Helsinki University of Technology   Automation Technology Laboratory
Series A: Research Reports No. 31
Espoo, December 2007

PATH PLANNING ALGORITHMS FOR AGRICULTURAL FIELD MACHINES

Timo Oksanen

TEKNILLINEN KORKEAKOULU
TEKNISKA HÖGSKOLAN
HELSINKI UNIVERSITY OF TECHNOLOGY
TECHNISCHE UNIVERSITÄT HELSINKI
UNIVERSITE DE TECHNOLOGIE D'HELSINKI

Helsinki University of Technology Automation Technology Laboratory
Series A: Research Reports No. 31
Espoo, December 2007

PATH PLANNING ALGORITHMS FOR AGRICULTURAL FIELD MACHINES

Timo Oksanen

Dissertation for the degree of Doctor of Science in Technology to be presented with due permission of the Department of Automation and Systems Technology, for public examination and debate in Auditorium AS1 at Helsinki University of Technology (Espoo, Finland) on the 14th of December, 2007, at 12 o´clock noon.

Helsinki University of Technology
Department of Automation and Systems Technology
Automation Technology Laboratory

Teknillinen korkeakoulu
Automaatio- ja systeemitekniikan osasto
Automaatiotekniikan laboratorio

Distribution:
Helsinki University of Technology
Automation Technology Laboratory
P.O. Box 5500
FIN-02015 TKK
FINLAND

Tel. +358 9 451 5562
Fax +358 9 451 3308
E-mail: timo.oksanen@tkk.fi

ISBN 978-951-22-9079-6 (print)
ISBN 978-951-22-9080-2 (pdf)
ISSN 0783-5477

Picaset Oy
Helsinki 2007

ABSTRACT OF DOCTORAL DISSERTATION	HELSINKI UNIVERSITY OF TECHNOLOGY P.O. BOX 1000, FI-02015 TKK http://www.tkk.fi
Author Timo Oksanen	
Name of the dissertation Path Planning Algorithms for Agricultural Field Machines	
Manuscript submitted 23.8.2007	Manuscript revised
Date of the defence 14.12.2007	
☒ Monograph	☐ Article dissertation (summary + original articles)

Department	Automation and Systems Technology
Laboratory	Automation Technology Laboratory
Field of research	Automation technology
Opponent(s)	Associate Research Professor Alonzo Kelly
Supervisor	Professor Arto Visala
Instructor	

Abstract

In this thesis, a coverage path planning problem is discussed in the case of agricultural fields and agricultural machines. Methods and algorithms to solve this problem are developed. The necessary condition is to cover the whole field, while the goal is to find as efficient a route as possible. As yet, there is no universal algorithm or method capable of solving the problem in all cases.

In this thesis, two new approaches to solve the coverage path planning problem in the case of agricultural fields and agricultural machines are presented. In the first algorithm, the field plot is split into subfields that are simple to drive. Each subfield is constructed from trapezoids, the sides of which are attached. Thus the opposite sides of each subfield are parallel. In the search method, the best driving direction for each subfield is found. In the second algorithm, the path is planned on the basis of the machine's current state. Inspired by model predictive control, all the possible routes are simulated over the horizon of one route around the field or one back-and-forth swath. The best of the simulated routes is selected and the first part of that route is applied, up to the next turning. In the next turning, the algorithm is repeated. There are advantages and drawbacks in both algorithms. Neither of them solves the universal problem optimally. Nevertheless, the developed algorithms are remarkable steps towards finding a way to solve the problem.

As a side-result, the properties of Finnish field plots are discussed and the field plot shape analysis is made. The classification of field plots based on shape was investigated, but only 25% of field plots belong to one or other simple-shape classes, such as rectangles and triangles, while the rest are more or less undetermined shapes.

The other side-result is a trajectory generation for a tractor-trailer vehicle in a headland. The problem is formulated as an optimal control problem and solved using available methods and tools. With this approach, the trajectory can be solved for any headland width and angle.

Keywords coverage path planning, agricultural machines, agricultural fields, shape analysis, optimal control, algorithms

ISBN (printed) 978-951-22-9079-6	ISSN (printed) 0783-5477
ISBN (pdf) 978-951-22-9080-2	ISSN (pdf)
Language English	Number of pages 110
Publisher Helsinki University of Technology, Automation Technology Laboratory	
Print distribution Helsinki University of Technology, Automation Technology Laboratory	
☒ The dissertation can be read at http://lib.tkk.fi/Diss/	

VÄITÖSKIRJAN TIIVISTELMÄ	TEKNILLINEN KORKEAKOULU PL 1000, 02015 TKK http://www.tkk.fi

Tekijä Timo Oksanen

Väitöskirjan nimi
Peltotyökoneiden reitinsuunnittelumenetelmät

Käsikirjoituksen päivämäärä 23.8.2007	Korjatun käsikirjoituksen päivämäärä
Väitöstilaisuuden ajankohta 14.12.2007	
☒ Monografia	☐ Yhdistelmäväitöskirja (yhteenveto + erillisartikkelit)

Osasto	Automaatio- ja systeemitekniikan osasto
Laboratorio	Automaatiotekniikan laboratorio
Tutkimusala	Automaatiotekniikka
Vastaväittäjä(t)	Associate Research Professor Alonzo Kelly
Työn valvoja	Professori Arto Visala
Työn ohjaaja	

Tiivistelmä

Tässä väitöskirjassa käsitellään peltoliikenteen reitinsuunnitteluun liittyviä ongelmia ja kehitetään menetelmiä ja työkaluja ongelmien ratkaisemiseen. Reitinsuunnittelun ehdottomana vaatimuksena on kattaa koko pelto ja pyrkiä suunnittelemaan reitti mahdollisimman tehokkaaksi. Vielä ei tunneta mitään yleistä menetelmää jolla reitinsuunnittelu voitaisiin ratkaista optimaalisesti kaikissa tapauksissa.

Tässä väitöskirjassa esitetään kaksi uutta lähestymistapaa ratkaisemaan peltoliikenteen reitinsuunnittelu. Ensimmäisessä menetelmässä pilkotaan hankalan muotoinen lohko useampaan osalohkoon, jotka erillisinä on helppo ajaa. Osalohkon muoto koostuu kyljistään yhteenliitetyistä puolisuunnikkaista, ja siten osalohkon vastakkaiset sivut ovat myös samansuuntaiset. Hakumenetelmässä etsitään kullekin osalohkolle paras ajosuunta. Toisessa menetelmässä suunnitellaan reittiä koneen sen hetkisestä tilanteesta lähtien simuloimalla kaikki mahdolliset reitit eteenpäin yhden ajokierroksen tai yhden käännöksen ajan. Tässä mallipredikiivisen säädön innostamassa lähestymistavassa simuloiduista reiteistä valitaan paras reitti ja siitä suoritetaan ensimmäinen yhtenäinen ajolinja, seuraavaan käännökseen saakka ja siinä kohtaa suoritetaan taas sama uudelleen. Kummassakin kehitetyssä lähestymistavassa on hyviä ja huonoja puolia ja kumpikaan ei optimaalisesti ratkaise yleistä ongelmaa. Kuitenkin esitetyt menetelmät ovat selkeä edistysaskel ongelman ratkaisun löytämiseksi.

Sivutuloksena väitöskirjassa käsitellään suomalaisten peltolohkojen ominaisuuksia ja analysoidaan erityisesti peltolohkojen muotoja. Peltolohkot pyrittiin luokittelemaan muodon perusteella, mutta tuloksena vain 25 % peltolohkoista kuului johonkin yksinkertaiseen muotoluokkaan ja loput ovat enemmän tai vähemmän epämääräisiä muotoja.

Toisena sivutuloksena, reitinsuunnittelualgoritmia varten, on ratkaistu optimaalinen traktori-työkoneyhdistelmän kääntämisreitti yhdessä päisteessä: ongelma on kirjoitettu optimisäätöongelman muotoon ja optimisäätöongelma on ratkaistu olemassa olevien työkalujen avulla ja tuloksena on saatu käännösreitti eri päistekulmien ja -leveyksien tapauksessa.

Asiasanat reitinsuunnittelu, maatalouskoneet, peltoviljely, pellot, muotoanalyysi, optimisäätö, algoritmit

ISBN (painettu) 978-951-22-9079-6	ISSN (painettu) 0783-5477
ISBN (pdf) 978-951-22-9080-2	ISSN (pdf)
Kieli Englanti	Sivumäärä 110

Julkaisija Teknillinen korkeakoulu, Automaatiotekniikan laboratorio

Painetun väitöskirjan jakelu Teknillinen korkeakoulu, Automaatiotekniikan laboratorio

☒ Luettavissa verkossa osoitteessa http://lib.tkk.fi/Diss/

Preface

I think that the first time I met the problem of coverage path planning for fields happened when I was about 10 years old. Probably the tractor was a Fiat 680 DT equipped with a harrow and the field was located next to our farmhouse. Very approximate cellular decomposition was solved by my father, delivered to me as verbal instructions and waving hands; I was a lower-level navigator and a steering actuator. Since then I have learned to use clutch and gears, and to operate and tune more demanding implements and to solve the planning problem by myself, at a very practical level.

The chance to solve the same problem mathematically seemed a fascinating opportunity and challenge; I consequently selected it as my topic for postgraduate research. At that time, I had no idea just how challenging the problem would be to solve, and I felt that I was not the only one.

This thesis is the result of research work carried out in the Automation Technology Laboratory at Helsinki University of Technology during the period 2003-2007. The research was funded by the *Graduate School of Electronics, Telecommunications and Automation* (GETA). My thanks go to the laboratory and to Professors Aarne Halme and Arto Visala for the opportunity they gave me to undertake my research in this environment. Thanks also go to GETA for funding, and to *Tekniikan edistämissäätiö* and *Emil Aaltosen säätiö* for their grants.

This research was related to the Agrix project (Automation System for Agricultural Implements, 2003–2005). I thank all those who were involved in this project (Matti, Mikko, Markus, Pasi, Liisa, Jere, Hannu, Seppo, Jukka, Sampsa, Henrik and numerous others). Your great work in the project gave me time to concentrate on this topic.

Special thanks go to my colleagues and friends Vesa Hölttä, Lasse Eriksson and Pasi Riihimäki. Without our academic discussions and happy moments, this thesis could have been a mission impossible.

Contents

Abstract 3

Tiivistelmä 5

Preface 7

Contents 8

Symbols, abbreviations and definitions 10

1 Introduction 13

1.1 Background and motivation 13
1.2 Overview of contents 15
1.3 Scientific contribution of the study 16
1.4 Author's role in research 17
1.5 Relation to earlier publications 17

2 Field analysis 19

2.1 Introduction 19
2.2 Literature review 20
2.3 Definitions 21
2.4 Shape indexes 21
2.4.1 Background 21
2.4.2 Convexity 21
2.4.3 Compactness 22
2.4.4 Rectangularity 22
2.4.5 Moments 22
2.4.6 Triangularity 23
2.4.7 Ellipticity 23
2.4.8 Ratio of principal moments 24
2.5 Results 24
2.6 Classification 29
2.6.1 Approach 29
2.6.2 Thresholding and conjunction 30
2.6.3 Self-organizing maps 33
2.7 Very hard field shapes for path planning 36
2.8 Conclusion 39

3 Headland turning 41

3.1 Introduction 41
3.2 Modeling of vehicle 42
3.3 Optimal control problem 44
3.4 Tractor-trailer in headlands 46
3.5 Effect of angle and width of headlands 48
3.6 Path approximation 49
3.7 Conclusion 50

4 Split and merge approach 53

4.1 Introduction 53

4.2 Literature review on path planning 54
4.2.1 Traditional path planning 54
4.2.2 Coverage path planning 54
4.2.3 Agricultural path planning 56
4.3 Planning task 58
4.3.1 Background 58
4.4 Algorithm 58
4.4.1 Objectives and definitions 58
4.4.2 Splitting 59
4.4.3 Merging 60
4.4.4 Selection criterion 60
4.4.5 Searching driving direction 61
4.4.6 Headlands 61
4.4.7 Prohibited driving directions 62
4.5 Test results 62
4.5.1 Tests with real fields 62
4.5.2 Headlands 65
4.5.3 Prohibited regions 67
4.6 Conclusion 68

5 Predictive recursive online approach 71

5.1 Introduction 71
5.2 Definitions 72
5.3 Algorithm 72
5.3.1 Assumptions and limitations 72
5.3.2 Polygon offsetting 72
5.3.3 Basic idea 73
5.3.4 Generating routes 74
5.3.5 Efficiency function 74
5.3.6 Selection of best route in search horizon 75
5.3.7 Simulated driving 75
5.3.8 Simplifying routes 75
5.3.9 Refilling and emptying of machine 76
5.3.10 Non-convex fields 78
5.3.11 Summary 79
5.4 Test results 79
5.4.1 Notes 79
5.4.2 Simple fields 80
5.4.3 Fields with curved edges 84
5.4.4 Fields with obstacles 86
5.4.5 Servicing 87
5.4.6 Complete examples 90
5.4.7 About computational speed of algorithm 91
5.5 Conclusion 92

6 Discussion 93

7 Conclusions 97

References 100

Appendix A: Complete examples 105

Symbols, abbreviations and definitions

Symbols

A	area
a	dimension of tractor-trailer system
a_R	acceleration of tractor in tractor-trailer system
b	dimension of tractor-trailer system
$C(x,u,t)$	path constraint function in optimal control problem
c	dimension of tractor-trailer system
d	dimension of tractor-trailer system
$E\left(x(t_0),x(t_f)\right)$	boundary condition function in optimal control problem
E_I	ellipticity index
$F(x,u,t)$	cost function in optimal control problem
$f(x,u,t)$	dynamics function
I_1	first affine moment invariant
$J(u)$	cost functional in optimal control problem
J_{route}	time efficiency of route
P	perimeter
P_A, P_B, P_C, P_D	points in Bezier curve
m_{pq}	moment of order p and q
r	radius of circle
s_W	total length of working in route
s_T	total length of turnings in route
$T,\ T_1 ... T_N$	service point(s) for machine in field
T_M	triangularity index
t_0, t_f	initial time and final time in optimal control problem
t_W	time consumed in working in route
t_T	time consumed in turnings in route
u	input of dynamical system
v_H	velocity of hook in tractor-trailer system
v_R	velocity of tractor in tractor-trailer system
w_h	headland width
x	state of dynamical system
(x_H, y_H)	position of hook in tractor-trailer system
(x_R, y_R)	position of rear axle in tractor-trailer system
α_F	steering angle in tractor-trailer system
β	auxiliary variable in tractor-trailer system

β_h	headland angle
θ	heading angle in tractor-trailer system
μ_{pq}	central moment of order p and q
φ	drawbar angle in tractor-trailer system
ω_F	angular velocity of steering angle of front wheels in tractor-trailer system

Abbreviations

CW	Clockwise
CCW	Counter clockwise
DOF	Degrees of Freedom
MBR	Minimum Bounding Rectangle
MPC	Model Predictive Control
NLP	Nonlinear Programming
OCP	Optimal Control Problem
SOM	Self-organizing map

Definitions

• *block*	polygon that is constructed by merging the parallel and equal sides of two or more trapezoids
• *central moment*	moment relative to expectation value
• *convex hull*	minimum convex polygon that encapsulates a region
• *critical vertex*	vertex in which the machine has to make turning
• *edge*	line segment in closed polyline
• *exterior polygon*	describes field outer boundary
• *field*	synonym for field plot, vernacular term
• *field plot*	uniform 2D area which is used for agricultural purpose, governmental term
• *headland*	part of a field where most of turnings are made, at the edge of the field
• *interior polygon*	describes field inner boundary, obstacles
• *moment*	here: statistical measure for uniform region
• *polygon*	closed polyline
• *polyline*	continuous line composed of one or more line segments
• *region*	uniform 2D area defined by one exterior polygon and $[0,\infty[$ interior polygons
• *swath*	continuous period of operation in field without lifting or switching off the implement
• *vertex (pl. vertices)*	point in 2D space where polyline segments are connected

1 Introduction

1.1 Background and motivation

In this thesis, the application area is *agricultural machines* in *agricultural field plots*. Several operations are required during the season in crop farming. Depending on the crop farming culture, the common operations are: cultivation, seeding, fertilizing, spreading chemicals (herbicide/fungicide/pesticide) and harvesting. A *tractor* is a general-purpose power machine that has an associated set of tools available for each operation. These tools, which can be connected to the tractor, are called *implements* in an agricultural framework. *Self-propelled agricultural machines* do not require a tractor.

Traditionally, little thought has been given to the path planning of field operations. Usually, the way to drive the field plots has developed over seasons by experience. This knowledge has been passed from generation to generation. However, there is no evidence that these *ad hoc* routes are optimal or near optimal. The optimality or the efficiency of a route affects the energy consumption needed to operate the field. Recently, the energy conservation issues have been widely discussed because of climate change and other environmental changes seem to be real. Commonly it is believed that there is room for optimization, and this was one of the original driving forces for this research.

The Finnish farm structure has been undergoing change since the beginning of 1990's. Actually, it has been under change all the way from mechanization of farming, but during the last 15 years the change has been remarkable. The total area of fields per farmer has been increasing and farmers cultivate field plots new to them, and therefore previous experience relevant to some field plots does not exist. Some agricultural experts have predicted that over as few as 15 years that *contracting* of field operations will be common, but it seems that only during the last few years this has started to become true. Each contractor will execute certain field operations for a large number of unknown field plots. Experienced contractors may see a good way to drive a field plot by taking a glance at a map and seeing the landscape. However, some contractors may not have such a good eye for good path planning. This gives rise to a clear need for computer-assisted path planning.

In certain operations, path planning involves more than just finding a path that covers the whole field. In most operations, some input is delivered to the field or some output is harvested from the field. The machines cannot carry an infinite amount of input or output, so the machines have to be refilled or emptied regularly. This part of operation should also be considered in path planning.

Lately so called *auto-guidance devices* have become more common in agricultural field machines. An auto-guidance device is installed on the tractor or on the self-propelled machine and it either automatically (electro-hydraulic steering) or semi-automatically (light bar to show steering request) keeps the vehicle on-lane based on accurate satellite positioning. Support for straight driving lines is available in all of them, but perfect support for curved lines or headland driving is rare. The path planning concept in these path-tracking devices is to copy the first (or the previous) realized path by the working width of the implement (or multiple of that) sideways. Especially in Finland, the shape of a field plot tends to be so complex that using only this simple path planning algorithm developed for a rectangular field plot is not giving satisfactory results. It seems that there are wishes to integrate more advanced path planning for these onboard devices and develop functionality for headland driving.

Some scientists and researchers believe in *agricultural field robots* for the future. These *unmanned ground vehicles* would be autonomous machines without a driver and they would work in the fields. Some field robot prototypes are developed around the world in universities and research institutes, but it can be said that there are no commercial field robots, yet. For them to become common in farms would require development in technical, economical, political and legislative domains and it is difficult to see when that would happen.

However, these future robots would execute some field operations autonomously. The whole crop production would change from averaged operations planned with common sense and gut feelings to individual plant caring and selective harvesting – generally known as *phytotechnology*. It is clear that, if unmanned field machines are executing the field operations, the path planning of those operations must be automatized. Hence, this is another motivation for this research.

1.2 Overview of contents

This thesis contains four parts: field shape analysis (Chapter 2), headland turning (Chapter 3), top-down approach to solve coverage path planning problem for fields (Chapter 4) and bottom-up approach to solve the coverage path planning problem for fields (Chapter 5).

The literature review is divided into chapters. The literature review for the field shape analysis can be found in Chapter 2.2, while the review for the main problem (coverage path planning for fields) can be found in Chapter 4.2.

At the beginning of this research, it was thought that, if the field machine was holonomic or omni-directional (can be turned in one place), the path planning would be rather easy; since there exist *coverage path planning algorithms* that can solve the problem for the holonomic machine or mobile robot. However, almost all the current crop farming machines are far from holonomic. Turning a vehicle takes time and this makes coverage path planning much harder. It is well known that the turning of a vehicle is crucial when searching for more optimal solutions. Solutions can take two forms: minimizing the number of turns or minimizing the time consumed in a single turning. The turning area at the edge of the field plot is called *headland*. On the other hand, the field plots in Finland are believed to be quite small and the shape of field plots is believed to be complicated compared to countries where crop farming is carried out in large open plains. Before going deeply into the coverage path planning problem, the environment is analyzed and a single turning of a machine is studied.

The coverage path planning with a non-holonomic machine is known to be a very hard problem to solve optimally. Furthermore, it seems that, currently, there is no universal algorithm to solve it. The scope of this study is therefore to develop new algorithm(s) that can find good and feasible solutions. Absolute optimality of possible solutions is not a goal.

In Chapter 2, real field plots in Finland are analyzed. The data set is from the Uusimaa region and it contains about 65000 field plots. Six different shape-describing indexes are calculated from field boundary lines, considering field plots as polygons. Based on the shape indexes, an attempt is made to try to classify the field plots.

In Chapter 3, a single headland turning is optimized by formulating the turning as an optimal control problem. The trajectory is a functional to be researched; the field boundaries are set as constraints. In addition, the kinematic constraints of the vehicle in question are considered. The optimal control problem is solved by utilizing existing methods and tools.

In Chapter 4, the approach to solve the path planning problem is top down: the complex-shaped field plot is divided automatically into blocks in which driving is easy. The most efficient block is researched in the field and is then removed. These two steps of the algorithm are repeated until the whole field is split.

In Chapter 5, the approach to solve the path planning problem is bottom-up: the driving is started by following some edge of the field and after each swath all the possible routes are simulated over a certain prediction horizon and the best of these is applied. This is continued until the whole field is covered by the operation.

1.3 Scientific contribution of the study

Path planning for agricultural machines is quite a fresh field of research, which has not been studied widely. The main objective of this research was to develop algorithms and tools that do the coverage path planning for agricultural fields and give suitable solutions. The absolute optimality of the solutions has not been an objective as it is practically impossible to prove that a route is optimal. Field-shape analysis and headland turning are side-results, which were performed in order to understand the main problem better.

