' , using Lemma 3.3.1, it can be concluded that,

$$\begin{aligned} \frac{|S|}{|S'|} &= \frac{1 + \tan^2 \theta}{(1 + \tan^2 \theta)^2} \\ \Leftrightarrow \frac{|S'|}{|S|} &= \frac{(1 + \tan^2 \theta)^2}{1 + \tan^2 \theta} \end{aligned}$$

Q.E.D

Algorithm Description

The algorithm is similar to the inclusion algorithm of Section 3.3.

Algorithm 5.1

INPUT: Convex polygon P , parameter k .

OUTPUT: S_{alg} .

1- Consider k directions $d_1, d_2, d_3 \dots d_k$. d_i makes angle $\pi/2k$ with x -axis

2- For $i = 1$ to k

$S_i \leftarrow$ axial minimal area covering square with one side along d_i .

$|S_{alg}| \leftarrow \text{MIN } \{|S_i|, 1 \leq i \leq k\}$

Lemma 5.1.2

If S_{alg} is the result of algorithm 5.1 then

$$\frac{|S_{alg}|}{|S_{min}|} \leq \left(1 + \tan\left(\frac{\pi}{2k}\right)\right)^2$$

Proof. The proof of this lemma is analogous to the proof of Lemma 3.3.2 (See Lemma 3.3.2). From the definition of the S_i squares in algorithm 5.1, there must exist a j , $1 \leq j \leq k$, such that one side of S_j makes an angle $\mu \leq \pi/2k$ with some side of S_{min} . Let S be the smallest square covering S_{min} whose side makes an angle of μ with some side of S_{min} .

From Lemma 5.1.1,

$$\frac{|S|}{|S_{min}|} = \frac{(1 + \tan \mu)^2}{1 + \tan^2 \mu}$$

From the algorithm 3.7.1, $|S_{alg}| \leq |S_j| \leq |S|$. Thus,

$$\frac{|S_{alg}|}{|S_{min}|} = \frac{(1 + \tan \mu)^2}{1 + \tan^2 \mu}$$

In Lemma 3.4.2, for $\mu \leq \pi/2k$, it was shown that

$$\frac{1 + \tan^2 \mu}{(1 + \tan \mu)^2} \geq \frac{1}{(1 + \tan(\frac{\pi}{2k}))^2}$$

This combined with the previous equation completes the proof.

$$\frac{|S_{alg}|}{|S_{min}|} \leq (1 + \tan(\frac{\pi}{2k}))^2$$

Q.E.D.

Complexity

The dominating step, Step 2, contains a loop which repeats k times, a task which is performed in logarithmic time. It follows that the overall time complexity for this enclosure problem for a convex polygon is in $O(k \log n)$. If the convex hull of a simple polygon is known in advance, our enclosure algorithm when applied on this convex hull gives the same complexity of $O(k \log n)$, otherwise, the cost of this enclosure problem for a simple polygon is in $O(n + k \log n)$ while the space requirements for both are $O(n)$.

5.2 Circumscribed Equilateral Triangle

The type of arguments used for converting the algorithm for the inclusion problem in Section 3.3 to the enclosure problem for squares in Section 5.1 are also valid here. Given a convex polygon P , the problem is to find the minimal area equilateral triangle that circumscribes P . This section gives an asymptotic algorithm that solves this problem.

Because the steps are essentially the same, we do not wish to bore the reader with the redundant details and skip right to the results. If T_{min} is the minimal area circumscribing equilateral triangle on P , and k is an arbitrary parameter, our algorithm produces a circumscribing equilateral triangle T_{alg} such that,

$$\frac{|T_{alg}|}{|T_{min}|} \leq \frac{4}{\tan^2\left(\frac{k-1}{k} \cdot \frac{\pi}{4}\right) + 1}$$

For a complete description of the algorithm, see Section 3.4 and Section 5.1. The overall time complexity for a convex polygon is in $O(k \log n)$ and $O(n + k \log n)$ for a simple polygon while the space requirements for both are $O(n)$.