The following features are believed to be original:

1. Applying shape analysis methods to field plots
2. Classification of Finnish field plots based on shape
3. Formulation of turning of an agricultural tractor-trailer system in headland as an optimal control problem and solving it using known methods in various cases
4. Developing a new algorithm to split field into subfields
    a. the idea itself
    b. the main contribution in the split and merge approach is the integration of the driving direction search method into exact cellular decomposition
    c. integrating regions with prohibited driving directions into the algorithm, adapting practical aspects to path planning
    d. automatic laying of headlands
5. Developing a new algorithm to coverage path planning for fields
    a. the idea itself
    b. utilizing the underlying idea of model predictive control in the case of recursive online coverage path planning
    c. definition of search horizon
    d. including refilling and emptying of field machine containers to the algorithm

1.4 Author's role in research

The author of this thesis has performed the research, development, analysis and reporting of this research, and drawn conclusions as to possible innovations and contributions, all by himself – except for those programming parts for the algorithm presented in Chapter 4. For that algorithm research, assistant Sampsa Kosonen carried out the implementation of automatic headland laying and support for fields with obstacles, developed the handling of many special cases that appeared in tests of the algorithm, improved the computational efficiency and fixed plenty of bugs found in the code during the period 2004-2005.

1.5 Relation to earlier publications

The material in Chapter 2 has not been published earlier. The main content of Chapter 3 has been published earlier in (Oksanen and Visala 2004). The main content of Chapter 4 has been published earlier in (Oksanen, Kosonen and Visala 2005, Oksanen and Visala 2006a, Oksanen and Visala 2007). The main idea of Chapter 5 has been published earlier in (Oksanen and Visala 2006b, Oksanen and Visala 2007), but Chapter 5 contains also unreleased features and new results.

2 Field analysis

2.1 Introduction

In some regions of the world, the path planning may be very simple as the fields are always rectangular and there are no natural obstacles. It is common knowledge that, in Finland, field plots are usually troublesome to operate because their shape varies a lot. A farmer would be very happy if the field plot were rectangular.

In some areas of Finland, the fields are nicely shaped, as the cultivated areas are large and open. Where this is so, it has been possible to lay ditches so that the fields are more-or-less rectangular. Nevertheless, in some other areas of Finland, the fields are bounded by nature, i.e. by rocks, lakes, rivers or forests or some other natural obstacles. When nature bounds the fields, the shape can be anything.

In this chapter, shape and other statistics of Finnish field plots are analyzed. All the Finnish field plots were scanned in the 1990's. Information from this source is mainly used for government policies. In this research, it has been possible to use the data relating to boundaries of real fields.

The original idea was to first analyze and develop the algorithms using field data in Uusimaa county (65000 fields) and later extend the analysis to all Finnish fields (one million). Unfortunately, authorities in *Maa- ja metsätalousministeriön tietopalvelukeskus* (MMM/Tike) outpriced the data of all Finnish field plots and it was not possible to follow the plan. Thus, the study concentrates only on fields in Uusimaa region; the idea of analyzing shape variation around the country was dropped.

The goals of this chapter are to: try to better understand path planning problem demands, try to analyze field-plot shapes, try to classify the fields into some sets based on shape if possible, and find typical field plots for testing the path planning algorithm.

2.2 Literature review

Unfortunately or fortunately, no international research related to the topic "field shape analysis" was found. The reason for this may be the fact that, in many countries, the field-plot shapes are so simple, i.e. rectangular, that there is no need for shape analysis.

Three notable research projects carried out in Finland were found.

Myyrä (2001) has analyzed field shapes in order to study the economic impact of farm structure. In that research, the field shape was characterized using the ratio of perimeter (P) and area (A) of field plot, P/A (used unit: m/ha). It was said that this ratio is compared to the ratio of square field plots, and that, the higher the index is, the more negative the impact on economic results is. In the research, the index variation was compared over the country and the conclusion was that the worst value is in Pirkanmaa region and the best in Etelä-Pohjanmaa region. Uusimaa region (where the data set in this research is from) was ranked as 11/15 for efficiency. This index is area dependent and does not only describe the shape but also the size of a field plot.

The other research conducted in Finland (Klemola *et al.* 2002) discussed the field-shape impacts at a very practical level. In the research, the field shape was considered from the time efficiency point of view, and time consumed in a field in practice was measured in different scenarios. In that research, it was concluded that "there are too many parameters (machine, driving techniques, driver's preferences, field conditions) that affect the driving and turning times, so that deterministic model for shape effect to times could be developed". Further, "The shape of a field has the greater effect on small fields. As the field size increases, the shape's effect decreases". In addition: "The effect the field size and shape have on the time spent working in the fields is almost entirely related to turning times", which means that the turning times and the shape and the size of a field are highly correlated.

The third Finnish research study where field-plot shape is discussed was conducted by Peltola *et al.* (2006). In that research, reasons for agricultural land value were studied. One of the properties of land that was analyzed in the study was the shape of a field plot. Also in that research, perimeter and area of field plot were used to form an index, but there it was developed as area independent, $A/(0.25 \times P)^2$. It was concluded that "this index gives only a rough and biased impression of the effect of shape". Later on in that research, a manual survey was used to classify about 2800 fields into shape classes: L-shaped, undeterminable, truncated triangle, triangle, trapezoid and rectangle. The realized selling prices for each class were analyzed and some differences were noticed, but no conclusions were drawn.

2.3 Definitions

In this thesis, a field plot is considered as a 2D uniform region that may contain holes. The edges of all Finnish agricultural field plots were digitized over 10 years ago and redigitized since. *Field* is a general term for area where crops and other agricultural plants are cultivated and *field plot (suom. peruslohko)* is a government term referring to a precise region.

Polygon denotes a closed path (in 2D) that consists of straight-line segments or edges. The points where edges end are called *vertices* or *corners*. *Edge* refers to a straight line between two vertices. A *simple polygon* means a polygon, the edges of which do not intersect with each other. *Region* is term for a set of simple polygons which describes a 2D space with holes, one simple polygon describes the outer edges and $[0,\infty[$ simple polygons describe the holes. An outer-edge polygon is called an *exterior polygon* and polygons describing holes are called *interior polygons*. A field plot is a region. The base unit for area in the agricultural context is a *hectare* (ha, 1 ha = 10000 m^2).

2.4 Shape indexes

2.4.1 Background

Shape analysis belongs to the field of computational geometry and its main uses seem to be in machine vision, image analysis and computer graphics. Many of the shape analysis algorithms are available for bitmap inputs, but also for polygon inputs. Here the field plots are represented as polygons, so it is natural to use polygon-type shape index algorithms instead of rasterizing the polygons and use bitmap input algorithms. Some of the following shape indexes are so well known that references are not needed.

2.4.2 Convexity

If the polygon is convex, all pairs of points inside or at the edge of the polygon can be connected with a straight line without crossing edges of the polygon. *Convex hull* is the smallest convex polygon that includes a set of points. In the case of polygons, convex hull is the smallest convex polygon that includes all vertices of a polygon. If the polygon is convex, the convex hull is equal to the polygon.

To measure the convexity of region, a convex hull is calculated from the exterior polygon and the ratio of areas of region and the convex hull describes the convexity. The value range is [0,1].

2.4.3 Compactness

Compactness is a shape index that describes how circular the polygon is. The compactness can be calculated using formula (1).

$$Compactness = 4\pi \frac{A}{P^2} \tag{1}$$

The value of compactness is largest for a circle and is equal to one. Thus the range of values is [0,1].

2.4.4 Rectangularity

Rectangularity is a shape index important in describing agricultural fields. Rectangular fields are easy to operate, as all the swaths can be straight and laid in parallel if no additional requirement exists.

In this research, a standard MBR (minimum bounding rectangle) index is used. The *rotating caliper* method (Toussaint 1983) is used to find the minimum bounding rectangle. In practice, this means that one side of the minimum bounding rectangle is collinear to one of the edges of the convex hull of the polygon and that the MBR can be solved in linear time. The index is the ratio of areas of original region and MBR, the value is in range [0,1].

2.4.5 Moments

Moments for polygons are used in physics, for example. For a homogenous 2D polygon, the *moment* of order *p* and *q* can be calculated by integrating over the polygon interior using formula (2).

$$m_{pq} = \iint x^p y^q \, dxdy \tag{2}$$

Moments can be calculated with respect to any point, but are usually calculated with respect to the origin. *Central moments* μ_{pq} are calculated in relation to the expectation value or, in a physical sense, with respect to the centre of mass. For central moments, the first moments (μ_{10}, μ_{01}) are always zero. The second moment is usually called *variance*. Usually higher-order moments are combined, for example *skewness* is calculated from third-order moments. Moments are not used directly in this thesis to describe shape, but they are needed in the following indexes.

2.4.6 Triangularity

To measure triangularity, several algorithms have been developed. Rosin (2003) reviews and compares several algorithms: moment invariant based, polygonal triangle approximation, projection based, minimum bounding triangle and moment matching based. On the basis of the conclusions of that comparison, the moment-matching algorithm was selected to be applied in this research. This algorithm is the most robust for aspect ratio changes and noise.

In moment matching, all the moments up to order 4 are calculated for a polygon. The prototype triangle has properties $m_{10}=m_{01}=0$. The moment-matching index describes the difference in moments,

$$T_M = \frac{1}{1+\sum\limits_{i+j\le 4}\left(m'_{ij}-m_{ij}\right)^2}, \tag{3}$$

and the range of the triangularity index is]0,1]. (Rosin 2003)

2.4.7 Ellipticity

Elliptical fields are not common in Finland, but this index was used as it was available and it was interesting to see whether such fields exist. Elliptical fields can be easy to operate by driving spiral. These kind of circular fields exist in some areas of the U.S., where fields have to be irrigated. They tend to be circular if a *center pivot* irrigation system is used.

Based on the comparison of ellipticity indexes in (Rosin 2003), the moment invariant method was selected for this research. The idea behind the moment invariant method is that any ellipse can be obtained from a circle using an affine transform. The simplest affine moment invariant, (4), characterizes the ellipses.

$$I_1 = \frac{\mu_{20}\mu_{02}-\mu_{11}{}^2}{\mu_{00}{}^4}\ . \tag{4}$$

For any circle, I_1 has constant value and therefore the index of ellipticity can be calculated as

$$E_I = \begin{cases} 16\pi^2 I_1, \text{ if } I_1 \le \dfrac{1}{16\pi^2} \\ \dfrac{1}{16\pi^2} I_1, \text{ otherwise} \end{cases}. \tag{5}$$

The range for the ellipticity index is [0,1].

2.4.8 Ratio of principal moments

If the polygon is not circular, the moment of inertia around different lines going through the centroid is different. The line, around which the moment is largest, is called the *first principal axis*. The second principal axis is perpendicular to the first. The moments of inertia around principal axes are called principal moments. The ratio of principal moments is another shape index for the compactness of the field.

For a square, the ratio has a value equal to one (1) and for a rectangle having an edge ratio 2:1, the value is 1/4 as moments are in the second order. As the ratio of shape distribution over principal axis is the property of interest, it is justified to use the square root of the moment ratio as a final index. Then the final value for a rectangle having an aspect ratio 2:1 is 1/2.

2.5 Results

The shape-index algorithms were run for all the field plots in the region of Uusimaa, Finland. The number of field plots is 65348. The data is from year 2005. Unfortunately, the set includes all field plots, also such *marginal strips* as those used between a cultivated field and waters, such as riverbanks and other areas, that prevent or restrict nutrient flow from field to waters. These marginal strips are not used for cultivation, but are subject to farming subsidies and therefore included in the data set. Based on visual inspection of the data, it seems that the number of these marginal strips in the data set is about 0.5%. The total area of field plots under study is 195000 ha. This area covers about 7.7% of Finnish fields. The data set contains information on obstacles if the area of one obstacle is over 25 m^2. Information on small obstacles such as electricity poles is not available.

First, in Figure 1, the histogram of field plot areas is shown on a logarithmic scale. The average area is 3.0 ha. The largest field plot is 92 ha, the second largest 69 ha and the third largest 65 ha. Seven fields are over 50 ha, 118 over 30 ha and 472 fields over 20 ha.

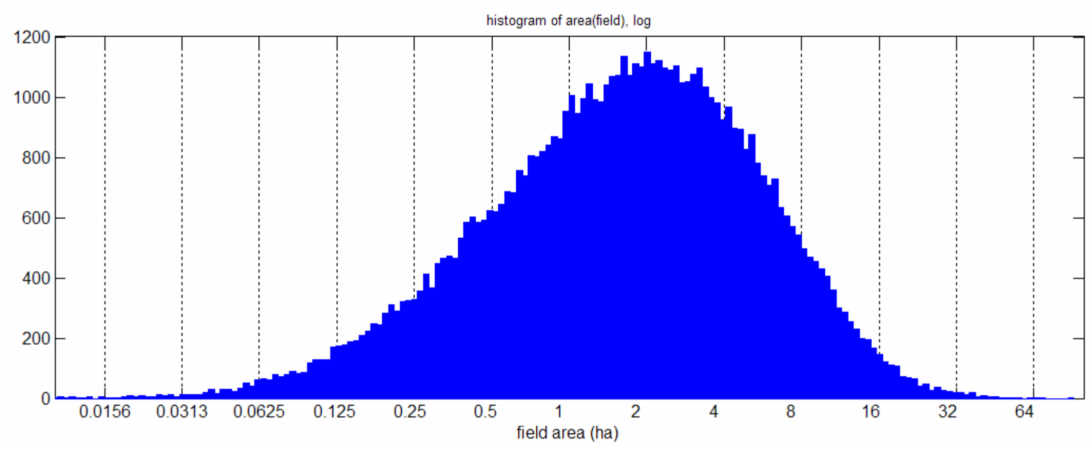


Figure 1. Histogram of field area

The histograms of shape indexes are shown in Figure 2. The first thing to note is that the distribution of convexity is far from normal. 2850 field plots (4.3%) have a convexity value of 1.0000 and about 8500 field plots (13%) have a value of ≥0.99. Ellipticity is another shape index that does not follow the normal distribution. There seems to be two reasons for this: there are quite a few fields that are elliptical or nearly so; on the other hand, the index seems not to be linear.

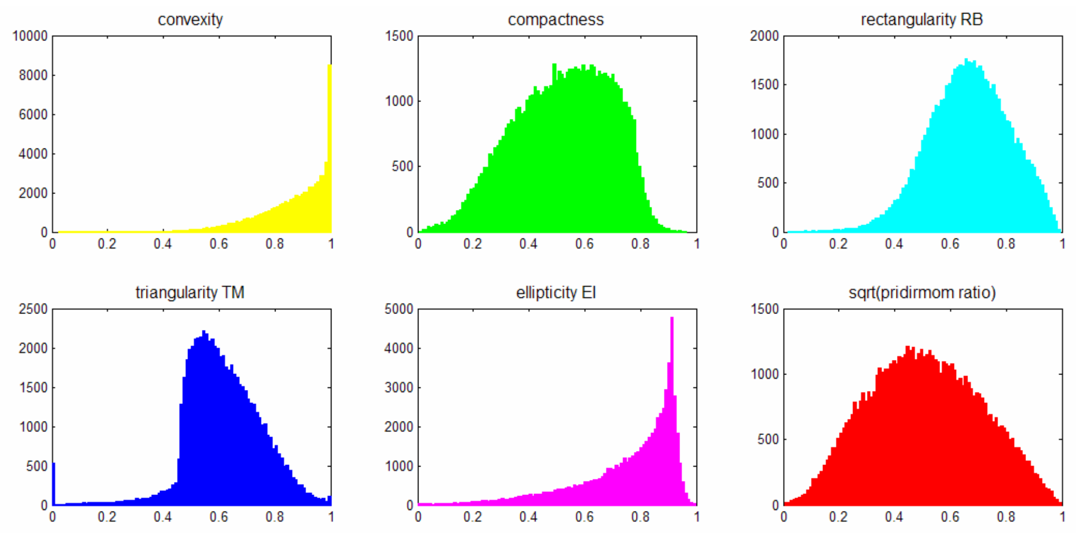


Figure 2. Histograms of shape indexes

In Figure 3, the matrix plot of shape indexes is presented. The histograms are plotted diagonally and the 2D projections to the other cells. The horizontal axis of projection is always the header of the column. For example in the plot at the fourth column and the second row, the horizontal axis is triangularity and the vertical axis is compactness. It can be seen that there is a correlation between shape indexes: rectangularity, ellipticity and convexity are correlated.

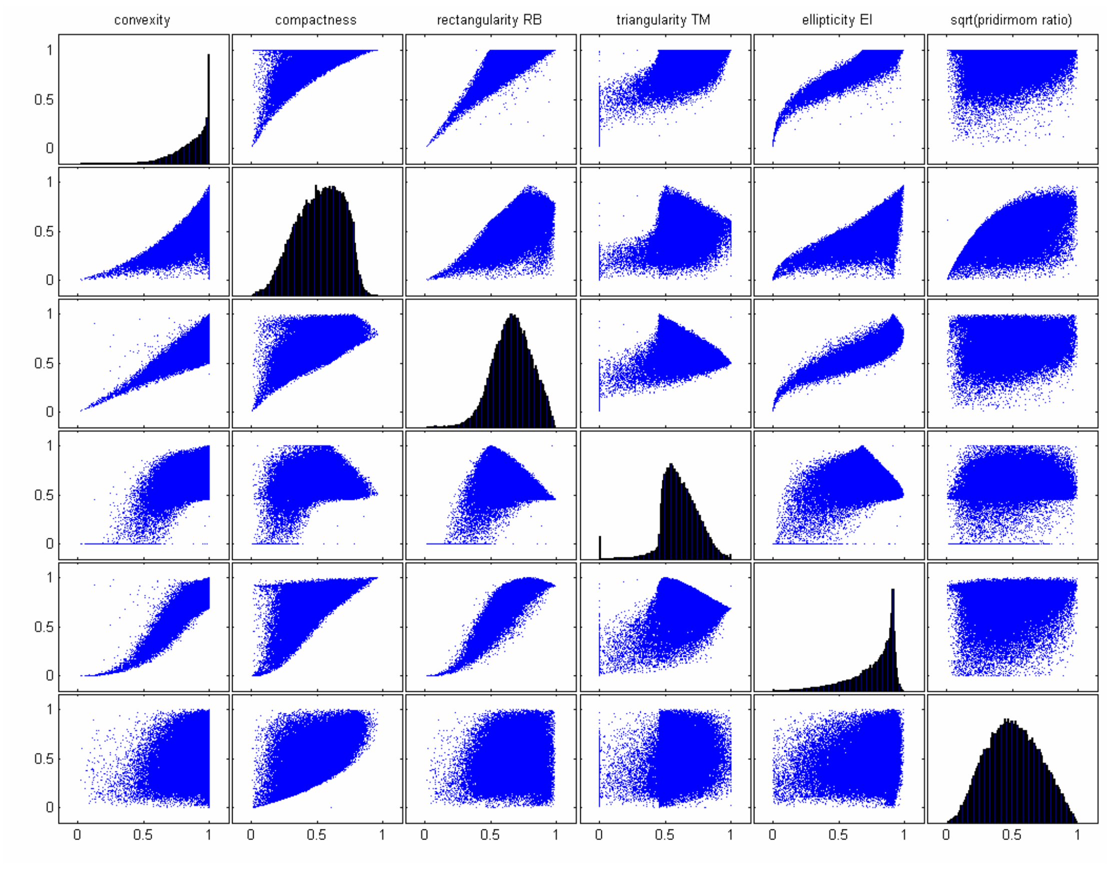


Figure 3. Matrix plot of shape indexes

In the following figures (4-9), some shapes with indexes are presented in order to analyze the linearity and scaling of each shape index, comparing them for visual analysis. The fill color varies so that the figures are easier to distinguish.