5.3 Circumscribed Rectangle

This section gives an asymptotic algorithm for finding the minimal area rectangle, say, R_{min} , that covers the given convex polygon P . In time $O(k^2 \log n)$, our algorithm produces a rectangle R_{alg} such that,

$$\frac{|R_{alg}|}{|R_{min}|} \leq \left(1 + \frac{\pi}{2k\rho(P)}\right)^2$$

Observe that R_{alg} is always greater than R_{min} in area and this difference approaches zero as k is taken larger and larger.

Following the techniques proposed in this thesis, we first need an algorithm for finding the minimal area axial fixed ratio rectangle covering P which has given ratio r , and one of whose sides is along some given direction d . We can find such a rectangle using the *tangent finding* operations in Section 5.1.1. We simply find the extremes of P along direction d and along a line perpendicular to d . The desired rectangle is the intersection of the orthogonal stripes that are formed by taking parallel tangents passing through the respective extremes. All of this is done by a constant number of *tangent finding* operations, and the desired axial fixed ratio rectangle is thus found in $O(\log n)$ time.

Next, we need a result that provides bounds on the ratio of R_{min} in relation to the ratio of P .

Lemma: 5.3.1

If σ denotes the ratio of the minimal area rectangle R_{min} , circumscribing convex polygon P , then

$$\sigma \geq \rho(P)$$

Proof.

Fig 5.3.1: y is no more than $L(P)$, and x no less than $W(P)$, so x/y is no less than $\rho(P)$.

Suppose x and y are the width and length of R_{min} . It is easy to see that each side of R_{min} is adjacent to some point on the boundary of P . Among these four points, let X and Y be adjacent to the shorter sides of R_{min} , respectively. Furthermore, note that since $|XY|$ is a diagonal chord in R_{min} , its length must be more than y . Consequently, since $|XY| \leq L(P)$, it follows that

$$y \leq L(P)$$

From this lower bound on $L(P)$, we can conclude that,

$$\begin{aligned} \frac{x}{y} &\geq \frac{x}{L(P)} \\ \Leftrightarrow \sigma &\geq \frac{W(P)}{L(P)} \cdot \frac{x}{W(P)} \end{aligned}$$

Now x is greater than the width of P along the direction parallel to x . Therefore, x is also greater than the width of P .

$$\Leftrightarrow \sigma \geq \frac{W(P)}{L(P)} \cdot \frac{W(P)}{W(P)}$$

$$\Leftrightarrow \sigma \geq \rho(P)$$

Q.E.D.

Lemma 5.3.2

Let R be a rectangle with ratio r . Given a ratio r' , $0 < r' < 1$, and an angle $\theta \leq \pi/2$. If R' is the minimal area rectangle with ratio r' circumscribing R and whose longer side makes an angle of θ with the longer side of R , then it satisfies the following:

$$\frac{|R'|}{|R|} \leq \frac{r \left(1 + \frac{\theta}{r}\right)^2}{r'}$$

Proof. The proof follows from Lemma 3.5.2 by reversing the roles of R and R' . See Lemma 3.5.2.

The pseudo code description of the algorithm is provided below as a formality, while the reader is referred to the analogous inclusion problem of Section 3.5. Following the notation in Section 3.6, R_{ij} is the minimal area fixed axial rectangle with length along θ_i , and ratio r_j .

Algorithm 3.5.1: Compute R_{alg}

INPUT : P, k

OUTPUT: R_{alg}

```

1- For  $i \leftarrow 1$  to  $k$ 
     $\theta_i \leftarrow \frac{\pi}{k} i$ 
     $r_i \leftarrow \frac{1}{k} i$ 
2- For  $i \leftarrow 1$  to  $k$ 
```

For $j \leftarrow 1$ to k
 Compute R_{ij}
 $|R_{alg}| \leftarrow \min \{ |R_{ij}| : 1 \leq i, j \leq k \}$

Complexity

In Step 2, k^2 calls are made to the logarithmic algorithm for finding covering axial fixed ratio rectangles described at the beginning of this section. Therefore, the overall time complexities for convex and simple polygons is in $O(k^2 \log n)$ and $O(n + k^2 \log n)$, respectively. Space requirements are in $O(n)$.

Lemma: 5.3.3

If R_{alg} is the output of Algorithm 3.5.1, then

$$\frac{|R_{alg}|}{|R_{min}|} \leq \left(1 + \frac{\pi}{2k\rho(P)}\right)^2$$

Proof:

From the distribution of the r_i 's from the interval $[0, 1]$, there must exist a t , $1 \leq t \leq k$ such that

$$\begin{aligned} \sigma &\leq r_t \\ \Leftrightarrow \frac{\sigma}{r_t} &\leq 1, \dots \dots \dots (I) \end{aligned}$$

Furthermore, there must exist s , $1 \leq s \leq k$, such that the angle β , between the corresponding sides of R_{min} and R_{st} is less than $\pi/2k$. Let R be the minimal area circumscribed rectangle on R_{min} whose length makes an angle of β with the length of R and whose ratio is r_t . If σ is the ratio of R_{min} , then from Lemma 5.3.2,

$$\frac{|R|}{|R_{min}|} \leq \frac{\sigma \left(1 + \frac{\beta}{\sigma}\right)^2}{r_t}$$

From the algorithm,

$$|R_{alg}| \leq |R_{st}| \leq |R|$$

This implies that,

$$\frac{|R_{alg}|}{|R_{min}|} \leq \frac{\sigma \left(1 + \frac{\beta}{\sigma}\right)^2}{r_i} \leq \left(1 + \frac{\beta}{\sigma}\right)^2 \frac{\sigma}{r_i}$$

Since $\beta \leq \pi/2k$ and $\sigma \geq \rho(P)$ (Lemma 5.3.2), the above inequality becomes:

$$\frac{|R_{alg}|}{|R_{min}|} \leq \left(1 + \frac{\pi}{2k\rho(P)}\right)^2 \frac{\sigma}{r_i}$$

This inequality together with inequality (I) completes the proof.

$$\frac{|R_{alg}|}{|R_{min}|} \leq \left(1 + \frac{\pi}{2k\rho(P)}\right)^2$$

Q.E.D.

We remark that the above given ratio of optimality is identical for both a convex and simple polygon. The reason follows from the following fact. If C is the convex hull of our simple polygon P then, since the width of C is more than the width of P while they have identical lengths, it follows that $\rho(C) \geq \rho(P)$. Thus, for a simple polygon,

$$\frac{|R_{alg}|}{|R_{min}|} \leq \left(1 + \frac{\pi}{2k\rho(C)}\right)^2 \leq \left(1 + \frac{\pi}{2k\rho(P)}\right)^2$$

Chapter Summary

In this chapter, we presented some asymptotic algorithms for solving the enclosure problems for rectangles, squares and equilateral triangles on arbitrary polygons. For a fixed degree of accuracy, our algorithms had almost linear and sub linear complexities for simple and convex polygons, respectively. Thus, they prove to be suitable for many practical applications where the exact result is not imperative.

In this chapter, we also observed that there is a sublinear algorithm for computing minimal area enclosing object, similar to a given type, whose orientation is fixed. That is, if R is an m -gon then we can find the minimal area object, enclosing some given convex polygon, similar to R and whose orientation is fixed, in time $O(m \log n)$. In the final chapter of this thesis, we express our hopes for using this algorithm for finding the minimal area m -gon enclosing a given convex polygon.

The complete summary of the results in this chapter is given in Table 5.1.