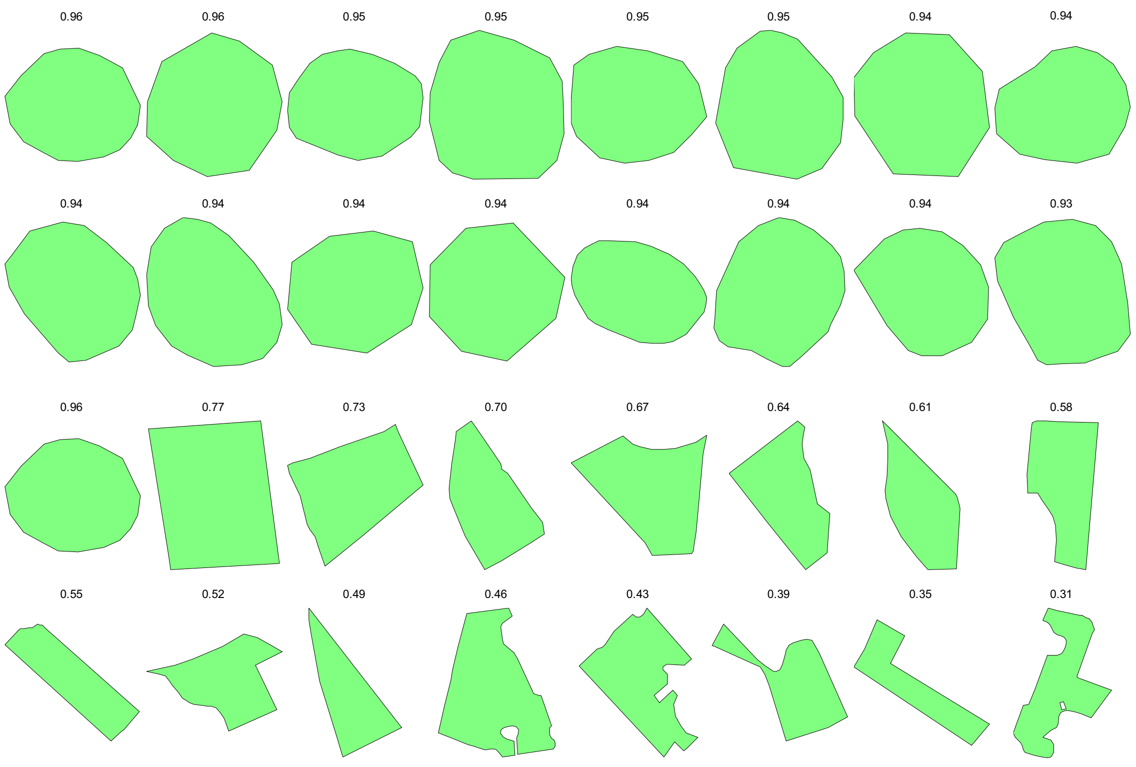


Figure 4. Compactness

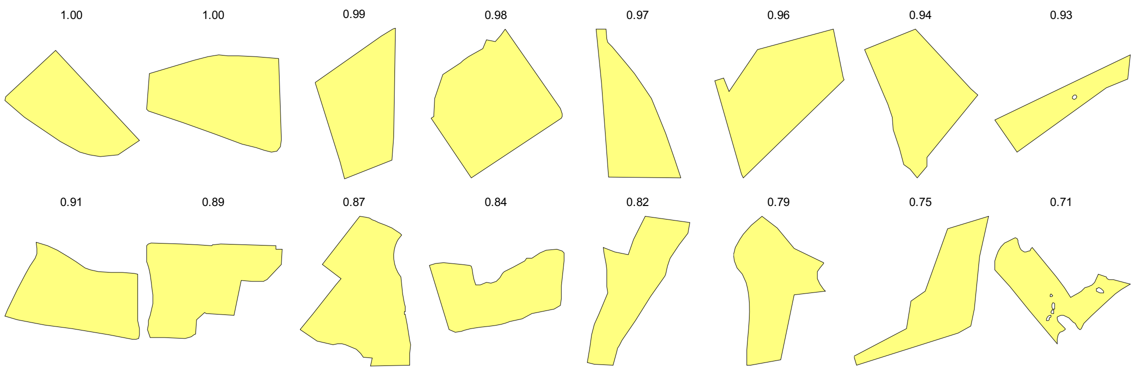


Figure 5. Convexity

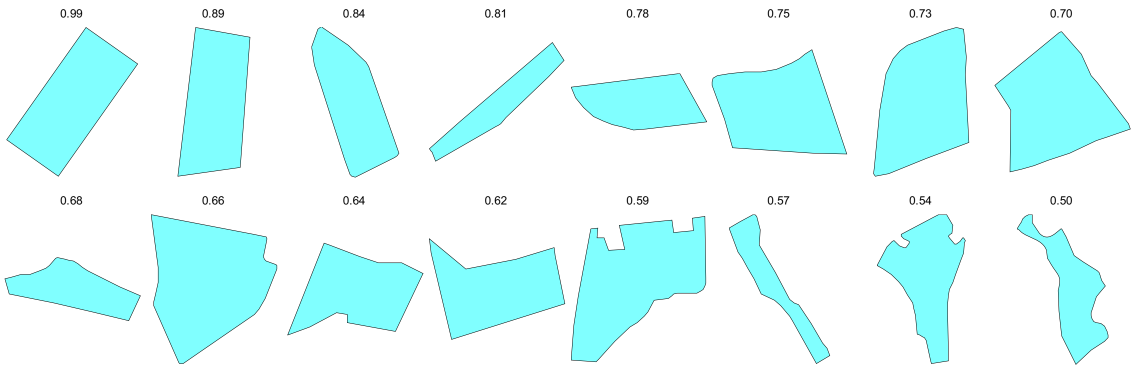


Figure 6. Rectangularity

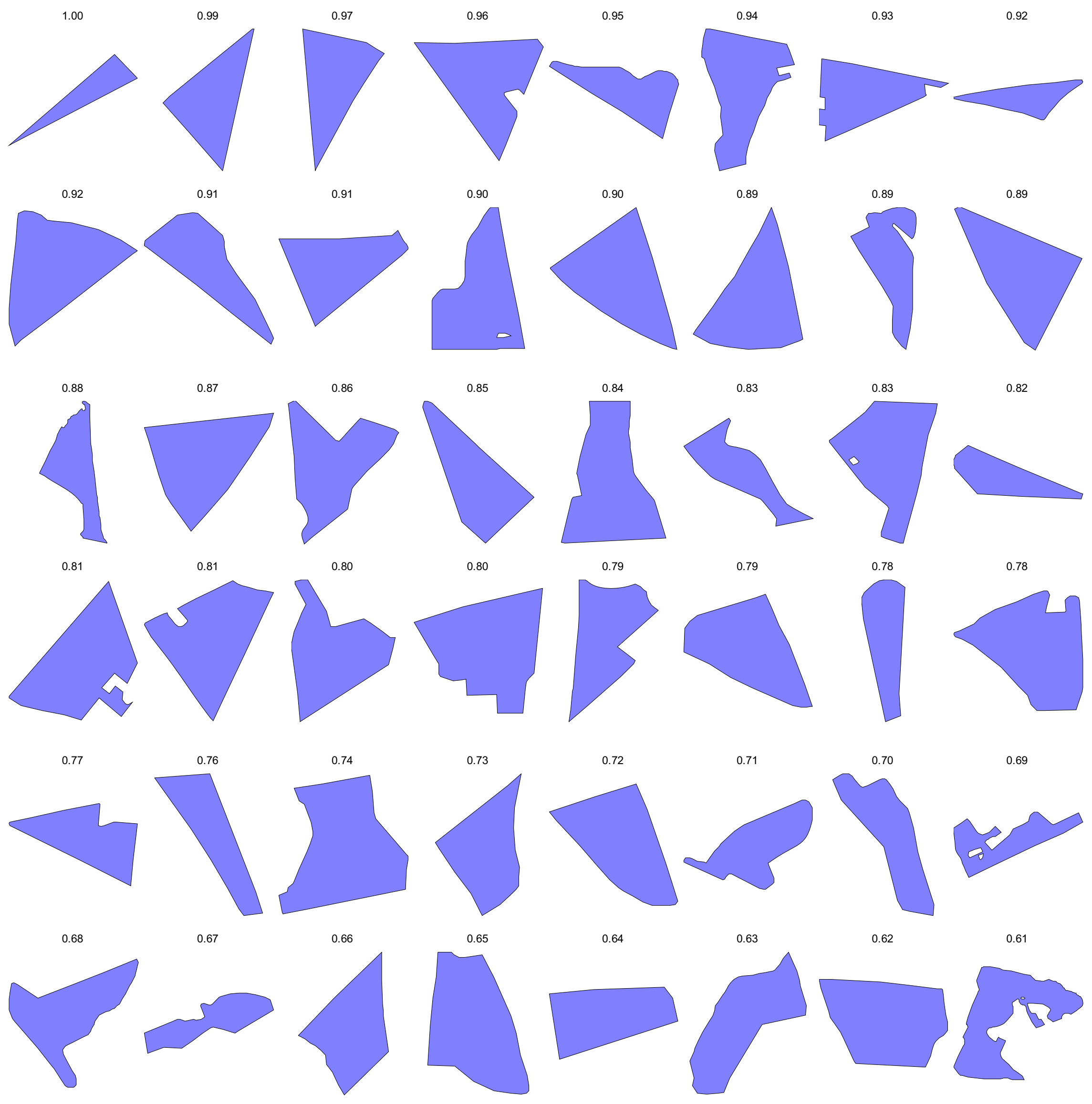


Figure 7. Triangularity

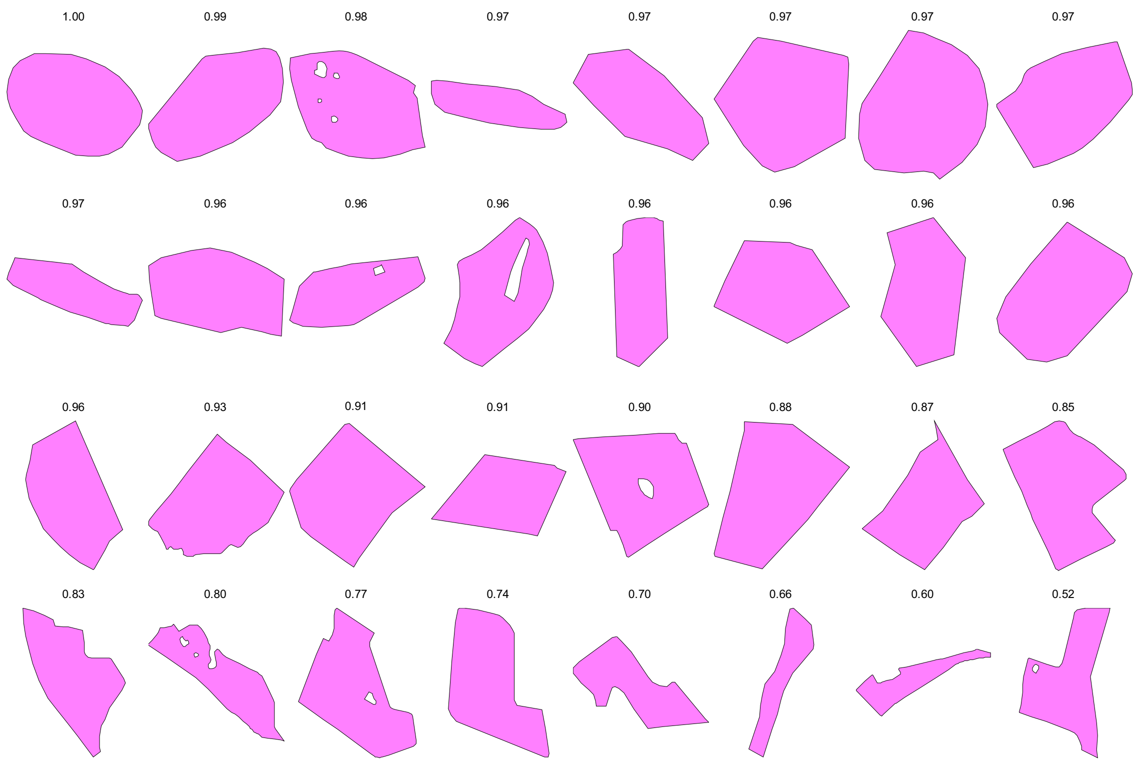


Figure 8. ellipticity

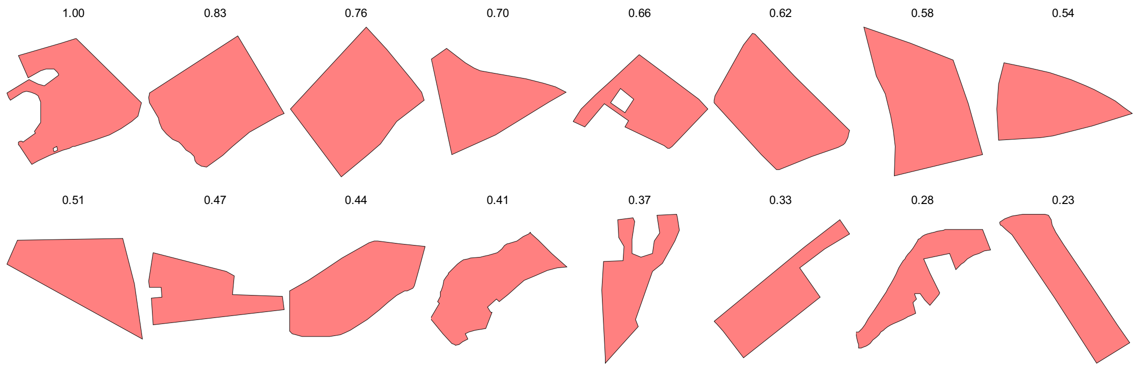


Figure 9. Ratio of principal moments

2.6 Classification

2.6.1 Approach

At first, self-organizing maps (SOM) were used on the six shape indexes to see if clusters could be found. Unfortunately, the clusters were not found in the projected space. We will come back to this topic later. A lack of clusters can also be seen in Figure 3.

Based on that experience, it was decided first to use the supervised classification and take out all the field plots that have well-defined and easily detectable shapes from the data set. SOM was applied to the rest of the data set to see if it could find any other generalizable shapes.

2.6.2 Thresholding and conjunction

The main regular shapes existing in field plots are rectangular and triangular. These can be detected using only a single measure, as rectangularity and triangularity indexes were previously used. In rectangular fields, the aspect ratio or the ratio of edge lengths is used to find subclasses, as this may affect the path planning.

For rectangularity, the threshold was determined visually to be 0.80, (see Figure 6). For the aspect ratio, the square root of principal moment ratio was used, i.e. one equals square, 0.5 equals 2:1 etc. The threshold value is in the middle of exact values. Conjunction operator (AND) is applied to thresholded variables.

For the triangularity, the threshold was determined to be 0.85. Very long strips have relatively high triangular value and, as they do not represent the triangle as one might think, the thresholded moment ratio was also used to classify only the "real" triangles. The threshold value for the moment ratio was set to 0.6, based on trial and error.

Circular fields were detected using ellipticity (≥0.97), compactness (≥0.6) and the square root of principal moment ratio (≥0.6).

Of the fields that are convex, 13.1% do not contain any obstacles. These are easy cases for coverage path planning, as we will see later; therefore, this class is set.

The simple classes and number of hits are listed in Table 1. About 19% of the fields are rectangular, less than 1% triangular and circular fields are very rare. About 25% of all field plots were classified into some of the presented classes; 75% were unclassified.

Randomly taken samples from each class are shown in Figures 10-18.

Table 1. Identified classes

Class	Field plots in class	% of field plots
Squared field	1735	2.7%
Rectangular 3:2	3461	5.3%
Rectangular 2:1	2886	4.4%
Rectangular 3:1	1919	2.9%
Rectangular 4:1	959	1.5%
Rectangular 5:1	483	0.74%
Rectangular >5:1	894	1.4%
Rectangular total	12337	18.9%
Triangular	448	0.69%
Circular	65	0.099%
Convex total (of all fields)	8588	13.1%
Classified total	16180	24,8%
Unclassified	49168	75,2%

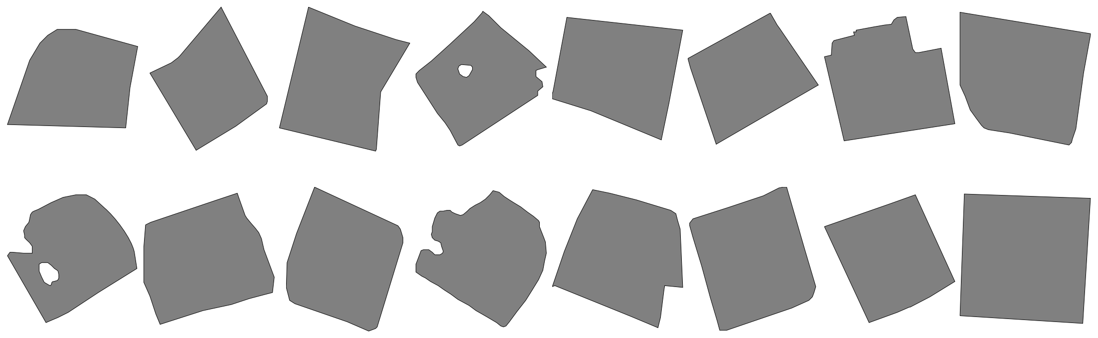


Figure 10. Squared fields

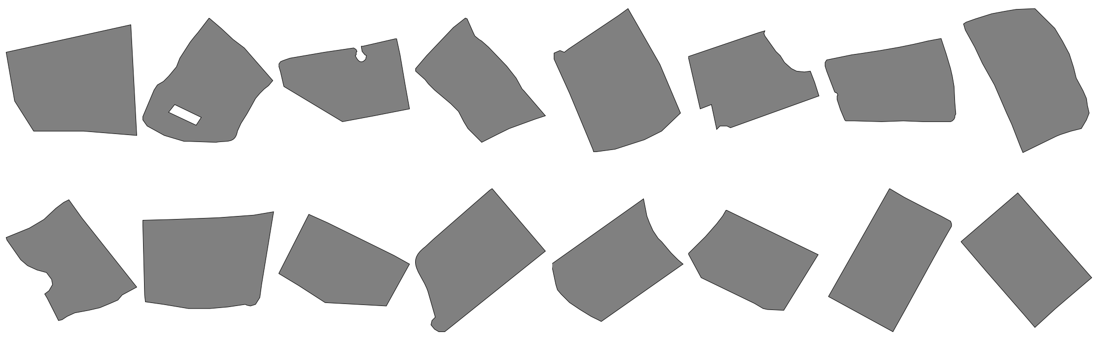


Figure 11. Rectangular 3:2 fields

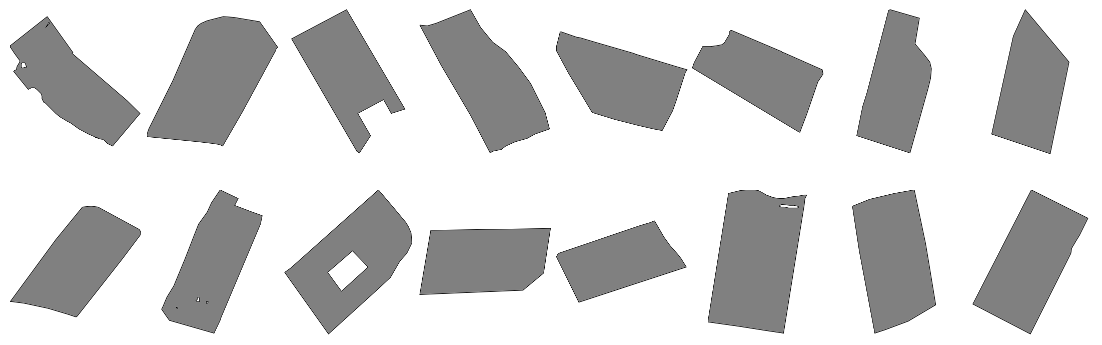


Figure 12. Rectangular 2:1 fields

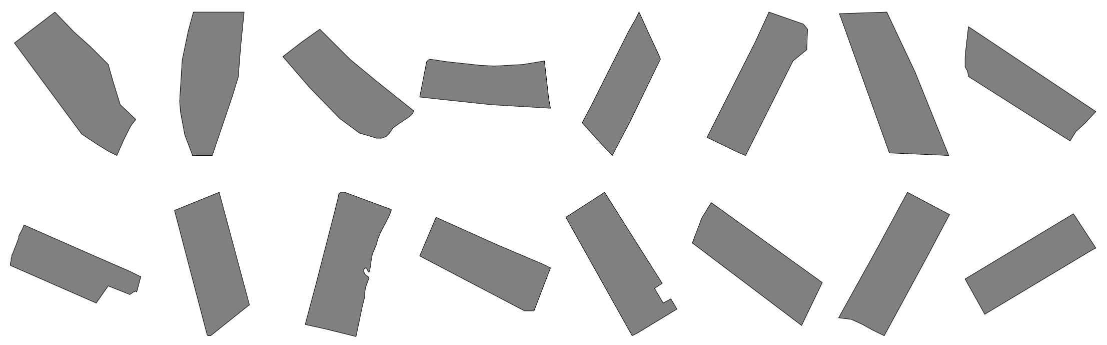


Figure 13.Rectangular 3:1 fields

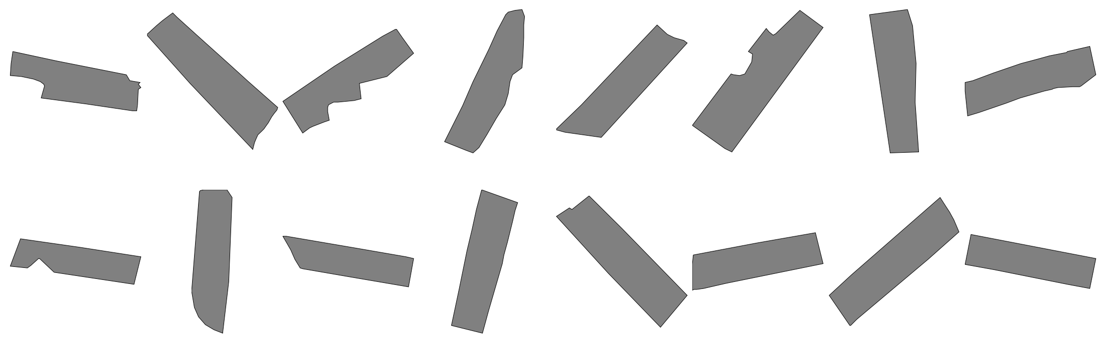


Figure 14. Rectangular 4:1 fields

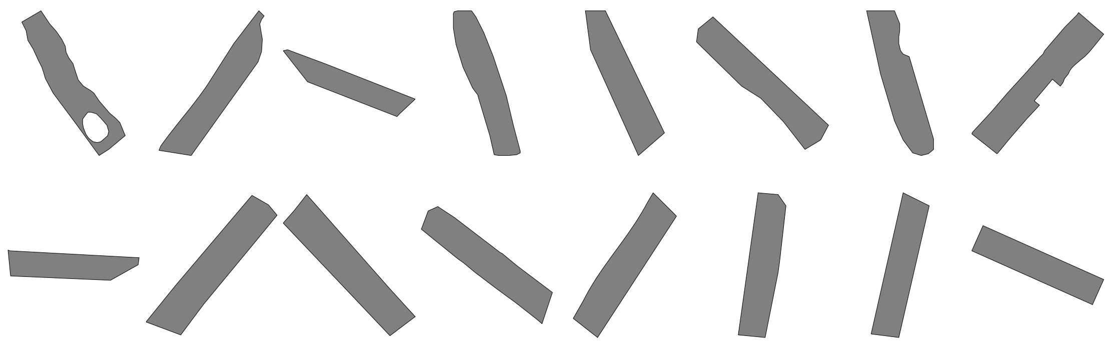


Figure 15. Rectangular 5:1 fields

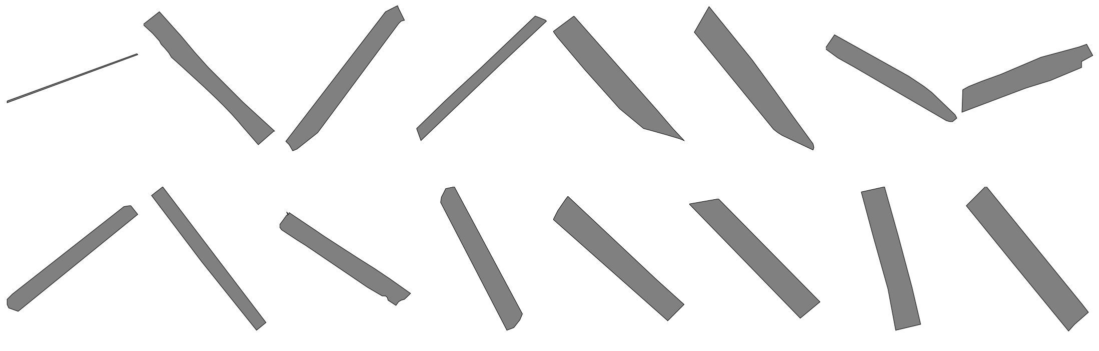


Figure 16. Rectangular >5:1 fields

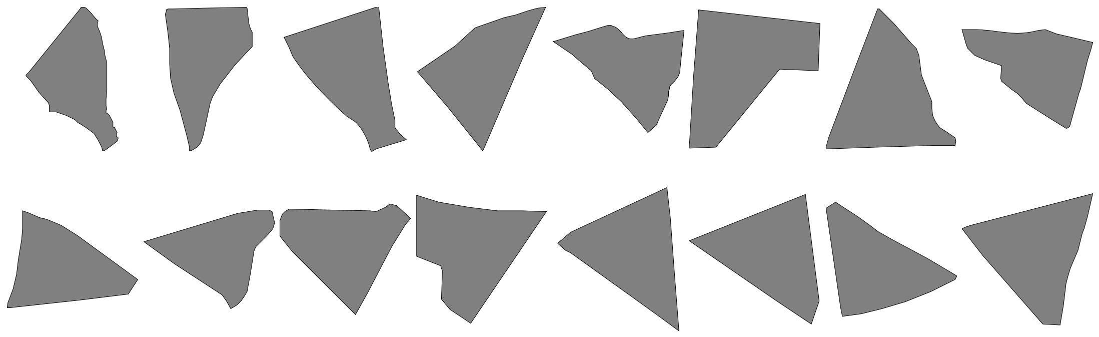


Figure 17. Triangular regular fields

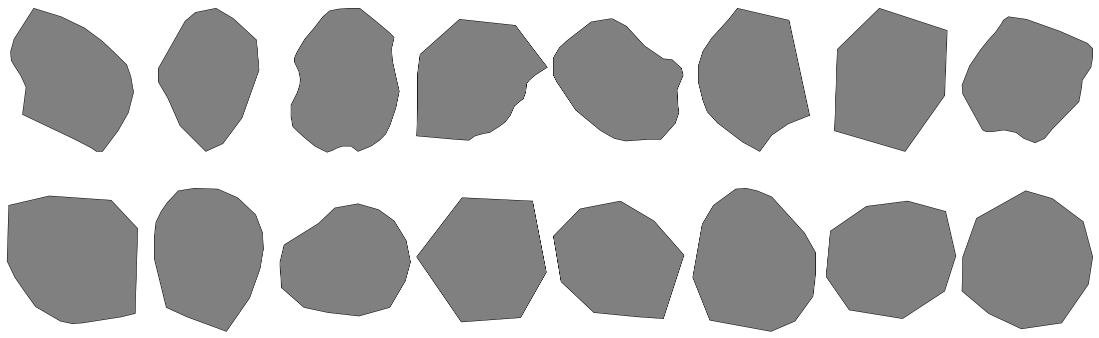


Figure 18. Circular fields

2.6.3 Self-organizing maps

A *self-organizing map* (SOM, Kohonen map) is a common way to represent multidimensional data in a lower-dimensional space. Usually the output space has two dimensions, so it can be easily visualized. A self-organizing map attempts to group similar data items together, while reducing the dimensions.