ENCLOSURE ALGORITHMS

CIRCUMSCRIBING CONVEX	CONVEX POLYGON
SQUARE	$O(k \log n)$ Upper bound on the ratio of optimality: $\left(1 + \tan\left(\frac{\pi}{2k}\right)\right)^2$
EQUILATERAL TRIANGLE	$O(k \log n)$ Upper bound on the ratio of optimality: $\frac{4}{\tan^2\left(\frac{k-1}{k} \cdot \frac{\pi}{3}\right) + 1}$
RECTANGLE	$O(k^2 \log n)$ Upper bound on the ratio of optimality: $\left(1 + \frac{\pi}{2k\rho(P)}\right)^2$

Table 5.1: Major results of this thesis regarding the enclosure problem

Chapter 6

CONCLUSIONS

This final chapter summarizes the work presented in this thesis. For a detailed description of the results, the reader is referred to the actual algorithms in Chapters 2 through 5. Directions for future research associated with the inclusion and enclosure problems are also discussed.

6.1 Summary of the Thesis

The inclusion and enclosure problems allow one to approximate the area of polygons with regards to many applications of computational geometry. These problems are as follows. Given a polygon, we would like to find the maximal area object, similar to a given shape, that can be contained in the polygon, or the minimal area object that can cover it. The former is an *inclusion* and the latter an *enclosure* problem.

Practical fast approximation algorithms have a number of advantages over slower exact ones. This is especially true of a discrete environment such as a visual output device or a robotic vision. In such environments, there exists a degree of error beyond which the search for more accurate solutions is rendered unprofitable. If this threshold of error is known, fast algorithms such as the ones presented in this thesis can be efficiently utilized.

This thesis formulated the problem and discussed a new approach for developing efficient asymptotic algorithms for solving the inclusion and enclosure problems for convex and simple polygons. We gave asymptotic algorithms for computing

maximal area inscribed square, equilateral triangle, rectangle and arbitrary triangle in convex and simple polygons. We also presented much more efficient algorithms for solving the associated enclosure problems of computing maximal area circumscribed square, equilateral triangle and rectangles. A number of similar results were also obtained for some special polygons such as centrosymmetric, semicircular and dome polygons.

For approximating objects with at least one similarity parameter, the ratio of the polygons effected the ratio of optimalities of the algorithms. Specifically, the respective asymptotic algorithms had better performance for somewhat circular polygons than elongated ones.

See Tables 2.1, 3.1, 4.1 and 5.1 for a complete summary of the actual results of the algorithms presented in this thesis.

6.2 Directions for Future Research

This thesis opens the way to a number of related questions. Apart from the usual hope of improving on the complexities of the algorithms presented, there are perhaps more rewarding open questions. The natural generalization of the machinery presented in this thesis may be used to develop efficient asymptotic algorithms for a number of more interesting enclosure and inclusion problems. The generalization will focus, loosely speaking, on the following research:

- We need to relate the area of a convex polygon with a specially chosen convex polygon which is inscribed in it and has some fixed orientation.
- Find a relationship between the ratios of the optimally inscribed convex polygon and the ratio of the simple polygon whose area is being approximated.

- And finally, relate the area of the optimally inscribed convex polygon with the *algorithmic* convex polygon that our asymptotic algorithm will return.

If the above machinery is made precise, we might be able to make far more significant contributions than the ones presented in this thesis. Consider, for example, the problem of computing the minimal area circumscribed m -gon of a convex polygon. In the summary of Chapter 5, we mentioned the $O(m \log n)$ complexity of finding the minimal area enclosing m -gon with fixed similarity and orientation. Following the techniques proposed in this thesis, this subproblem may be used to develop an $O(k^m m \log n)$ algorithm for computing the minimal area asymptotically optimal enclosing m -gon. We urge the reader to compare this result with the $O(n^2 \log n \log m)$ algorithm for this problem by Chang and Yap [Chang and Yap, 1984], keeping in perspective that k is a constant for many applications with known noise levels. With the same token, if it is possible to compute a maximal area inscribed m -gon, in a simple polygon, with fixed similarity and orientation, with complexity in $O(m n^2 \log n)$, we might then have an $O(k^m m n^2 \log n)$ algorithm for computing the maximal area inscribed m -gon in a simple polygon. For fixed k and m , this asymptotic algorithm clearly is a significant improvement. Recall that for $m=3$, Melissaratos and Souvaine [Melissaratos and Souvaine, 1992] have the best algorithm with complexity in $O(n^4)$. So far as we are concerned, for an arbitrary m , there is no known exact algorithm for this problem. And finally, these techniques might very well be used to develop an efficient asymptotic algorithm for the potato peeling problem of Chang and Yap [Chang and Yap, 1986] mentioned in Section 1.3.

Comparatively much less work has been done on the corresponding problems in three dimensions. In the introduction to this thesis, we mentioned the $O(n^4)$ algorithm of O'Rourke [O'Rourke, 1984] for finding the minimal volume rectangular box enclosing a

polyhedron. Examples of the associated questions in three dimensions, for the inclusion and enclosure algorithms presented in this thesis, are as follows.

- We would like to know the maximal volume cube or tetrahedron that can be contained in a given convex body.
- What is the minimal volume cube that encloses a given polyhedron ?

The techniques proposed in this thesis might very well apply to these questions in three dimensions. However, certain analogous concepts need to be redefined. For instance, we need a parameter that contains geometric information of geometric objects in three dimensions. That is, we need to generalize our concept of *ratio* of a polygon to three dimensions.

REFERENCES

- Aggarwal, A., Chang, J. S. and Yap, C. K., (1985). "Minimum area circumscribing polygons", *Visual Computing*, **1**, pp. 112-117.
- Aggarwal, A. and Suri, S., (1987). "Fast algorithms for computing the largest empty rectangle", *Proceedings of the 13th ACM Symposium on Computational Geometry*, pp. 278-289.
- Boyce, J. E., Dobkin, D. P., Drysdale, R. L., and Guibas, L. J., (1985). "Finding extremal polygons", *SIAM Journal on Computing*, **14**, pp. 134-147.
- Chang, J. S. and Yap, C.K., (1984). "A polynomial solution for potato-peeling and other polygon inclusion and enclosure problem.", *Proceedings of the 25th Symposium on the Foundations of Computer Science*, pp. 408-416.
- Chang, J. S. and Yap, C.K., (1986). "A polynomial solution for the potato-peeling problem", *Discrete Computational Geometry*, **1**, pp. 155-182.
- Chazelle, B. and Guibas, L., (1989). "Visibility and Intersection Problems in Plane Geometry", *Discrete Computational Geometry*, **4**, pp. 551-581.
- Dehne, F., (1990). "Computing the Largest Empty Rectangle on One- and Two-Dimensional Processor Arrays", *Journal of Parallel and Distributed Computing*, **9**, pp. 63-68.
- DePano, N. A., Ke, Y. and O'Rourke, J., (1987a). "Finding Largest Inscribed Equilateral Triangles and Squares", *Proceedings of the Allerton Conference*, pp. 869-878.
- DePano, N. A., (1987b). "Polygon approximation with optimized polygonal enclosures: applications and algorithms", PhD Thesis, the John Hopkins University.