In this thesis, Matlab SOM Toolbox (Alhoniemi *et al.* 2005) has been used to make the unsupervised learning for the field-shape indexes and project the data to a two-dimensional output grid. Only the unclassified field plots from the previous phase are used here. The results are shown in Figure 19 and Figure 20, the unified distance matrix (U-matrix) and the components projected to the output grid, respectively. It can be seen that the minimum values of each component are all projected to the same corner of the output grid. Actually, there are all the marginal strips, all field plots with very long perimeter, low area and curved edges. Correlation between convexity, rectangularity and ellipticity can be seen, too.

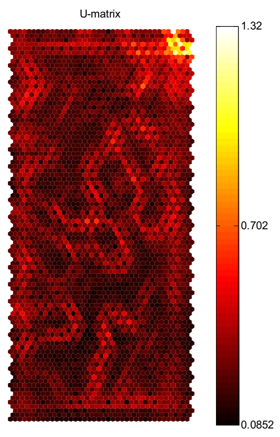


Figure 19. SOM: unified distance matrix

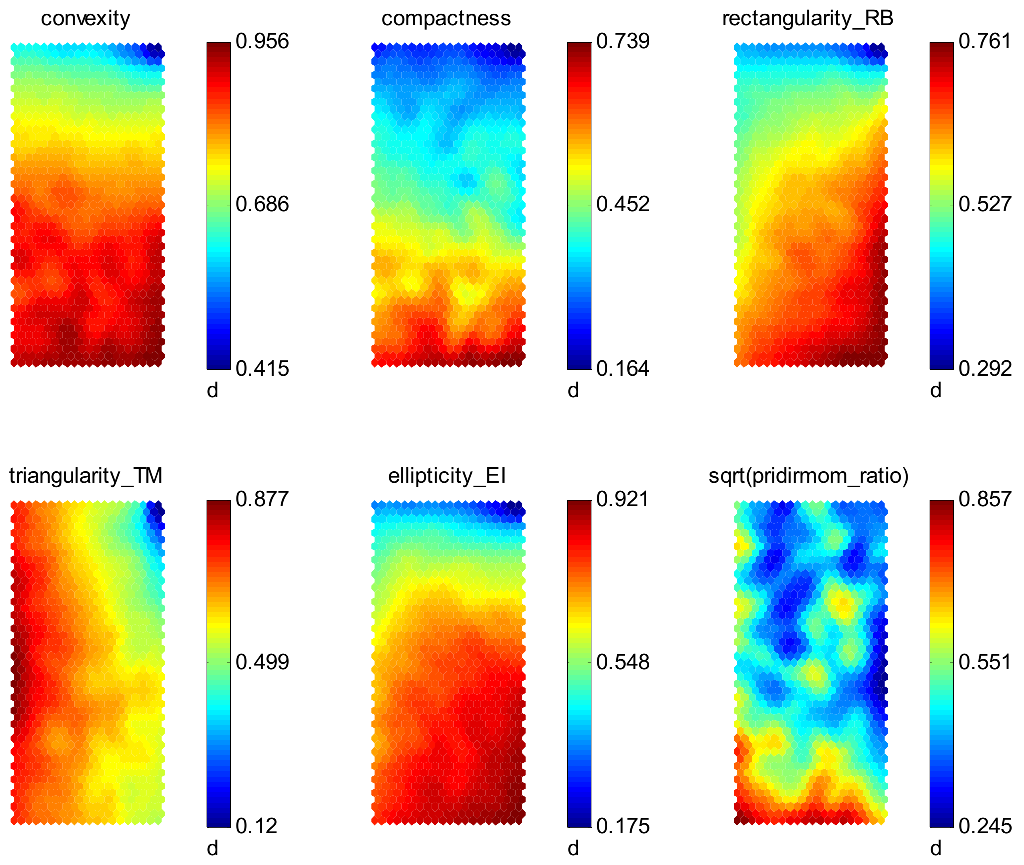


Figure 20. SOM: component projection

One may think that some spots that are visible both in component projection and U-matrix could be some kind of field-plot shape classes or clusters. Unfortunately, they are not suitable for classes, because if all the field plots in one cell or cluster are plotted and analyzed by a human classifier, there is no clear visual correspondence between them.

In Figure 21, field plots are presented in trained SOM output space. In other words, field plots to the top-left are from the top-left in grids shown in Figure 19 and Figure 20. Not all cells can be printed here; the resolution was decreased to 20×14. In each output cell, many field plots were projected, and one of them was randomly selected for this print. Field plots in a single cell were analyzed and, generally, it can be said that there was no correspondence between them. In other words, if two neighboring shapes look similar, that may be just a coincidence. Only in some cells, such as those on the middle-left where the most triangular field plots are projected, is internal correspondence evident. Based on Figure 21, it can be said that a deeper classification is hard to achieve.

Some more field plots could be classified into triangular and rectangular classes, but that is a matter of thresholding and always a matter of taste, when there are no clear clusters. In addition, there are at the bottom some shapes that could belong to the shape of "trapezoid", but those fields do not form a compact group. It would need another shape index (trapezoidality) to make this classification.

It is clear that, at the bottom (in Figure 21), from path planning point of view there are more "nice" fields than on top. But "nice" or "dirty" or "emmental" are not good bases for classification. So the result is that 25% of field plots were classified; it was not possible to classify more into shape-based classes.

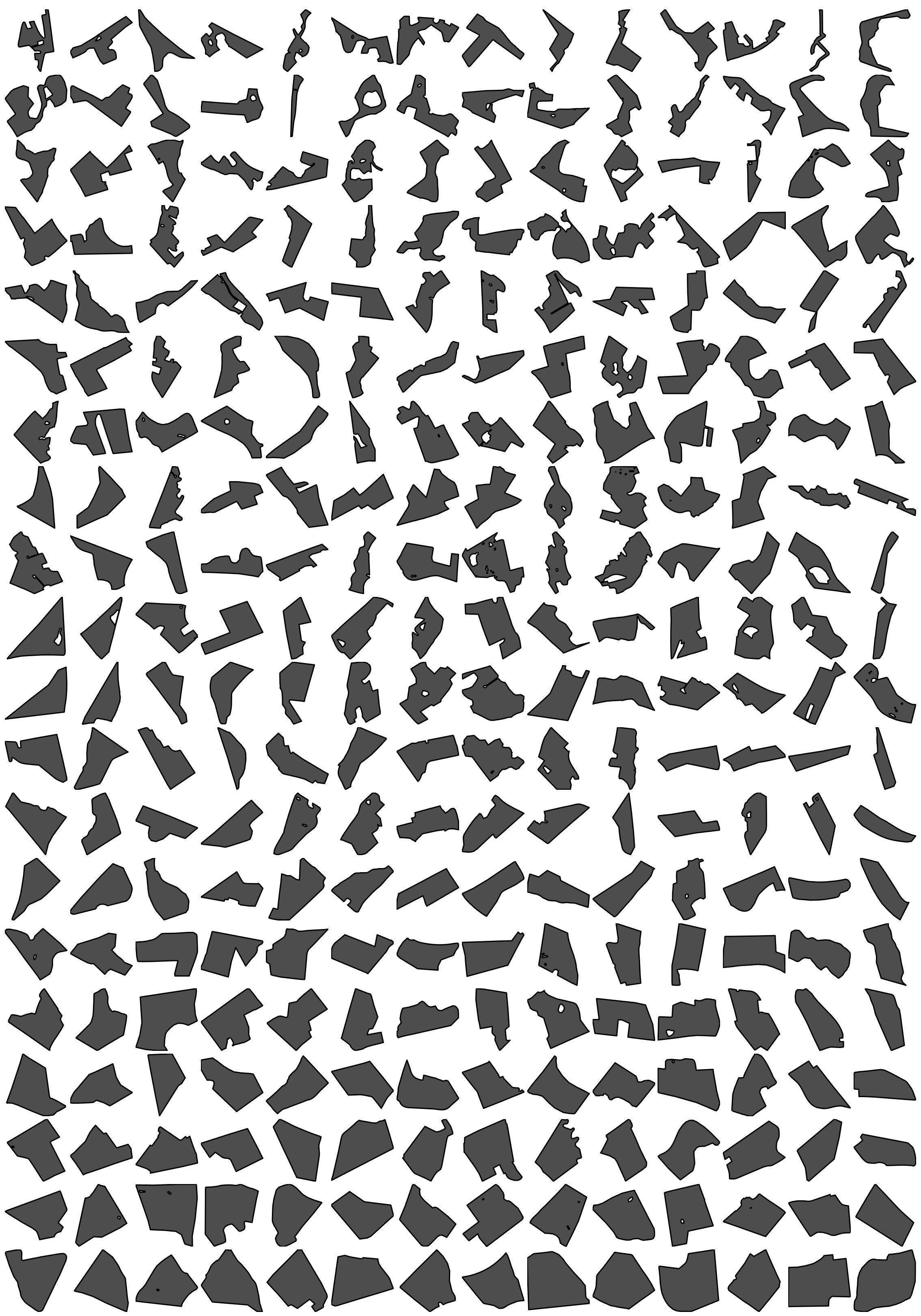


Figure 21. Sample fields in SOM space

2.7 Very hard field shapes for path planning

The data set of Finnish fields in the region of Uusimaa, uncovers plenty of field plots for which the path planning would be very challenging. Some of these were also detectable in Figure 21. For interest, some of these "most challenging" fields are shown in the following figures. At this time, before going deep to the path planning problem in later chapters, the reader is encouraged to think how these fields could be driven efficiently, using just common sense.

The purpose of including the following extreme examples in the thesis is to show how the universal path planning algorithm should cope with difficult fields. It can be said that, even if the optimal solution could be found for these, it could not be said to be efficient in terms of time consumed per area operated, as the number of turnings per area unit would be high.

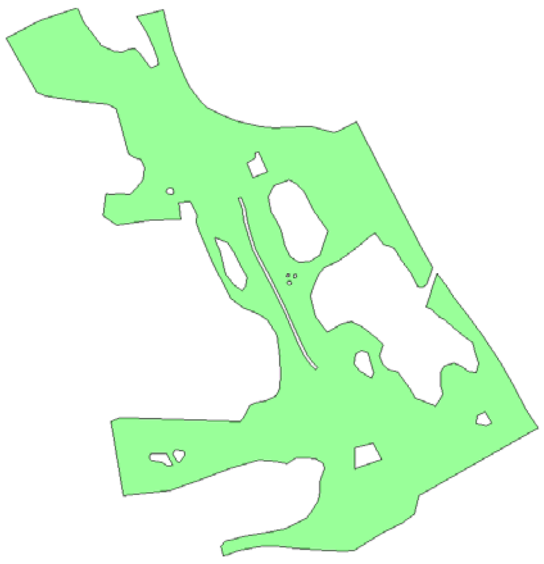


Figure 22. Very incompact field A (13 obstacles, 11,5 ha)

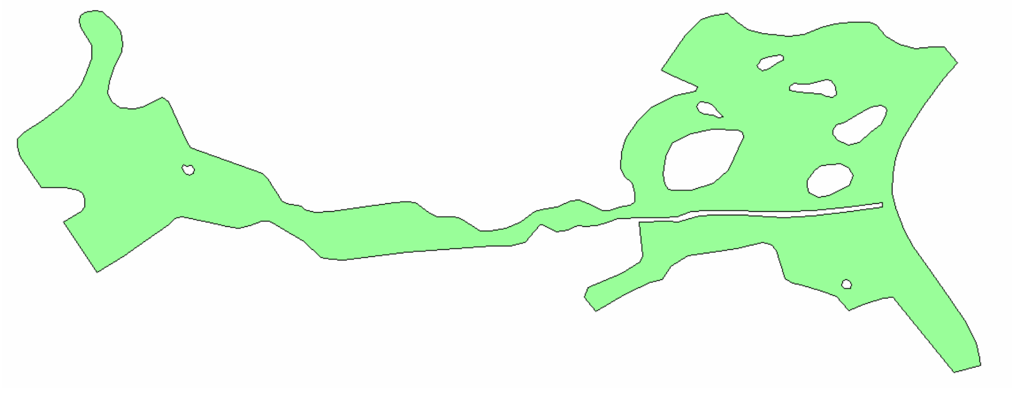


Figure 23. Very incompact field B (4,1 ha)

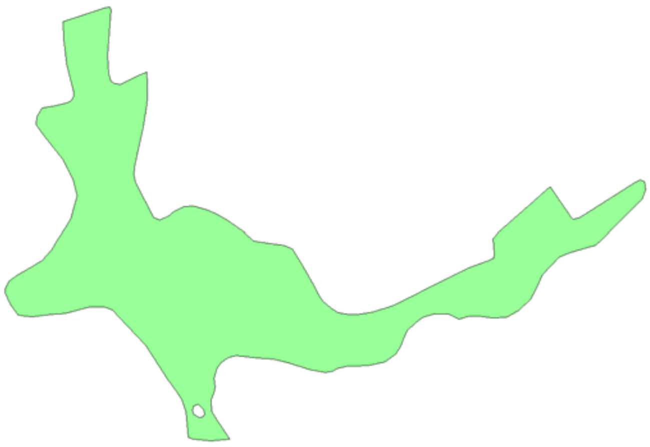


Figure 24. Incompact field C (3,8 ha)

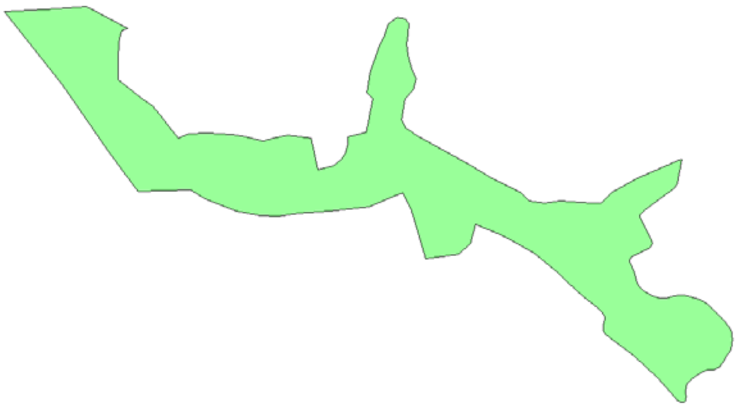


Figure 25. Incompact field D (8,3 ha)

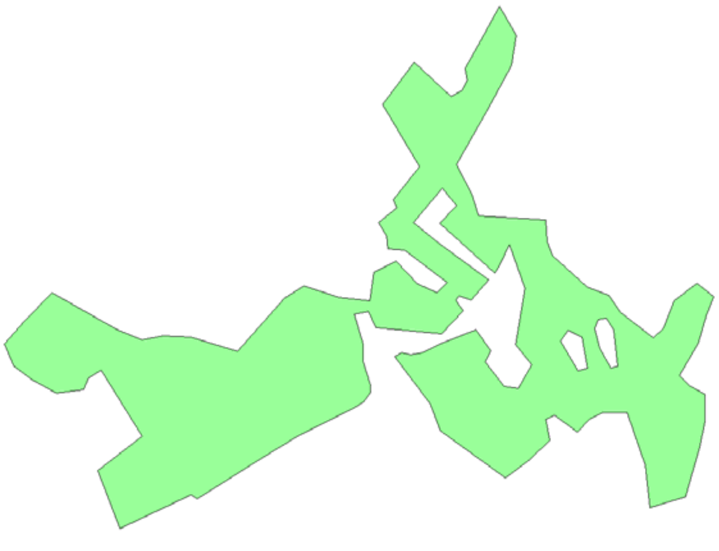


Figure 26. Incompact field E (6,9 ha)

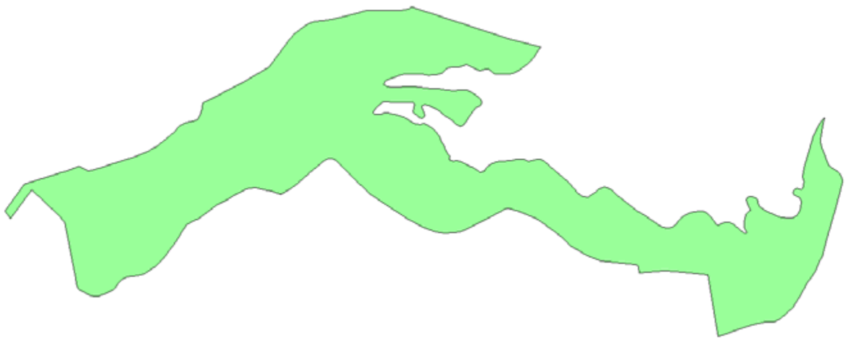


Figure 27. Incompact field F (34,6 ha)

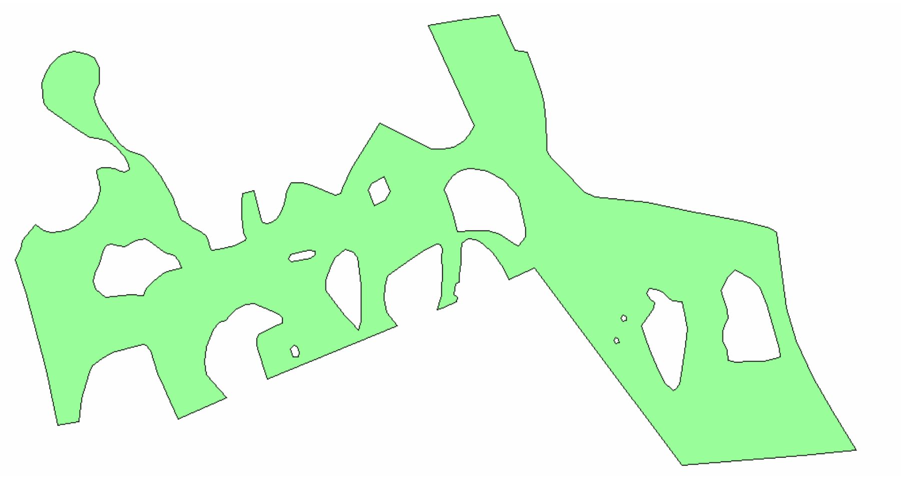


Figure 28. Incompact field G (13,4 ha)

2.8 Conclusion

The shape of Finnish fields has not been extensively researched. The goal of this study was to see from the data if something could be said about the shapes using known shape-describing indexes: convexity, compactness, ratio of principal moments, rectangularity, triangularity and ellipticity. All the indexes were successfully calculated for the subset of Finnish field plots, all fields in Uusimaa region. The classification of field shapes was not as successful as believed would be the case when starting this work; only 25% of field plots were classified to some clear class. The rest are more or less troublesome to classify, or can be classified into the class of *complex shapes*. The main problem is that field plots, which do not have a simple shape, can be anything, and if classification were to be done, the number of classes would be very large. However, in this research, much more information was obtained than before: earlier research had concentrated only on analysis of area versus perimeter of field plot.

One interesting shape index that may give more information to classification is trapezoidality. However, no algorithm to calculate that was found and it was not considered reasonable to start developing such an algorithm in this research.

Unfortunately, it was not possible to carry out the study of all Finnish field plots in this research. There is some variation in shapes across the country, as has been shown in earlier studies.

In this chapter, the challenge for path planning algorithms for agricultural field machines is also shown. There are simple shapes for which simple back-and-forth swathing can be applied directly, such as rectangular and convex fields. However, 75% of field plots are not simply shaped. This fact presents the real challenge for path planning. Some examples of very tricky fields are shown in this chapter.

However, later in this thesis, the split and merge algorithm is presented. This algorithm can also be used to classify fields, based on the output statistics that the algorithm gives.

Development of a trapezoidality index, analysis of all Finnish field plots and analysis based on split and merge algorithm results could be topics for future research.

3 Headland turning

3.1 Introduction

When optimizing field traffic, it is most significant to minimize the time spent in the headlands. The fewer turnings that are made in a field, the less time the turnings take in total. But efficient vehicle behavior in a single turning is also important.

A tractor-trailer combination or a tractor with a towed implement, such as a seed driller, harrow, fertilizer or sprayer, is a common form of agricultural machine. The example of such a machine is shown in Figure 29, which shows a trailer being towed as a combined fertilizer-seed drill. The behavior of a tractor-trailer combination in a headland is studied in this chapter.

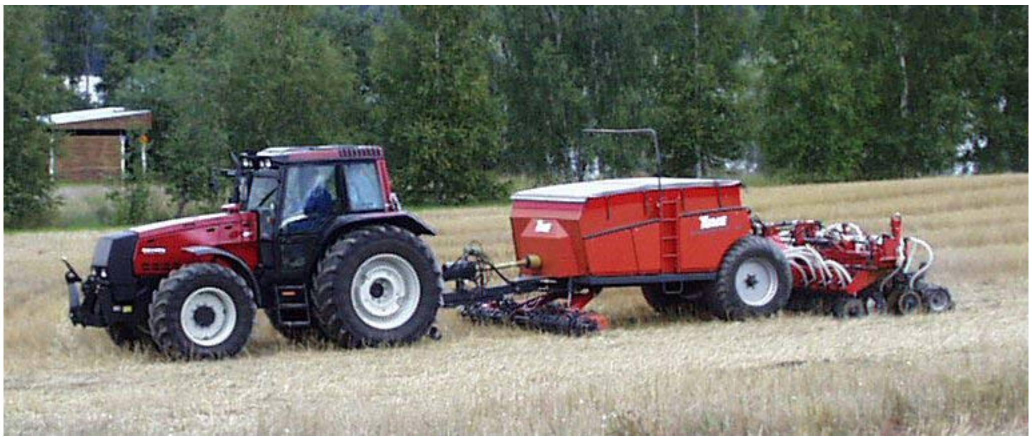


Figure 29. Tractor-trailer vehicle

In this chapter, the tractor-trailer combination is modeled as a differential equation system or dynamical system with six states and two inputs. The movement is restricted to a 2D-plane. The supplementary requirements for the model are the ability to reverse drive, allowing the switch-back turning and the standard formulation of the optimal control problem. The dynamical model is parameterized in terms of the mechanical parameters of the vehicle.

Optimal control theory and tools are reviewed. The minimum-time optimal control problem of headland turning with a tractor-trailer vehicle is presented. The vehicle's mechanical and the field's geometrical constraints are considered. The optimal control problem is highly nonlinear and cannot be solved analytically.

Numerical algorithms for solving this kind of optimal control problems demand high computational resources and cannot be solved online (online trajectory generation) even with modern desktop computers. Here the optimal control problem is solved with fixed vehicle parameters and varying headland parameters, headland angle and width. In order to use optimal control problem solutions online, the solutions in variable headland cases are approximated with simple trajectories. The rules for selecting a turning method with fixed vehicle parameters and certain headland parameters are presented.

3.2 Modeling of vehicle

The dynamical model will be used in the optimal control problem so the system should be modeled carefully. The vehicle is considered to move in a 2D-plane, thus the tractor itself contains 3 degrees of freedom (DOF). (x_R, y_R) is the position of tractor rear axle and θ is heading angle of tractor. The trailer can be added to the tractor model with one DOF, the angle of drawbar φ. These parameters are visualized in Figure 30.

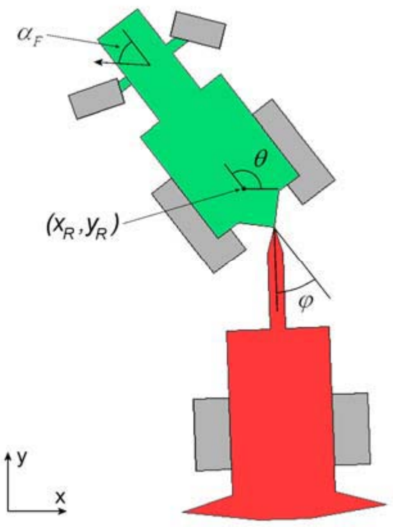


Figure 30. Vehicle parameters

The physical parameters of the vehicle are the axle spacing of the tractor (a), the spacing between tractor rear axle and drawhook (or drawbar coupling) (b) and the spacing between drawhook and trailer axle (c). This parameterization allows the joint of the vehicle to be in the drawhook, in the rear hitch drawbar coupling or in the free joint-drawbar. The functional component of the trailer can be in front of the trailer axle or behind it (*d*). If it is behind, the parameter *d* is positive and vice versa. Only the front wheels of the tractor are steerable. The parameters *a*, *b*, *c* and *d* are visualized in Figure 31.

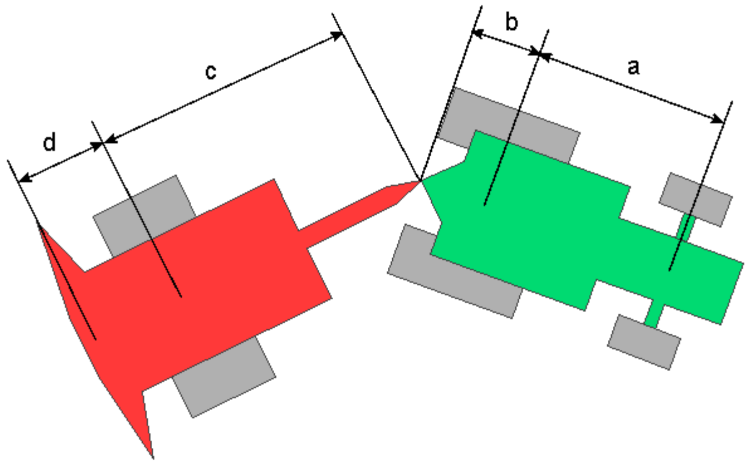


Figure 31. Physical parameters of the model.

The dynamical model is presented in (6) and the auxiliary variables (x_H, y_H, v_H, β) needed are presented in (7). The model contains six states and two controls, x and u respectively. (x_R, y_R) is the position of tractor rear axle, θ is heading angle of tractor, φ is drawbar angle. There are two physical controls for the system, the velocity of the tractor (rear) wheels and the steering angle of the front wheels. In order to avoid the derivation of states in checking optimal control constraints (e.g. acceleration of vehicle), the physical controls are considered as states (v_R, α_F) of the system and the change (time derivative) of those are the inputs of the system (a_R, ω_F), acceleration and angular velocity of steering wheel angle.

$$\begin{cases} \dot{x}_R = v_R \cos(\theta) \\ \dot{y}_R = v_R \sin(\theta) \\ \dot{\theta} = \dfrac{1}{a} v_R \tan(\alpha_F) \\ \dot{\varphi} = \dfrac{1}{c} v_H \sin(\beta - \varphi - \theta) - \dot{\theta} \\ \dot{v}_R = a_R \\ \dot{\alpha}_F = \omega_F \end{cases} \tag{6}$$

$$\begin{aligned} \dot{x}_H &= \dot{x} + b\sin(\theta)\cdot\dot{\theta} \\ \dot{y}_H &= \dot{y} - b\cos(\theta)\cdot\dot{\theta} \\ v_H &= \sqrt{\dot{x}_H^2 + \dot{y}_H^2} \\ \beta &= \arctan(\dot{x}_H, \dot{y}_H) \end{aligned} \tag{7}$$

$$x = \begin{bmatrix} x_R & y_R & \theta & \varphi & v_R & \alpha_F \end{bmatrix}^T, u = \begin{bmatrix} a_R & \omega_F \end{bmatrix}^T \tag{8}$$

In order to complete the model, the physical limits of the vehicle have to be considered. The dynamical system has been chosen so that all the limits are saturation type. The limits for the example vehicle are shown in (9). The first one is the limit for velocity (negative speed corresponds to driving backwards), the second is the acceleration and deceleration limit, the third is the maximum angle of front wheel steering angle, the fourth is the angular velocity of steering wheels and the last one is the limit for drawbar angle.

$$\begin{cases} v_R^{\min} \le v_R \le v_R^{\max} & km/h \\ a_R^{\min} \le a_R \le a_R^{\max} & km/h/s \\ -\alpha_F^{\max} \le \alpha_F \le \alpha_F^{\max} & ^\circ \\ -\omega_F^{\max} \le \omega_F \le \omega_F^{\max} & ^\circ/s \\ -\varphi^{\max} \le \varphi \le \varphi^{\max} & ^\circ \end{cases} \tag{9}$$

Usually the numerical optimal control problem solvers can handle single state constraints more efficiently than handling them all together. Therefore it is important to divide simple single-state constraints and mixed-state constraints. The constraints for the last three states and both controls can be considered as static constants opposite to the limits of three first states. These describe the position of the vehicle in the field. Therefore those limits for three first states are separated from the others. Generally speaking, the limits of first three states cannot be separated; the mixed path-constraints have to be used for them.

3.3 Optimal control problem

The earliest origins of optimal control are in the calculus of variations in the 17th century. The classic optimal control theory was developed mainly in the 1950's and 1960's; the motivation was space and aeronautics engineering. The principle of optimality, dynamic programming, Bellman equations and Pontryagin maximum principle come from that era. The problem of these theories is that they solve the optimal control problem only for certain problem types, such as problems with linear dynamics and a quadratic cost function, for which the analytic solution can be obtained from necessary and sufficient conditions of optimality. For nonlinear dynamics and the non-quadratic cost function, a numerical solver is needed (Bryson 1996).

An optimal control problem (OCP) can be formulated as:

$$\min J(u) = \int_{t_0}^{t_f} F(x(t), u(t), t)\, dt \,, \tag{10}$$

$$\dot{x}(t) = f(x(t), u(t), t),\ t_0 \le t \le t_f \,, \tag{11}$$

$$C(x(t), u(t), t) \le 0,\ t_0 \le t \le t_f \,, \tag{12}$$

$$E(x(t_0), x(t_f)) \le 0 \,, \tag{13}$$

where x is a state vector, f dynamics function, t_0 and t_f the time limits, J the cost functional, F the cost function, C the path constraint function and E the boundary condition function (von Stryk *et al.* 1992).

The numerical methods that rely on solving the necessary conditions of the Pontryagin maximum principle are usually called indirect methods. The direct numerical methods have improved during the 1990's and are still under research. In direct methods, the infinite dimensional problem (from maximum principle) is approximated with a finite dimensional problem. The philosophy in direct methods is to discretize the problem in time and to approximate the states and the controls in those time points and convert the optimal control problem (OCP) to a nonlinear programming (NLP) problem. The NLP problem is a standard optimization problem that can be solved efficiently with numerical tools.

One of the first direct methods was called direct shooting through parameterization of controls $u(t)$. The method relies on numerical integration to satisfy the dynamical model. Another method uses discretization of both the controls and the states in time, model dynamics is satisfied in discrete points with equation constraints. End-point constraints are also of the equation type, but path constraints are inequality constraints in NLP. States are approximated with polynomials or more advanced approximations between discrete time points. The result is a nonlinear programming problem. This method is often called *direct collocation*. In the very basic direct collocation method, the dimension of the NLP problem grows quite large, so the calculation time of numerical NLP solvers is quite long. The disadvantage of direct collocation methods is that they produce less accurate solutions than indirect methods. The usability of the solution depends on the accuracy requirements of the original optimal control problem. The other disadvantage of the method is that the numerical solution may be a local minimum, not global, and the local minimum found can be far from the global one (von Stryk *et al.* 1992).

One of the latest direct numerical methods is the so-called pseudospectral approximation, which is an adaptive algorithm (Ross and Fahroo 2002). The pseudospectral method can efficiently handle discontinuities in particular, but also other nonsmoothnesses of state trajectories. However, it also converges fast.

There are commercial tools available for solving OCP numerically. One of them is DIDO, commercially available from Tomlab Optimization. DIDO uses the pseudospectral approximation algorithm (Ross 2004). DIDO utilizes the SNOPT algorithm (Gill *et al.* 2002) to solve NLP problem.

The OCP solver tool is supplied with a mathematical description of equations of the dynamical system, all boundary conditions of states, all state and control limits and the optimization criterion. In DIDO, all the descriptions are programmed as MATLAB-functions.

3.4 Tractor-trailer in headlands

As stated previously, the time spent in headlands is crucial when minimizing the time spent in the whole field. The turning of the tractor-trailer combination in headlands is studied below. The parameters used in this case are shown in Table 2.

Table 2. The fixed parameters of the tractor-trailer combination.

a (m)	3	$v_R^{\max}$ (km/h)	10
b (m)	1	$a_R^{\max} = -a_R^{\min}$ (km/h/s)	2
c (m)	4	$\alpha_F^{\max} = -\alpha_F^{\min}$ (°)	45
d (m)	1.5	$\omega_F^{\max} = -\omega_F^{\min}$ (°/s)	40
$v_R^{\min}$ (km/h)	-5	$\varphi^{\max} = -\varphi^{\min}$ (°)	60

The cost function used is $F(x,u,t) \equiv 1$, in (10), which means that the optimization criterion is minimum time.

The headland type here is selected so that the headlands boundary is straight and the width (w_h) and the angle (β_h) of headlands are variables. The situation in headlands is presented in Figure 32.

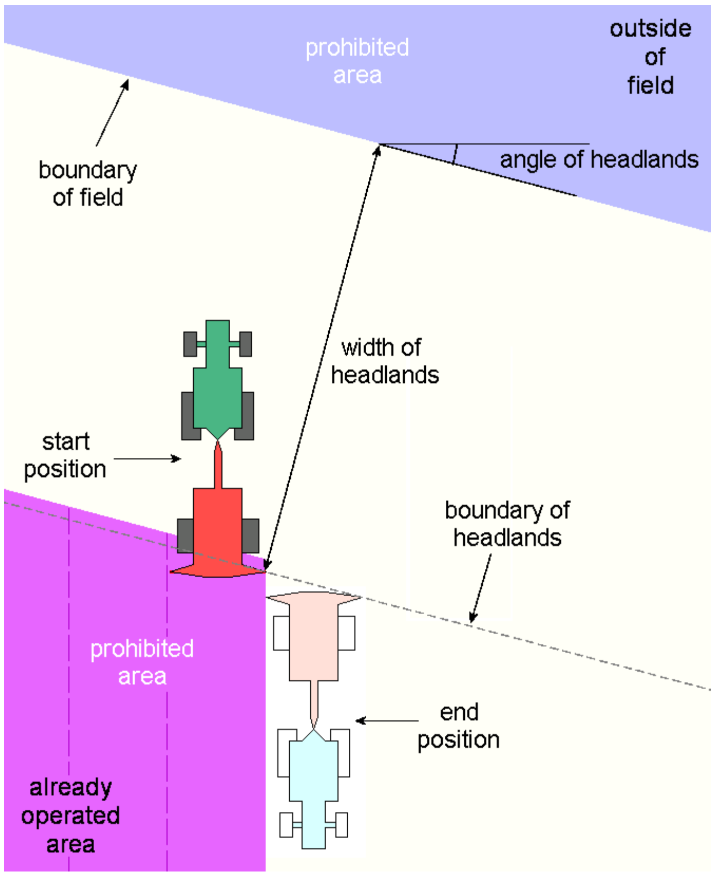


Figure 32. Headlands situation and parameters

The problem is to find the optimal trajectory (for states and controls) in order to move the vehicle from the start position to the end position. The problem is formulated as an OCP and solved with DIDO. A minor problem with solving this problem is that the solving takes quite a lot of time on a desktop computer, i.e. the solution of OCP with fixed headland parameters usually takes more than five minutes. That is why the trajectories have to be approximated in order to utilize the method in online applications.

One of the problems in solving the optimal control problem with a numerical algorithm is the periodicity of angle. The problem exists in the boundary conditions, where the end boundary condition differs by 2π if the turning has been made using reverse-back turning compared with pure forward driving. In reverse-back turning, the angle difference between start and end position angles is π, while in forward turning it is $-\pi$. This problem has been worked around by solving the OCP in both cases and selecting the best of them.

The example solution for both turning types is presented in Figure 33. The parameters in this example are $w_h = 17$ m, $\beta_h = -15°$. Twenty discrete points (nodes) in time are used.

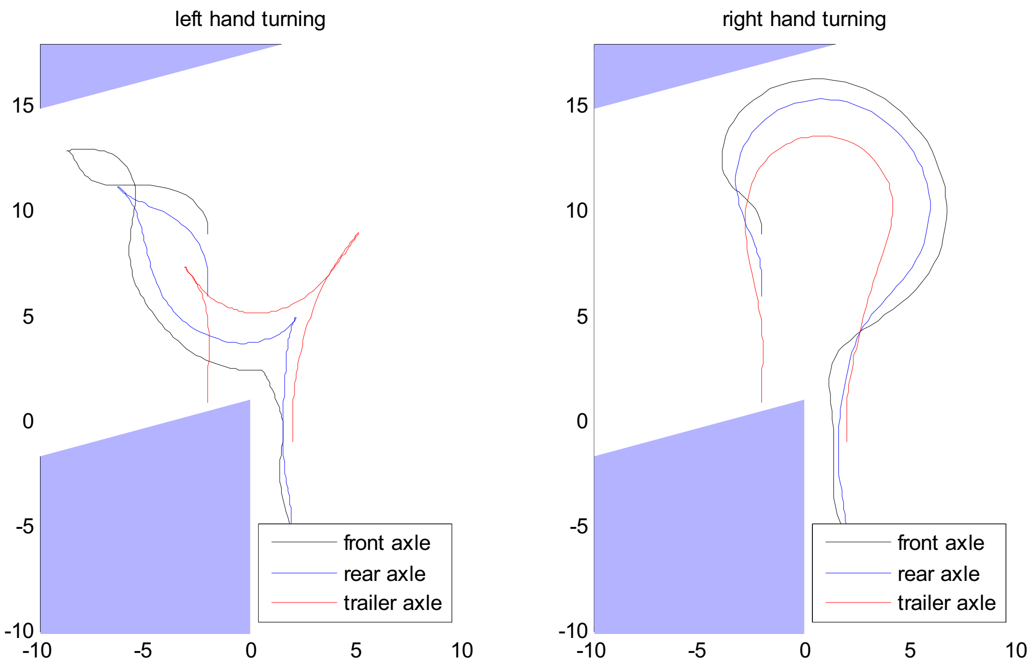


Figure 33. Example of optimal turning path

3.5 Effect of angle and width of headlands

The headland parameters used had a width of from 14 to 20 meters and an angle of from -60 to 60° with 7 and 8 samples, respectively. This range represents a normal range for the modeled machine in fields. Both the left- and right-hand side turning was solved. The turning time in both turning cases with varying headland angle and width is presented in Figure 34. Vertical bars correspond to time. The left side bar in each position represents the time spent in turning until the nose of tractor has turned left by π (180°) and right side bar right by π. The longer one of the pair of bars is in each position means a longer turning time. The difference of left- and right-hand side turning times is also presented in Figure 34. If the horizontal bar below points right, it means that right-hand side turning is preferred.

In Figure 34, it can also be seen that both turning types are not always possible or far from time optimal. It can be concluded that, with a narrow headland and small angle, the right side solution is not possible. In some cases, the numerical solution is not found at all. These results apply only to the vehicle used in this example; for other parameters, the results would be different. Nevertheless, the same methodology could be used to investigate the time-optimal turning problem with another vehicle.

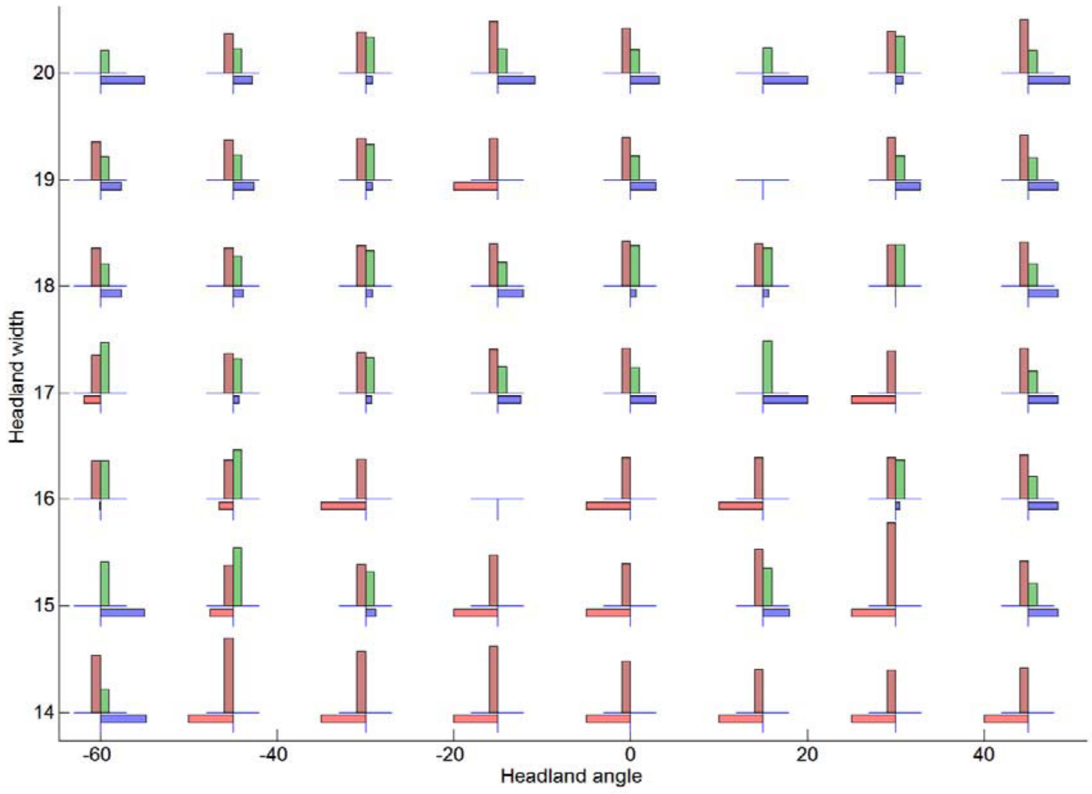


Figure 34. Turning time with varying headland angle and width. Red vertical bar represents time in reverse-back turning and green vertical bar time in loop turning. Horizontal bar shows which one is quicker. If bar is empty, the solver did not find a solution.