- Dobkin, D. and Snyder, L., (1979). "On a general method for maximizing and minimizing among certain geometric problems", *Proceedings of the 20th IEEE Symposium on Foundations of Computer Science*, pp. 9-17.
- Epstein, D., (1992). "New Algorithms for Minimum Area k -gons", in Proceedings of the 3rd Annual ACM Symposium on Discrete Algorithms, Orlando, Florida, pp. 82-88.
- Goodman, J. E., (1981). " On the largest convex polygon contained in a non-convex n -gon, or How to peel a potato", *Geometriae Dedicata* , **11**, 99-106.
- Guibas, L., Hershberger, J., Leven, D., Sharir, M., Tarjan, R., (1987). "Linear time algorithms for visibility and shortest path problems inside triangulated simple polygons", *Algorithmica*, **2** , pp. 209-233.
- Hart, S. and Sharir, M., (1984). "Non-linearity of Davenport-Schinzel sequences and of generalized path compression schemes", Technical Report 84-011, The Eskenasy Institute of Computer Sciences, Tel Aviv University.
- Klee, V. and Laskowski, M., (1985). "Finding the smallest triangles containing a given convex polygon ", *Journal of Algorithms* , **6**, pp. 359-375.
- Kranakis, E. and Rafique, E., (1992). "Algorithms for asymptotically optimal contained rectangles and triangles", Carleton University, School Of Computer Science Technical Report, TR-212.
- Lee, D. T., (1983). "Visibility of a simple polygon". *Computer Vision, Graphics and Image Processing*, **22**, pp. 207-221.
- McCallum, D. and Avis, D., (1979). "A linear algorithm for finding the convex hull of a simple polygon", *Information Processing Letters*, **9**, pp. 201-206.

- Mckenna, M., O'Rourke, J., and Suri, S., (1985). "Finding the largest rectangle in an orthogonal polygon", Technical Report JHU/EECS-85/09, Dept. of Elec. Eng. and Comp. Sci., The John Hopkins University.
- Melissaratos, E. A. and Souvaine, D. L., (1992). "On solving geometric optimization problems using shortest paths", *SIAM Journal on Computing*, **21**, No. 4, pp. 601-638.
- O'Rourke, J., (1984). "Finding minimal enclosing boxes", Technical Report, Dept. Elec. Eng. and Comp. Sci., The John Hopkins University.
- O'Rourke, J., Aggarwal, A., Maddila, S., and Baldwin, M., (1986). " An optimal algorithm for finding minimal enclosing triangles", *Journal of Algorithms*, **7**, pp.258-269.
- Preparata, F. P. and Shamos, M. I., (1990). "Computational Geometry : An Introduction", Texts and Monographs in Computer Science, Springer-Verlag.
- Thomas, R. H. K., (1979). "Pairs of convex bodies", *Journal of London Math. Soc.*, **19**, No. 2, pp. 144-146.
- Toussaint, G. T., (1980). "Pattern recognition and geometrical complexity", *Proceedings of the 5th International Conference on Pattern Recognition*, pp. 1324-1347.
- Wood, D. and Yap, C. K., (1985). "Computing a convex skull of an orthogonal polygon", *Proceedings of the Symposium on Computational Geometry*, Baltimore, Maryland, pp. 311-316.

List Of Lemmas

2.1.1	17
2.1.2	18
2.1.3	23
2.2.1	27
2.2.2	33
2.2.3	38
2.2.4	42
2.2.5	44
2.2.6	45
3.1.1	53
3.1.2	57
3.1.3	59
3.1.4	60
3.3.1	63
3.3.2	65
3.3.3	69
3.4.1	74
3.4.2	77
3.5.1	81
3.5.2	83
3.5.3	86
3.6.1	92
3.6.2	93
3.6.3	94
3.6.4	96
3.6.5	99
4.1.1	106
4.2.1	107
4.3.1	126
4.4.1	130

4.5.1	136
4.6.1	142
4.6.2	144
4.6.3	145
5.1.1	150
5.1.2	151
5.3.1	154
5.3.2	155
5.3.3	156

List Of Tables

Table 1.1

A comparison of the results of this thesis against those in the literature. 12

Table 1.2

Major results of this thesis regarding the inclusion problem 13

Table 1.3

Major results of this thesis regarding the enclosure problem 14

Table 2.1

Summary of the results for the inclusion problem for special polygons. 51

Table 3.1

Summary of results for inclusion algorithms presented in Chapter 3. 103

Table 4.1

Results of Chapter 4. 147

Table 5.1

Major results of this thesis regarding the enclosure problem 159

END

14-12-93

FIN