For online purposes and headland trajectory generation, "interpolation" of trajectories may be used, as the computational time to solve an optimal trajectory for one headland is high. However, the online application in real-time systems is beyond the scope of this thesis.

3.6 Path approximation

The solutions of the OCP solver are trajectories of states and controls. The vehicle control system can be built so that it can follow a certain 2D-path, so the reference path for the vehicle is needed. Here the piecewise Bezier curve method is used to approximate the path from OCP solver trajectories.

A Bezier curve is tuned with eight parameters, four of which fix both end points of the curve. The other four are used to modify the shape of the curve or to shape control points. A piecewise approximation is made so that the first Bezier curve is fitted to path data from the start to a position where the approximation error is within certain limits. Here the suitable error is assumed so that the maximum error in path is 5 cm. Next, the Bezier curve is fitted from the end of the last Bezier curve. Two consecutive curves have common parameters so that the end point (x, y) of the last and the start point of the next are the same, but so is the tangent. Therefore there are five parameters to be fixed within one place. The cubic Bezier curve formula is expressed in (14), P_A and P_D are the end points, P_B and P_C are the control points.

$$P_{BEZ}(w) = w^3 P_A + 3w^2(1-w)P_B + 3w(1-w)^2 P_C + (1-w)^3 P_D,\ w \in [0,1] \tag{14}$$

The piecewise Bezier curve approximation is presented in Figure 35, where a 2D path of the tractor's rear axle center point is shown with a blue line. The bold red line is the approximation and the blue line represents the original data.

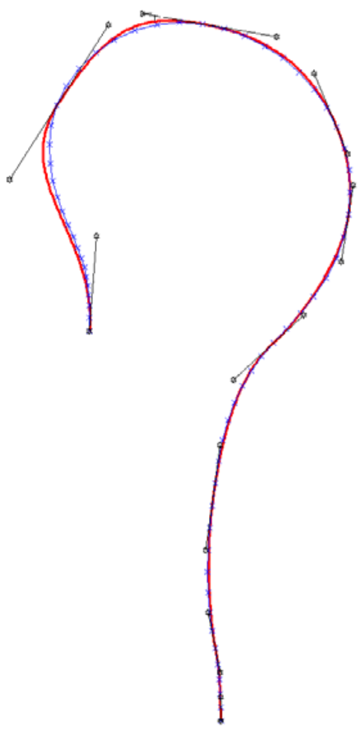


Figure 35. Piecewise Bezier curve approximation of the path.

The result is quite good; the only problem is that the Bezier curve cannot approximate circles well. This can also be seen in Figure 35.

3.7 Conclusion

Automatic path creation in headlands is important if the field machines are fully automatized. In this chapter, a tractor-trailer vehicle was modeled as a dynamical system, and both kinematic and areal constraints were determined. It is quite easy to extend this model to support two-link trailers, or trailers with even more than two links.

The path of a tractor-trailer combination in the headland can be solved using an optimal control problem solver. Numerical optimal control problem solvers still have some drawbacks, most important of which are the uncertainty of solution convergence and the numerical computation power needed from the computer. The path solved can be approximated with Bezier curves, and used in online applications.

4 Split and merge approach

4.1 Introduction

Tractors and other farming machines moving on the fields are traditionally driven by a human driver. The human driver designs the driving strategy of a single field by himself, without any assistance. He/she chooses the strategy on the basis of the task type, working machine and his/her own experience. On family-size farms, the strategy is based mostly on experience and remains the same over the years. If the field shape is not rectangular, or if there are obstacles, the development of the strategy is not so simple.

Autonomous field machines or robots will work the fields, sooner or later. The new issues for autonomous operation are safety, detection of failures, recovering after failures, and automatic refilling or emptying. If a human driver no longer operates the machine, automatic path planning is also needed; the robot has to find a route to execute the task. An optimal solution would be perfect, but a valid solution near optimal would be sufficient in most cases.

It is assumed here that the localization problem is solved and the vehicle's pose is known precisely. Nowadays, a commercial GPS-based real-time kinematic positioning system can achieve an accuracy of less than five centimeters and there are also accurate orientation sensors. Even if there are some reliability problems with these sensors, the assumption of precise positioning is valid. With respect to the environment, the field boundaries are known and they do not vary from year to year. However, the soil properties may vary and some operations are related to conditions in the field. To meet these conditions, adaptation or a behavior-based property is required, but this is beyond the scope of this thesis. Here the environment is assumed to be fixed and the current position and attitude measured precisely.

In this chapter, a new algorithm to solve the coverage path planning problem is presented. This approach is called *split and merge*. The split and merge approach is well known, but here that is extended to support the special requirements of agricultural field machines and agricultural fields.

4.2 Literature review on path planning

4.2.1 Traditional path planning

Roboticists understand the path planning as an algorithm that has to find a path from point A to point B so that no collisions with obstacles occur and the path is optimal with respect to any particular measure, traveling for a minimum time, for example. In robotics, path planning has been divided into two classes: qualitative and quantitative navigation. In qualitative navigation, the environment is structured so that the robot can identify landmarks and navigate, using them to follow a route. In quantitative or metric navigation, an exact map describes the world and it is not dependent on any particular viewpoint (Murphy 2000).

In agricultural robotics, the task is usually to cover the whole field, not only to go from point A to point B. This kind of path planning is so different from the traditional robot path planning that the algorithms are not directly applicable.

4.2.2 Coverage path planning

Coverage path planning is needed in several application areas, such as floor cleaning, lawn mowing, demining, painting, underwater searching etc. The research interest in mobile robotics (indoors and outdoors) has clearly motivated the research of coverage path planning.

Cao *et al.* (1988) defines the criteria for the region filling operation (for a mobile robot) as follows:

1. Robot must move through an entire area (cover the whole area)
2. Robot must fill the region without overlapping path
3. Continuous and sequential operation without any repetition of paths is required
4. Robot must avoid all obstacles
5. Simple motion trajectories (e.g. straight lines or circles) should be used (for simplicity in control)
6. An "optimal" path is desired under available conditions.

As Cao *et al.* (1988) notes: "It is not always possible to satisfy all these criteria for complex environments. Sometimes a priority consideration is required". This is very true in the case of agricultural fields, as they tend to be quite a complex, and the key question will be the prioritization.

The coverage path planning problem is related to *covering salesman problem*, a variant of the *travelling salesman problem* where, instead of visiting each city, an agent must visit a neighborhood of each city. This minimizes the length of travel for the agent (Choset 2001, Arkin and Refael 1994). As is well known, the traveling salesman problem is NP-hard. The computational time to solve the problem explodes when the dimension of the problem increases.

Many algorithms either implicitly or explicitly use cellular decomposition of the free space to achieve coverage. In *cellular decomposition*, the free space is broken into simple regions, which should guarantee the coverage. The cellular decomposition algorithms can be classified into three classes: *approximate*, *semi-approximate* and *exact* (Latombe 1991, Choset 2001).

The other way to classify the algorithms is *offline* and *online*. Offline algorithms rely on only the stationary information and the environment is assumed to be known. Usually online algorithms are needed if some kind of adaptivity to the environment is needed. Online algorithms may use real-time measurements. It should be noted that this thesis deals with the offline problem, but the online algorithms are also reviewed here. Online algorithms can also be called *sensor-based coverage algorithms*.

One approach to solve the problem is *randomize*. This is an approach that some floor-cleaning robots rely on. If you sweep the floor randomly long enough, it *should become* cleaned. There are also advantages to this approach (Balch 2000, Choset 2001). The main advantage is that no sensors are needed for localization. The only sensor needed is the one to detect the hit to boundaries. On the other hand, no expensive computer running complex algorithms is needed onboard. In the research of Balch (2000), it has been shown that the efficiency of robots with random algorithms is approximately 20% of those that use more advanced methods. Choset (2001) concludes that, if a robot with a random algorithm can be constructed at 1/5 of the price of one with localization and advanced path planning, it is cost-efficient. However, for agricultural field operations, it is difficult to think that randomized "algorithm" could be usable, as the cost of operating the vehicle (fuel & time) would be unthinkable, even much higher than the cost of construction.

In approximate cellular decomposition, the region is approximated with a grid, which approximately covers the region, and the algorithm is applied to the grid. The wavefront algorithm (distance transform) is conventionally used to solve the route from cell A to cell B in robotic path planning. Zelinsky *et al.* (1993) present an algorithm that solves the coverage path planning problem in a grid utilizing the wavefront algorithm. The main problem in that algorithm (for use in this context) is that it does not count kinematic constraints.

Acar *et al.* (2002) have introduced the use of cellular decompositions not only for path planning between two points, but also for coverage of free space; various patterns for decomposition are presented. Acar *et al.* (2003) discuss coverage path planning in relation to a demining application. In that article, the vehicle is considered holonomic, and two algorithms are discussed: exact cellular decomposition with back-and-forth motion and probabilistic method with online sensor data. The probabilistic method is not applicable here as it relies on real measurements. Choset and Pignon (1997) call the exact cellular decomposition, where back-and-forth motions are used, as *boustrophedon decomposition*; the word comes from Greek "the way the ox drags a plough".

Yang and Luo (2004) present a neural network approach to solve coverage path planning problem. Their application area is cleaning robots in nonstationary environments.

Actually, Arkin *et al.* (1993) have proved with the "lawnmower problem" that the problem of coverage path planning is NP-hard.

4.2.3 Agricultural path planning

Traditionally, farmers have planned the routes using common sense and their experience. However, there have been some guidelines drawn up for farmers, such as those of Sipilä *et al.* (1954), who suggest that, for malformed fields in ploughing, a *figure ploughing* method should be used. In this method, in a field with non-straight edges, the ploughing should be started at the center and be driven as a spiral. Other cases discussed are triangular and underdrained fields.

Palmer (1984) has analyzed driving efficiency. The overlapping of routes due to inaccurate driving is said to be about 9% of the implement width. Another source for inefficient driving is the circumventing of obstacles (Liu and Palmer, 1989). Palmer *et al.* (1988, 2003) have analyzed the routes driven with spraying vehicles with a positioning system. From this study it is concluded that 16% of driving distance could be saved. Palmer *et al.* (2003) presents a simple method to generate field courses, searching one driving direction and applying that to the whole field. However, the method to find the optimal direction is not reported. This simple method is somehow similar to exact cellular decomposition with a straight function. Liu and Palmer (1989) present an efficient course around the obstacles.

Gray (2001) has reported navigation development with respect to the orchard tractor. Orchards are not open fields. Trees form blocks, navigation around which is one problem to be solved. The whole task is another problem. Optimal patterns to apply to orchards are reported.

Driving patterns for field machines are analyzed and described in several publications, e.g. Witney (1996), Hunt (2001), Bochtis *et al.* (2006), Hansen *et al.* (2007). The studies usually assume that the field is rectangular (as the fields are in some countries) and patterns are developed to minimize the time consumed in headlands, as the most efficient way is not to turn to the next swath, as the turning radius of the vehicle may be large. In this way, it is indeed possible to minimize the field traffic (distance and time), but generally the pattern approach is applicable only to rectangular field plots. Rounding the field was analytically compared to back-and-forth swathing in a rectangular field by Pandey and Devnani (1987).

Path planning for agricultural machines between two states has been researched earlier. Noguchi and Terao (1997) have used neural networks to describe the motion model (dynamics) and genetic algorithms to optimize the path. Vougioukas *et al.* (2005) have proposed solving the path planning problem between two states under kinematic constraints using first random search to find a feasible solution and then applying optimization to find the optimal solution.

The coverage path planning problem for agricultural field operations has been discussed in the following publications. In Sorensen *et al.* (2004), a method is presented for optimizing vehicle routes by defining the field nodes as a graph and formulating the routing problem as the Chinese Postman Problem. In Stoll (2003), the idea of dividing the field into subfields based on the longest side of the field or the longest segment of a field polygon is presented.

Ryerson and Zhang (2006) have proposed using genetic algorithms to solve the coverage path planning problem. In this approach, the field is represented as a grid and the route to cover the grid is formulated as a sequence of cells. The genetic algorithm is applied to find the minimal distance and maximum coverage.

In case of agricultural robotics, the planning does not only include the path planning, but also the planning of the operation. *Mission planning* contains both the operation planning and the path or route planning (Reid 2004).

As the conclusion of the literature review, it can be said that the path planning of coverage type of task is still being researched and an optimal and universal algorithm has not been developed yet, so there is a need for further research of coverage path planning for agricultural fields.

4.3 Planning task

4.3.1 Background

The shape and size of fields vary a lot. Fields, especially in Finland, are usually bounded by other terrain types, such as forests, lakes etc., and shapes are far from orthogonal and convex, as was seen in Chapter 2. If the field is convex and it does not contain any obstacles, path planning for an agricultural task is quite simple. The main driving direction has to be found. The whole field is driven in that direction except for headlands, if needed. The selection of the main driving direction on the basis of the longest edge of field has been a rule of thumb for farmers. In the following algorithm, this rule of thumb based on common sense has been dismissed.

If the field is non-convex, which means that it has "bays", finding the optimal solution is hard. One possible way in which to solve the problem is to use the split and merge approach used in computer vision for segmentation. The field is split into simple shaped subfields that are convex or near convex. An optimal solution to drive one subfield can be found with some parallel swathing technique. If the shape of a subfield is, for example, rectangular, finding the optimal driving strategy is rather simple. The drawback of this method is that the output, the driving route, is not necessarily globally an optimal solution, but suboptimal.

Here it is assumed that the layout of the environment (field) is known. This can be assumed because fields are not changing over the years and the mapping has been done, at least in Finland. The requirements for a good coverage path planning algorithm are: suitability for all kinds of fields, for all kinds of machines, and computationally efficient enough in order to be solved in reasonable time.

Here the approach is such that a higher-level algorithm is used to divide a complex-shaped field into simple subfields in which the route planning is easy to do, with, for example, back-and-forth swathing (the so-called boustrophedon method) or with some other pattern. The algorithm is suitable for all kinds of agricultural field machines where the whole field has to be covered.

4.4 Algorithm

4.4.1 Objectives and definitions

The goal is to divide a complex-shaped field into simple-shaped subfields. The approach is first to search for the largest or the most efficiently driven subfield, remove it from the original field, and apply the same procedure for the rest of the field until the whole field is split. In the search of each subfield, the optimal driving direction is determined. In each step, the field is split into trapezoids, the trapezoids are merged to larger blocks and the selection is made using a criterion that takes into consideration the area and the route length of the block and the efficiency of driving. So, a *block* is a polygon that is constructed by merging the parallel and equal sides of two or more trapezoids.

4.4.2 Splitting

Crop-farming machines have a certain work width, which usually remains constant. The best efficiency and quality is reached if the driving lines are exactly side-by-side, there are no gaps, no overlapping and the turning in headlands is made in minimum time. Auto-guidance systems (light bars and autopilots) help the human driver to keep the machine in lane.

The proposed algorithm assumes that, in order to have a good driving strategy, the driving lines should be side-by-side and parallel to each other. Due to that assumption, a trapezoid has been selected as a prototype of the shape. This has two opposite sides in parallel corresponding to the driving direction and the other sides correspond to the edge of the field or the headland.

The field is split into trapezoids; this can be defined as an exact cellular decomposition (Latombe 1991). All nodes of the exterior polygon are projected at a given direction to all sides, and trapezoids are determined. If the field contains obstacles, the interior polygon nodes are also projected to all sides of the polygons. The example of triangulation is presented in Figure 36. In the field on the left, the number of trapezoids is 11 and, in the field on the right, the number is 18.

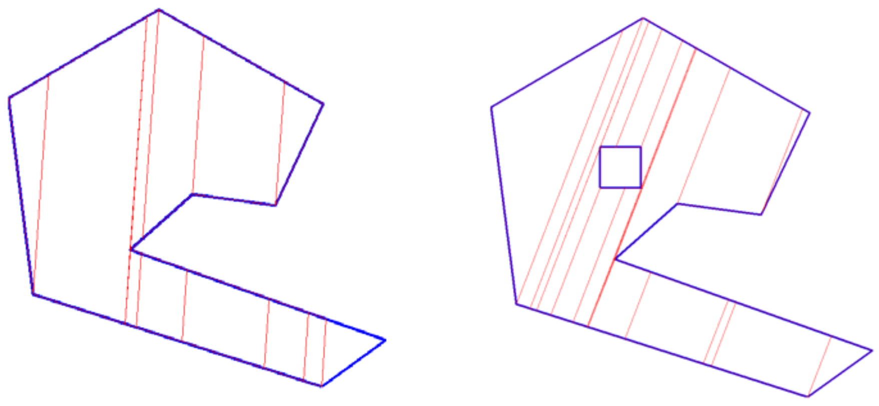


Figure 36. Two examples of triangulation.

4.4.3 Merging

After splitting the field into trapezoids, the next step is to combine as many of them as far as possible. The requirement is that two trapezoids have to have exactly matching sides and the angle of ending sides is not too steep. The second requirement prevents combining trapezoids that are far from a rectangular shape and should be handled separately in the later phases. The minimum angle between matching side and ending side is set to 20 degrees (90 degrees means a square corner). An example of merging trapezoids is presented in Figure 37.

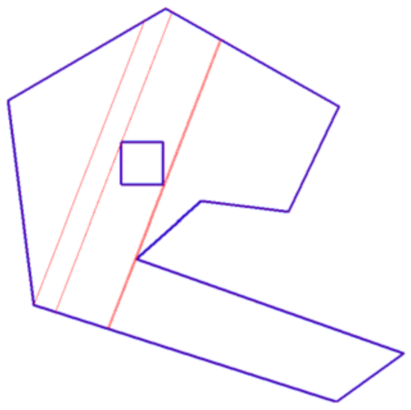


Figure 37. Merged trapezoids.

4.4.4 Selection criterion

After the blocks have been constructed, the best one of them has to be selected. The area of the block, the distance of the route fitting inside the block and the efficiency of driving are taken into consideration. The area is simply the area of the block. The distance is calculated using the tool width information, and the headland width is subtracted from that. The distance corresponds to the distance that can be driven at normal driving speed. The time consumed in the block is estimated from the calculated distance and using normal driving speed information. The time spent in headlands is added. The estimate of turning time in a particular headland angle can be calculated by using optimal control techniques, for example. These are described in Chapter 3.

Efficiency is, in practice, the one which is desired to be maximized, but that would easily lead to a situation where narrow long strips were selected first, and the direction of that strip with a small area may ruin the global efficiency. Therefore the other two measures are also needed. All the measures (area, distance, efficiency) are normalized and the cost is a weighted sum of those. The weights tuned by hand are: efficiency 65%, area 15% and distance 20%. These weights are used in the examples below.

If some subfields are already selected, a small bias is added to the calculated cost in the directions of them. This prevents adjacent subfield directions not to differ from each other by small angles. With most cropping machines, a small correction in direction leads to inefficiency and to quality losses.

4.4.5 Searching driving direction

Splitting the fields into trapezoids and merging them into blocks is made in a particular direction. However, the direction is not known and it has to be solved. The characteristics of the blocks do not change smoothly when the direction is changed in infinitesimal steps, so the cost function of the search is not smooth. This means that all possible directions should be gone through (between 0 and 180 degrees) and it takes a lot of calculation time. Here heuristics have been used.

The search algorithm is as follows:

1. Cost is calculated in 6 directions, 0°, 30°, 60°, 90°, 120° and 150°.
2. The three best directions are selected, others are dropped
3. The step size in direction angle of search is halved
4. New search directions are added to both sides of the three best directions
5. Cost is calculated for directions that are not yet calculated
6. If the goal resolution is reached, exit, otherwise go to step 2

After 5 loops, the resolution is below one degree which has been found to be sufficient.

This heuristic search algorithm was tested with random fields and the solution was compared to the brute-force solution with the same resolution. The result was that over 97 % of the solutions matched and only less than 1 % were far from the global maximum.

4.4.6 Headlands

As described above, the headland width is reduced from the main driving lines when calculating the efficiency. In this way the solution will be correct, but in some cases a headland is not needed. If the direction of blocks after the first iteration varies from each other, it is evident that one end of the block is common to the parallel side of the other block and generally then the headland is not needed. The other case when headland is not needed is when a block that has a very steep headland angle, e.g. below 15° (90° means again a right angle). In this case, the headland can be driven by bending the driving line. The number of swaths needed in the headland is an input variable for the algorithm.

4.4.7 Prohibited driving directions

For some environments and some operations there are limitations to the driving direction. This can be taken into account in the path planning algorithm by defining a *prohibited region*, in which a range of prohibited driving directions are set (in degrees). If the set of prohibited driving directions is not uniform, multiple prohibited regions may be used.

In the algorithm, the prohibited regions are taken into account in the split phase. If the current search angle is in the angle range of the prohibited region, the prohibited region is handled as an obstacle or interior polygon. After selection, in the removing phase, the prohibited regions are cropped if needed. It is required that prohibited regions are inside the field region.

4.5 Test results

4.5.1 Tests with real fields

A subset of Finnish fields was used to test the algorithm. The set size was about 1500 fields and the average area was 3.87 ha. All fields are from Uusimaa region and the subset was randomly selected from the data set where field area was over 1 hectare.

In test fields, the machine-specific parameters were: toolwidth 2.5 m, headland width 7.5 m and the driving speed 10 km/h.

The algorithm was able to find a solution for all fields.

All the solutions were checked by hand and it can be said that almost all of them are valid, but it is difficult to say how near or far they are from optimal. Problems occurred when the field contained many small obstacles, which resulted in solutions which were very different from the solutions without obstacles.

The number of subfields varied from 1 to 67. The histogram of the number of subfields is presented in Figure 38. When the number of subfields is low, the number tends to be odd. Generally, it can be said that the smaller the number of subfields is, the better the overall efficiency is.

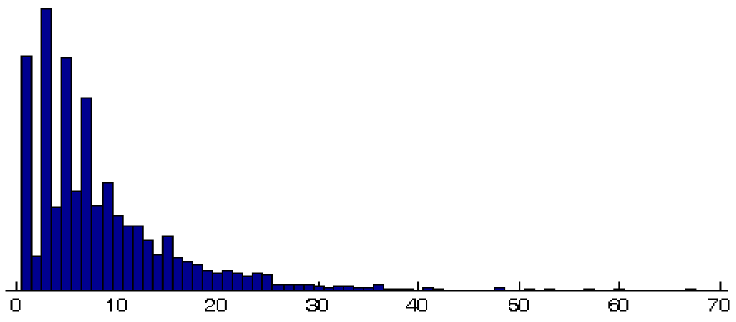


Figure 38. Number of subfields in histogram.

It is difficult to measure the overall optimality of the solutions as the optimal solution is not available.

It can be said that, for convex fields without any obstacles, the solution is optimal along with the criterion used, because only one subfield exists. It was found that, if a convex field has a clear longest edge, the direction given by the algorithm was congruent with the longest edge of the field. Solutions for some convex or near-convex fields are presented in Figure 39. In the plot on the left, the driving direction is congruent with the longest edge of the field. In the plot on the right, the field is not convex, and there exists no clear longest edge, but the algorithm has found a solution that can be assumed to be a good one.

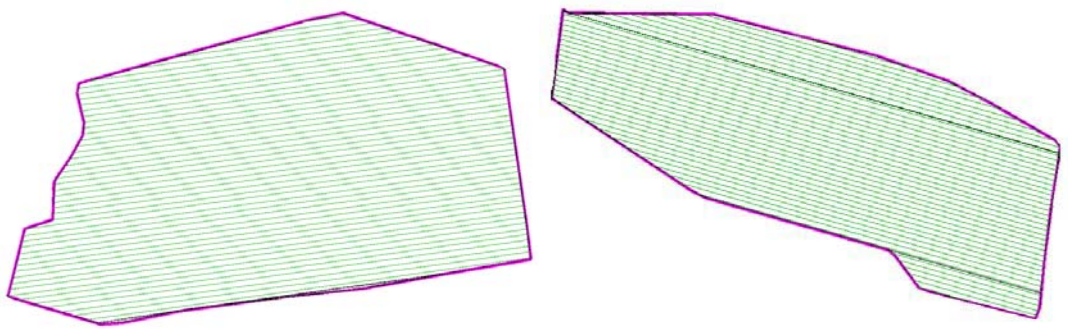


Figure 39. Easy fields

If the field is not convex or near convex, but has clear corners and clear straight edges, the solution is easily found. One such field is illustrated in Figure 40 (left). The plot on the right in Figure 40 is near convex, but has an obstacle in it. The solution mainly follows the longest and the second longest edges.

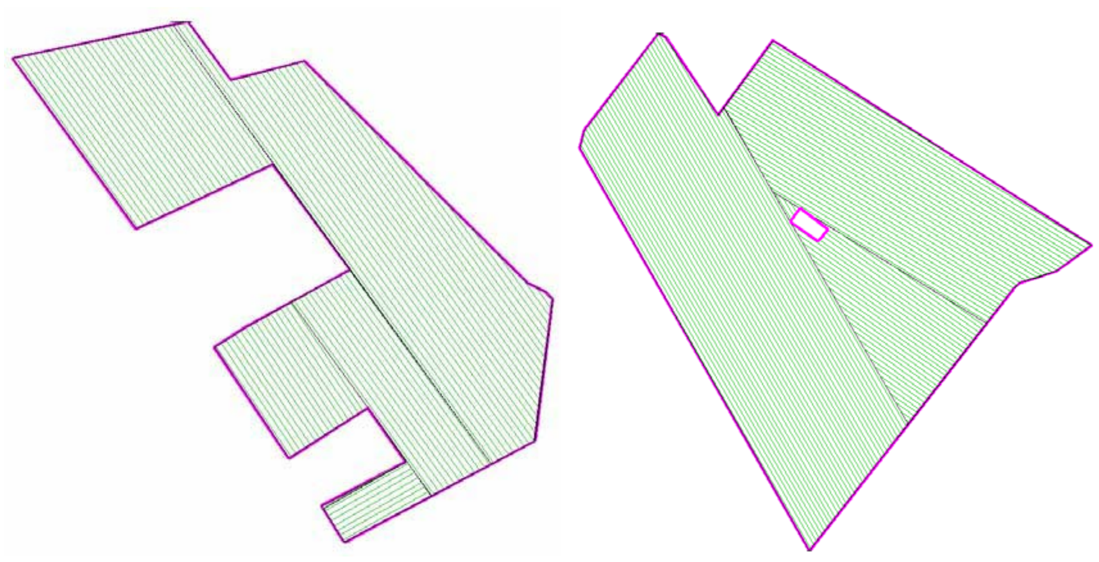


Figure 40. Typical fields

In such fields that are narrow, long and curved without any clear straight edge, the optimality of solution is not clear. If the machine allows curved lanes, then it may be reasonable to follow the edges of the field. In Finland, this kind of field shape is not rare, because some fields are limited by nature (forests, rocks or lakes), not by the fields of the neighbor. Two examples of curve-edged fields and the solutions are presented in Figure 41.

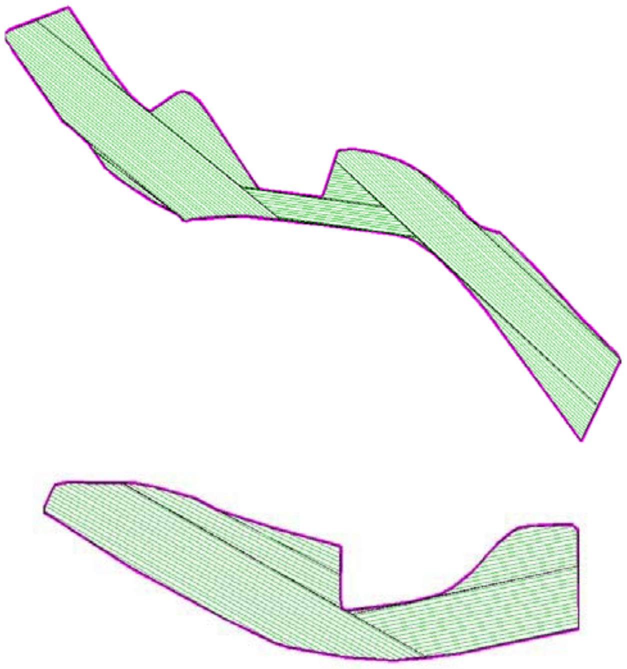


Figure 41. Curved fields

There exist some complex-shaped fields in the test data. Those fields contain many "islands" and "bays" and the driving takes a lot of extra time, regardless of the path. The solutions for two different complex-shaped fields are presented in Figure 42 and Figure 43.

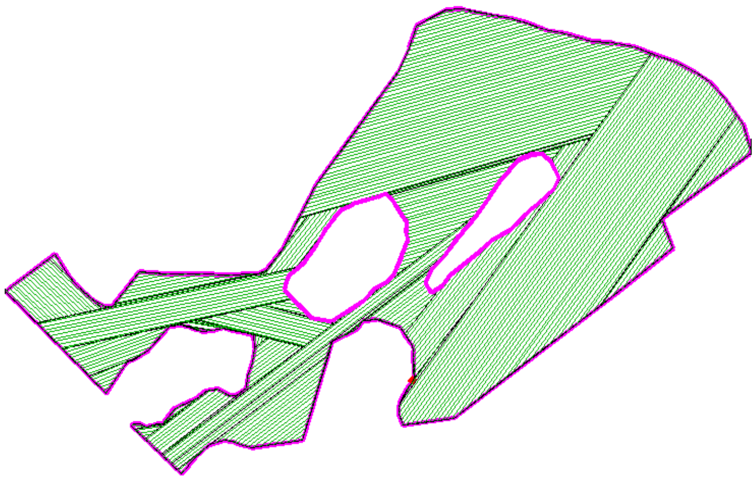


Figure 42. Solution for incompact field

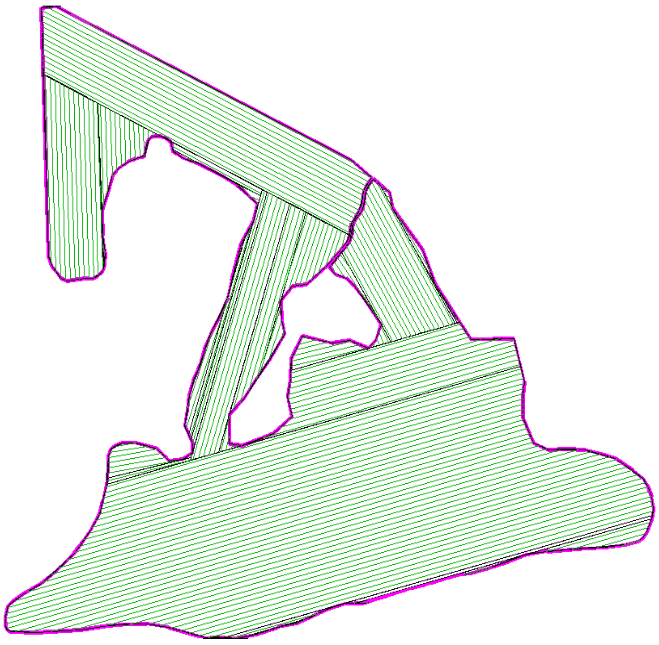


Figure 43. Solution for incompact field

4.5.2 Headlands

Automatic determination of headlands was developed. In the figures below, the headlands are drawn in blue, while the main swaths are drawn with green. In Figure 44, an H-shaped field is presented with the solution. At first, the algorithm finds two long vertical blocks on each side and finally the horizontal block between vertical blocks is handled. The headlands are needed only at the end of vertical blocks and they are automatically generated.

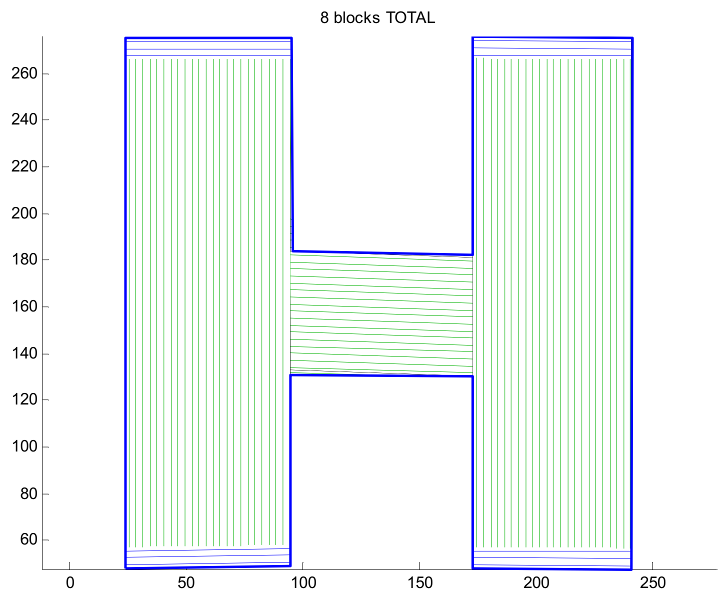


Figure 44. H-shaped field with headlands

In Figure 45, a field with many bays is shown. As can be seen, the main driving direction was determined on the basis of the largest block in the middle. For three of the four bays, the same driving direction is found to be top-rated (NB: a small bias was given to the direction of neighboring blocks). The headlands are needed in most edges, but if the direction of edges is near enough to the direction of swaths (in these tests 5°), the headland is not laid.

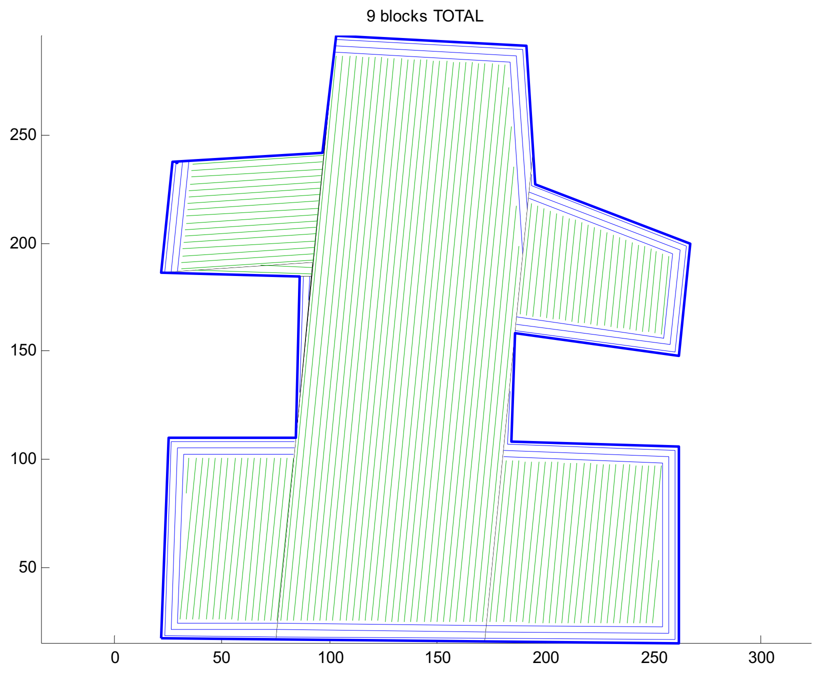


Figure 45. Field with many bays

4.5.3 Prohibited regions

As described, with the prohibited regions it is possible to define impossible driving directions due to height variation and machine properties or to define unwanted, inefficient driving directions.

In Figure 46, a C-shaped field is shown. On the left, there is a solution without any prohibited regions. The final solution consists of five blocks, saving two headlands. On the right, a fictional escarpment (steep slope) is inserted into one corner; this is marked with a dashed line and small red triangles. The "bow" represents the forbidden driving directions. This means that the driver does not want to drive the escarpment up-down-up, but diagonal driving is allowed. Maybe his/her tractor does not have enough horsepower to drive it uphill. However, it can be seen that the solution found without the prohibited region has changed dramatically. The driving direction is changed all around the field plot and headlands are required all around the field.

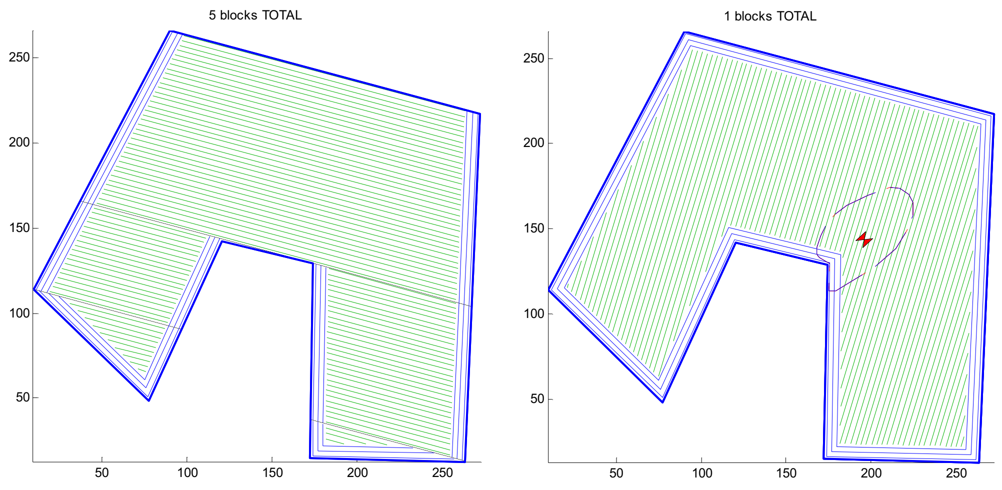


Figure 46. C-shaped field without and with prohibited regions

Let's consider another example. In the Northern Europe most of the field plots are underdrained. *Underdrainage* is important especially in soil types that are not transmitting water easily. In certain field operations, such as ploughing, it is not recommended to drive in the same direction as that in which the pipes are laid; a ploughed furrow is also a kind of "pipe". When the furrows and pipes cross, the effect of drainage is at its best. In Figure 47, a field with an underdrainage system is presented. A bold blue dashed line represents the collector pipe in the drainage system and blue lines are lateral pipes. Two prohibited regions are marked with red dashed lines and red "bows" are marking the forbidden driving direction range.

In Figure 48, on the left, the solution of the algorithm without taking underdrainage into account is presented and on the right it is considered. In both cases, one dominant driving direction exists, but the right one fulfills the requirements of a prohibited region. Actually, in this case, the efficiency is almost the same in both cases. In simulation, the right one is only 0,2% worse than the left one, if using total driving time as a measure. Naturally, this fact applies only for this particular field.

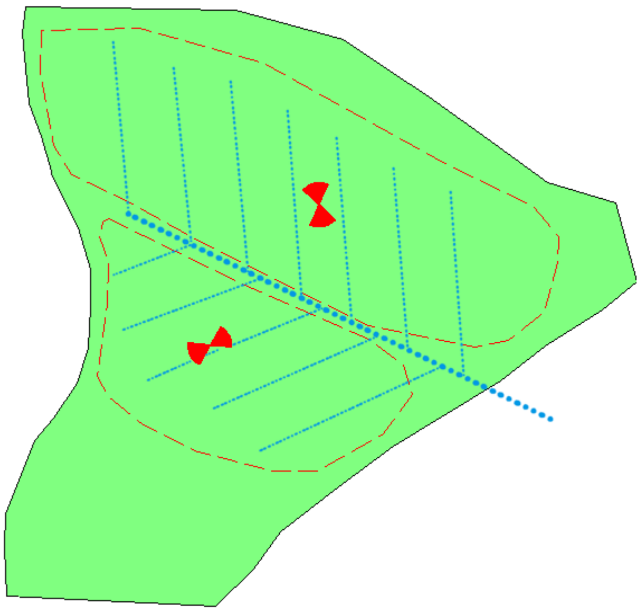


Figure 47. Field with underdrainage system

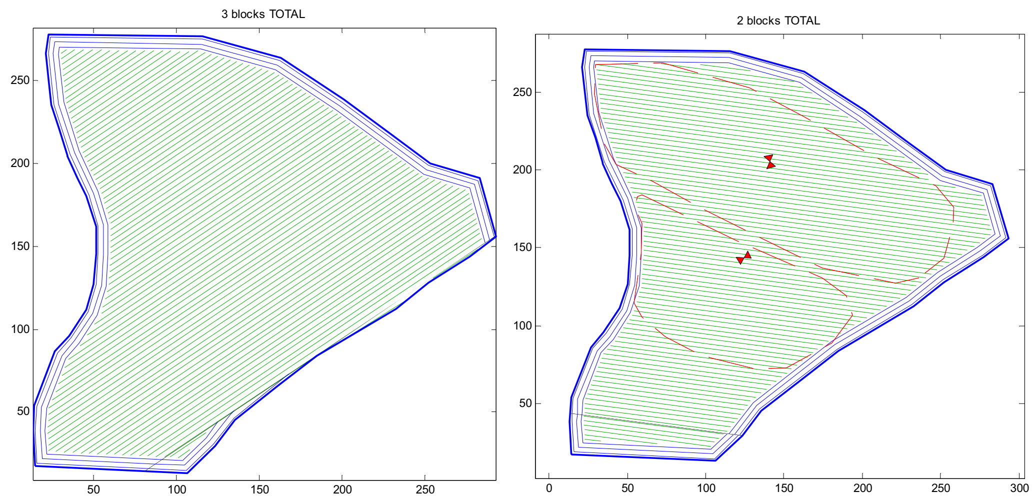


Figure 48. Solution without and with underdrainage

4.6 Conclusion

Path planning for robots working in fields has not been solved yet. Various algorithms for path planning have been introduced, but they are still more like a collection of algorithms.

In this chapter, an algorithm for dividing a field into subfields is presented. The shape of a subfield is simple, so it can be driven using parallel swathing techniques. The algorithm relies on splitting the field into trapezoids, merging them into larger blocks, using a search algorithm to select the best driving direction and recursing the search until the whole field is divided. The algorithm can be classified into exact cellular decomposition. Trapezoidal decomposition (a common form of exact cellular decomposition) has been utilized as a part in the algorithm.

A set of Finnish fields have been used to test the algorithm. The algorithm can find a solution for all fields. It is difficult to measure the overall optimality of the solutions.

One drawback of the algorithm is that it can only use straight driving lines. Some fields do not have straight boundaries. Especially in fields which are narrow, long and curved, the solution may be far from optimal. Currently, the algorithm takes the headlands into consideration in the criterion, but they are not considered as their own blocks in the result. The headlands are automatically generated where needed. With prohibited regions, the previous operations, underdrains and steep gradients can be taken into account. In these regions, certain driving directions are marked prohibited.

In some agricultural tasks there are also some limitations for selecting the driving direction; in plowing the direction of underdrains has to be considered, for example. Also, refilling or emptying the machine should be included in path planning. A general usable coverage path planning algorithm should be able to adapt to specific requirements of the agricultural task.

As a final remark, it is noted that this part of the research was reported originally in publications Oksanen *et al.* (2005) and Oksanen and Visala (2006a). There seems to be some research being undertaken currently that, in some sense, extends this work. Some results are published by Jin and Tang (2006).

5 Predictive recursive online approach

5.1 Introduction

In the previous chapter, a top-down approach was used for solving the coverage path planning problem with agricultural machines. The drawbacks of the algorithm mainly concern the straight driving lines. This is a considerable limitation, as there really exist many fields with no straight edges. Consequently, the quality of the results may be unsatisfactory, as it was seen in the previous chapter.

As it is very hard to generalize the trapezoid prototype used in previous chapter to some other shape that would allow curved edges, a somewhat complicated approach had to be developed. The underlying idea in this proposed algorithm is to follow the shapes of field edges, and not to force them straight. This is how some farmers really operate most of their fields, following the shape of some edge or rounding the field as a spiral. However, the simple back-and-forth approach applied to the longest edge or rounding may not be optimal, so this is a little bit generalized.

The centre driving lines of swaths can be generated from field boundaries using offsetting. As the fields here are polygons, the centre driving lines are constructed of line segments. Somehow, the vertex, the machine can not drive with the implement down, must be detected; and the turning has to be made.

Theoretically, a full combination of all the remaining swaths should be simulated in order to find the best next swath. However, this cannot be done in a real situation, as the search space would be so huge that no computer could solve the problem. The key question is that how the search space could be restricted in an "intelligent" way — so that the computational time would be sensible. However, the search covers a larger search space than one swath.

5.2 Definitions

To begin with, certain definitions that are required to understand the algorithm are given. *Polyline* is a continuous line composed of one or more line segments. A c*losed polyline* is a polyline where the starting point and ending point is common. V*ertex* is the point where polyline segments end. *Edge* is a line segment in a closed polyline. One vertex is shared by exactly two edges. *Polygon* is a synonym for closed polyline. *Critical vertex* is a vertex where the vehicle must stop the operation and make a turn.

5.3 Algorithm

5.3.1 Assumptions and limitations

Some assumptions and limitations are first set:

1. all the swaths must be side-by-side,
2. the turning times and path lengths are known a priori for all headland turnings for the machine being used, or they are very quick to calculate,
3. the working width is constant.

For assumption 2, precalculation and sampling may be used, see, for example, the method in Chapter 3 or Noguchi *et al.* (2001). This precalculation applies if the machine properties remain the same. The algorithm has to solve the turning time so many times that a look-up table is needed in order to keep the computing time reasonable.

5.3.2 Polygon offsetting

The most important sub algorithm is the so called *polygon offsetting*. Its aim is to move each edge of the polygon inwards (or, in some cases, outwards) so that the perpendicular distance between the subtracted polygon and original is as wanted. This problem is analogous to the field operation where the field is rounded once, doing some operation, and the inner boundary of the operated area is to be identified. This is a well known problem in computational geometry. There are several methods to solve this, see, for example, Yang and Huang (1993), Choi *et al.* (2001a) and Choi *et al.* (2001b). Here the *straight skeleton* method is used, see Aichholzer *et al.* (1995) and Felkel and Obdrzalek (1998). The straight skeleton method can handle convex regions as well as holes in the region. Implementation of the straight skeleton method by Felkel and Obdrzalek was used with their permission. Offsetting based on the skeleton was implemented in this research.

5.3.3 Basic idea

Let's consider a simple rectangular field, see Figure 49. Let the initial state of machine be I, both the location and the orientation in 2D space. The A-B-C-D polygon represents a region, the interior of which needs to be operated; these points may be considered also as critical vertices. As it was set, the requirement that each new swath must be side-by-side with some previous swath, a restricted set of movements exists. The problem is to search the best route.

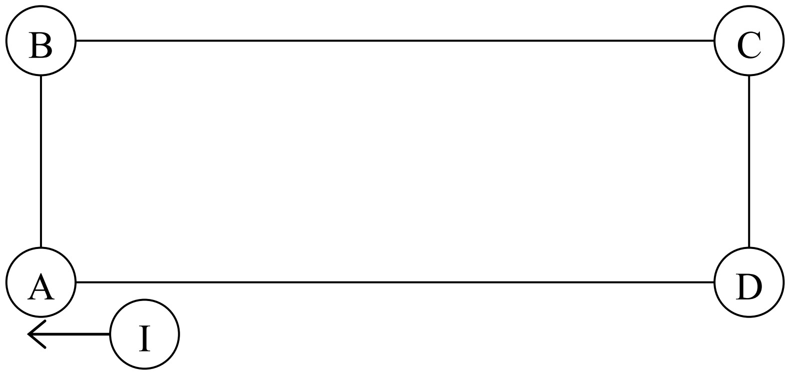


Figure 49. A simple field

In order to find the global optimum with respect to a certain cost, all possible routes to the end have to be compared, e.g. I-A-B-C-D-A'-B'-C'-D'-A''-... until the whole region has been operated. The computational cost of this problem will increase exponentially as the number of corner points increases and/or area/working width ratio increases.

In control engineering, a similar problem arises in optimal control. A search from all possible control functions is generally impossible. Optimal control methods that minimize certain cost criteria exist, but in real-time operations and nonlinear systems these are not practical. *Model predictive control* (MPC) is a way to improve feedback control: at a certain time step, the system behavior is analytically predicted over the *prediction horizon*. The control law can be solved using predicted behavior and by minimizing the criterion over the *control horizon*. Only the first action of the solved control function is applied at each step; at the next time step all the computations are repeated. Model predictive control is also known as *receding horizon control*. It is known that generally MPC does not result in optimal control, but, in practice, the control performance is very good (Maciejowski 2002).

The approach here is inspired by model predictive control; the same analogy is applied to path action search. The "control horizon" and "prediction horizon" are actually equal in tests later, but the "prediction horizon" could be longer. Roughly, it should be a multiple of the "control horizon". In this thesis the "control horizon" is called *search horizon*.

The search horizon is defined as follows:

- starting vertex is the nearest critical vertex to current position,
- starting direction is free, clockwise (CW) or counter-clockwise (CCW),
- stop when near starting point A' (offset A),
- zero or one reversion in direction,
- segments may be skipped, but not all,
- if a segment is skipped in one direction its pair must also be skipped after reversion.

Let's get back to our simple field, Figure 49. The possible routes could be I-A-B-C-D-A'; I-A-D-C-B-A' rounding the whole field, the subsets of those with skipping edges. The routes with single reversion in CW direction would be I-A-B-C-D-C'-B'-A' ; I-A-B-C-B'-A' ; I-A-B-A', plus the subsets. Similarly, the routes would be calculated in CCW direction.

This will lead to

$$2\cdot\left(\left(2^{N}-1\right)+\left(2^{N-1}-1\right)\right)=3\cdot 2^{N}-4 \qquad (15)$$

number of choices, where *N* is the number of critical vertices. The multiplier 2 is the number of directions; the first part is for rounding the field and the second part for reversing at different points. For *N*=4 (in our case), the number of choices will be 44. For *N*=7, it is 380 and for *N*=10 it is 3068, for *N*=15 it is about 100000 and for *N*=20 it is over 3 million. So the reasonable number of critical vertices is around 10–13, naturally depending on computing resources and on the complexity of the efficiency function.

5.3.4 Generating routes

In order to generate all possible routes in the search horizon, the polygon offsetting algorithm is utilized. The region boundary is offset by the half of working width (in a practical application, the overlap must be considered), offsetting is made three times. The first offset gives the first centre line of the swath for the machine, the second is a boundary of the first operation swath, and the third is for the reversive centre line.

5.3.5 Efficiency function

All the driving is divided into two groups: working and turning. The latter contains all driving where the implement or some functional part of machine is not in an operational state. The efficiency function is calculated for all possible routes in the search horizon. Let the route lengths be s_W, s_T, for working and turning, respectively, and

t_W, t_T for route driving times. The driving speed may vary. The cost, or the time efficiency function, is

$$J_{route} = \frac{\sum_i s_W^i}{\sum_i t_W^i + \sum_i t_T^i}, \tag{16}$$

where the sum operates over route segments. In other words, the cost function measures efficiency of route-operated area divided by operation time.

5.3.6 Selection of best route in search horizon

The time efficiency function (16) is calculated for all possible routes and the route having maximum efficiency is selected. This phase of the algorithm can be speeded up using approximate turning times and turning path lengths, utilizing precalculation and look-up tables. When the best route is selected, a more computationally intensive and more accurate trajectory planning algorithm may be used. As stressed above, only the first segment of the selected route is applied at each step of the algorithm.

5.3.7 Simulated driving

In the simulation phase, the route to be applied in each step must be subtracted from the original region, representing the field still not operated. This phase turns out to be difficult, because the shape and characteristics of the offset region may change dramatically. Basically, a new boundary of region representing the route to be applied is found by finding the next offset polygon line segments and replacing the previous outer boundary with that, and also removing the head and tail line segments of the operational area from the original region. Treatment of special cases is needed if the offset method is not working.

5.3.8 Simplifying routes

If the original polygon(s) representing the field has many vertices, there are two ways to speed up the calculation of the algorithm. One is to reduce the number of vertices, by approximating the polygon with another having fewer vertices, and the other is to use the original polygon but merging line segments to polylines. Here the latter way is applied.

In merging line segments, a condition is needed. The natural condition to merge or not to merge is the curvature of polyline representing region boundary. The curvature limit and the maximum turning radius of the machine in operational state have an inverse relation. The problem is that here the region format of a polygon was selected and the curvature cannot be calculated (appearing singularly), because the curve should be smooth in order to calculate curvature. If only the change in direction is calculated (traveling along polygon around), it may lead to incorrect results if the line segments are very short and direction change is very small in all cases.

Several different conditions were tested; the following seems to work best. Let *r* be the turning radius of machine (note: in operational state). For every vertex in the polygon, a circle with radius 2*r* is drawn, see Figure 50. In both directions from the vertex, the first segment that intersects with the circle is selected and the curvature estimate (direction change) is calculated. The limit for this value is a tuning variable; in tests, thirty degrees was used with *r*=6 m. This algorithm can detect the critical vertices where the turning is needed.

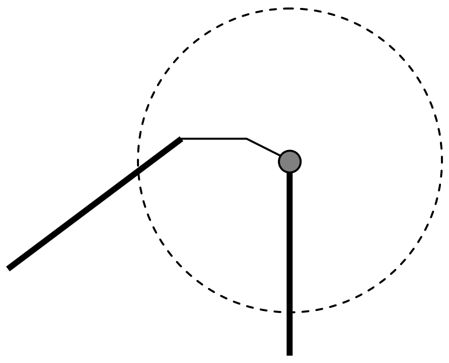


Figure 50. Merging condition

5.3.9 Refilling and emptying of machine

Refilling or emptying is a crucial part of field operations. Seed drills, fertilizer spreaders, sprayers, dung and sludge spreaders, lime spreaders and several other implements spread various input materials into the field and usually tanks of those implements have to be refilled during the operation. Combine harvesters, forage harvesters and other crop harvesting machines collect the harvest and those machines empty their tanks usually to some trailer or truck in the field. The refilling or emptying rate depends on the rate and tank size of machines. For sprayers the refilling rate may be 15 hectares and for combine harvesters it may be 0.5 hectares. In this subchapter, automatic planning of refilling and emptying (hereafter called *servicing*) during the operation with the following subalgorithm is presented. It is assumed that the application rate / yield is known, the application rate may be variable within the field, but yield is assumed to be constant as it can be measured only after the operation has been applied (harvested) in a certain position, and it may be hard to predict. However, if the prediction method is available, variable yield can also be handled with this algorithm.

The service support to the original path planning algorithm is presented through an example. Let's take one generated route at a time. A schematic diagram in a case of three segments is presented in Figure 51. The vehicle is currently at state S (*x,y,heading*). The swaths to be operated are A_1–A_2 , B_1–B_2 , C_1–C_2 and the route always includes the return to the first point. As without servicing, the time, working distance and turning distance are calculated through simulation for this scenario.

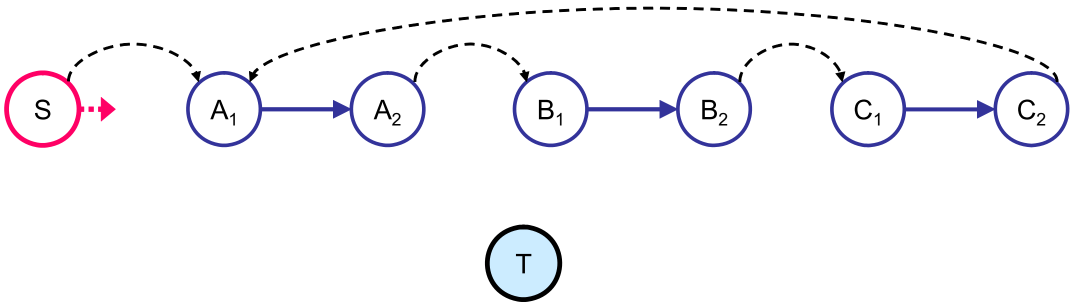


Figure 51. Tanking scenario

Now the tank fillment level is also simulated. At the current state, there is also a variable for tank fillment level. The tank fillment level is simulated over the generated route. If in the simulation it is noticed that the tank reaches the critical level (either in harvesting the tank is full, or in spreading the tank is empty), the following procedure is applied. Otherwise, if no critical level is reached, the algorithm works as normal, just the tank fillment level is simulated over the action. But if the tank reaches the critical level, then the segment in which it happens is searched. If this happened in our example in swath C (in Figure 51), the original generated route would have been replaced with all such routes (which handle the same segments) that include servicing before the segment where the critical level is reached. The service point is marked with *T*. In this example, the new generated routes would be (S-T-A_1-A_2-B_1-B_2-C_1-C_2-A_1), (S-A_1-A_2- T-B_1-B_2-C_1-C_2-A_1) and (S-A_1-A_2-B_1-B_2-T-C_1-C_2-A_1). After this modification of routes, the route is resimulated and checked if it requires another service (this is a very rare situation). If service was needed, the same procedure would be applied to the generated route with already servicing, but the new services would be simulated only after the first service. After these modifications of generated routes, it is guaranteed that in every generated route the operation can be performed safely. After the selection phase, the service is performed if the servicing is scheduled before the first segment, otherwise the first segment is applied as usual and no servicing is performed in that step.

For every generated route:

1. simulate the degree of filling of the tank (integrate over application rate map),
2. if the tank is empty or full after simulation, service is needed:
    a. find the route segment in which zero crossing happens,

b. replace the original route with all such routes that cover the same segments and include the visit to the service point before the zero crossing happens,
c. check if another service is needed.

In the simulation phase, three parameters are needed: application rate (negative for spreading, positive for harvesting), tank capacity and safety margin (measurement or prediction errors).

If multiple service points exist (in large fields this may be practical), in the replacement phase, the quickest visit to the service point is selected in every case (T is replaced with $T_1...T_N$). Non-stationary service points, such as tractor-trailers moving alongside a combine harvester, are not supported in this algorithm. However, the slowly moving service points are supported. The condition is that the service point must be stationary over the prediction horizon, they may move during the swath and the movement should be predictable.

5.3.10 Non-convex fields

If the field is non-convex (concave) or contains obstacles, the region representing the non-operated area will easily split into separate regions, unless the region is no longer uniform. In these cases, the route segment generation is similar: the region offsetting is made for all regions, separately.

The actual problem in these cases is that the number of possible routes will be large, as the "jumps" from one region to another must be free. As the number of critical vertexes increases and the number of separate regions increases, the number of possible routes explodes rapidly. As the problem itself is NP-hard, this is also. This means that, in theory, all the possibilities should be checked in order to find the best one. With heuristics, it is possible to limit the search space, but hard to prove that some heuristics does not ever close out the best solution.

During this research, many kinds of tricks or heuristics were investigated. One was to create links between separate regions and allow jumping using only them, but that did not work well generally, only in some special cases. The other trick was to limit the jumping only to the nearest region, which was in some cases good, but still the search space was exploding very rapidly. The next one was to allow only one segment to the route after the jump, that seemed to work well, but was inconsistent with the original idea of "make a route and come back to the start point". The trivial solution for this is to handle each region separately, complete one at a time. However, doing so usually closes out the best solution from the search space. It was always possible to find a counter example where the heuristics did not work.

No universal trick to drop down the dimension of search space was found during this research. This was quite disappointing, as a lot of time was used to find such heuristics. In the following examples, the approach of "jump and allow only one segment after jump" is used. Additionally, the shortest segments are removed from the search space if the number of segments is over 10; the way is to remove all except for the nearest two segments and the longest eight. In this way, the best route could be lost. Nevertheless, this way is followed for the sake of the computational time taken in solving the path planning problem.

5.3.11 Summary

Here are the main steps of the algorithm described below as pseudo code:

```
CurrentState = InitialState;
NonOperatedRegion = TheField;
WHILE ISEMPTY(NonOperatedRegion)
     NonOperatedRegion = SIMPLIFY(NonOperatedRegion);
     Skeleton = MAKE_SKELETON(NonOperatedRegion);
     OffEdges = DO_OFFSETTING(Skeleton);
     CriticalVx = DETECT_CRITICAL_VERTICES(OffEdges);
     Lines = CONNECT_OFFSETTED_EDGES(OffEdges,CriticalVx);
     RetLines = SOLVE_RETURNING_LINES(Lines,OffEdges);
     RouteSet = GENERATE_POSSIBLE_ROUTES(Lines,RetLines,CurrentState);
     RouteEff = CALCULATE_EFFICIENCY(RouteSet);
     pBestEff = SELECT_BEST(RouteSet,RouteEff);
     BestRoute = SOLVE_EXACT_TRAJECTORY(pBestEff,RouteSet,CurrentState);
     IF BestRoute.FirstSegment.ServiceNeededBefore
          CurrentState = DO_SERVICE(BestRoute,CurrentState);
     END
     [NonOperatedArea,CurrentState] = OPERATE(BestRoute.FirstSegment,
                                             NonOperatedArea,
                                             CurrentState);
     CurrentState = UPDATE_TANK(BestRoute.FirstSegment,CurrentState);
END
```

5.4 Test results

5.4.1 Notes

The machine-specific parameters in the results below are: working width 3 m, driving speed 10 km/h in working phase, 6 km/h in turnings, minimum turning radius 6 m in headlands and 10 m during operation – if nothing else is said.

Headland or turning area is not considered yet. However, for many operations it is suitable to drive round the field three times for example, and then apply this algorithm.

In some of the following figures, the colors are used to clarify the order of swaths generated during the algorithm. The first swath is marked with blue and the last with red.

5.4.2 Simple fields

It is natural to see first how the algorithm works with simple fields. The squared field is the most basic prototype. The solution of the algorithm without any servicing is presented in Figure 52. In this case, the algorithm has ended up as back-and-forth swathing, the starting point is in bottom-right corner. It should be noted that, during the last ten swaths, the turning is not made anymore to the next swath but instead to the opposite side of the unoperated area. This is known as the *squeeze swathing* technique. This happens because turning to the next one is more time consuming than to skip a couple of swaths.

Whether the solution ends up as rounding the field or going back-and-forth, depends on the time needed to make the different kinds of turnings. If the minimum turning radius in headland was 12 m instead of 6 m, the result would be rounding the field, as shown in Figure 53. Figure 54 shows a solution for a rectangular field with aspect ratio 4:1. Here at the end also, the two opposite sides are driven instead of going back-and-forth.

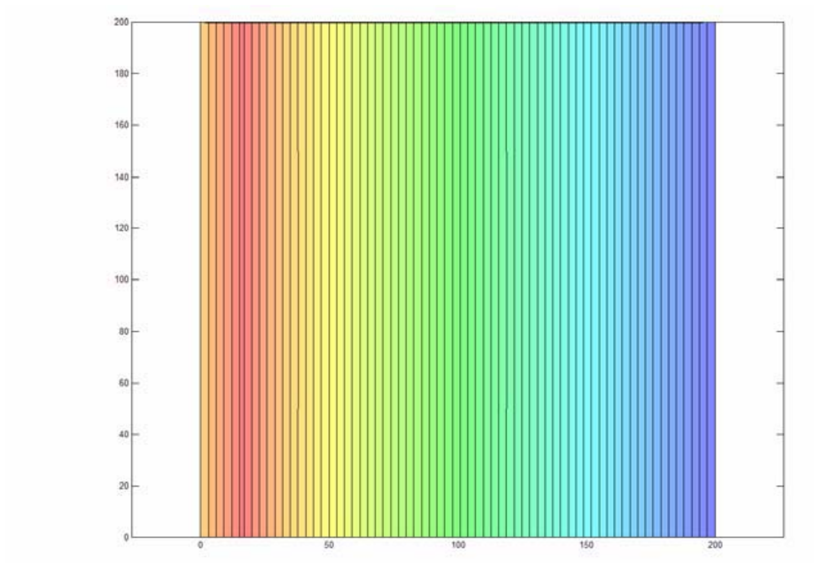


Figure 52. Square field

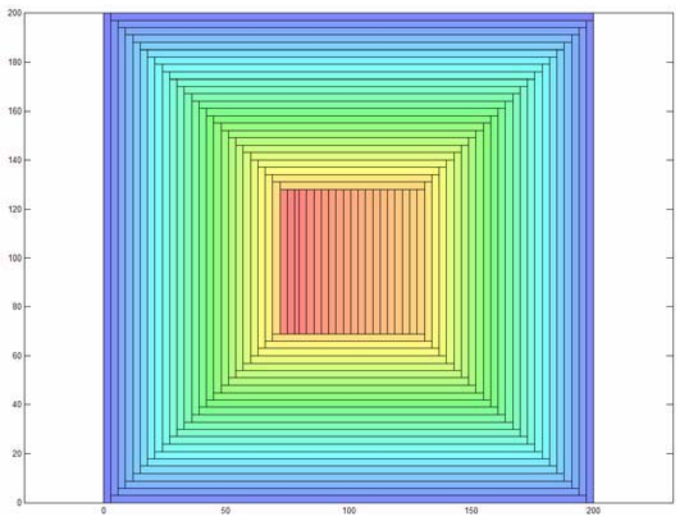


Figure 53. Square field with turning radius 12 m

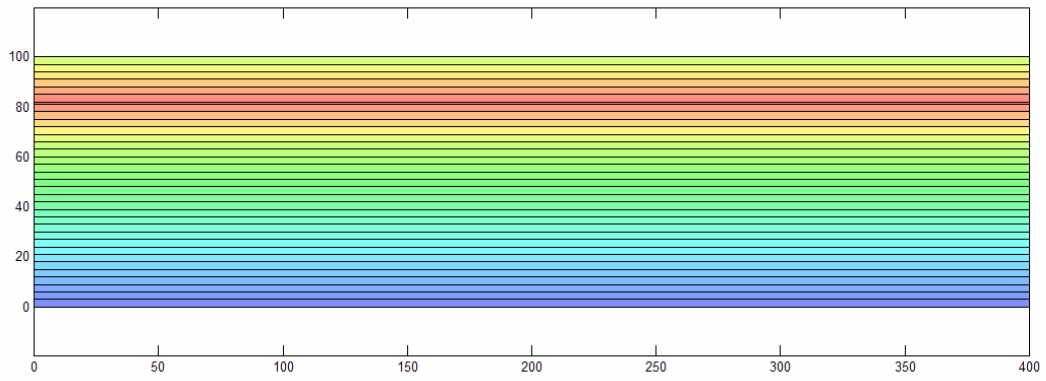


Figure 54. Rectangular field with aspect ratio 4:1

The next trivial case is a triangle. With default parameters, the solution is similar to that in Figure 55. Again, if the turning radius was 12 m, the result would change dramatically, as seen in Figure 56. With another triangle with side length ratio 2:2:1, the result will end up driving both long edges and shifting always in the sharpest corner, see Figure 57.

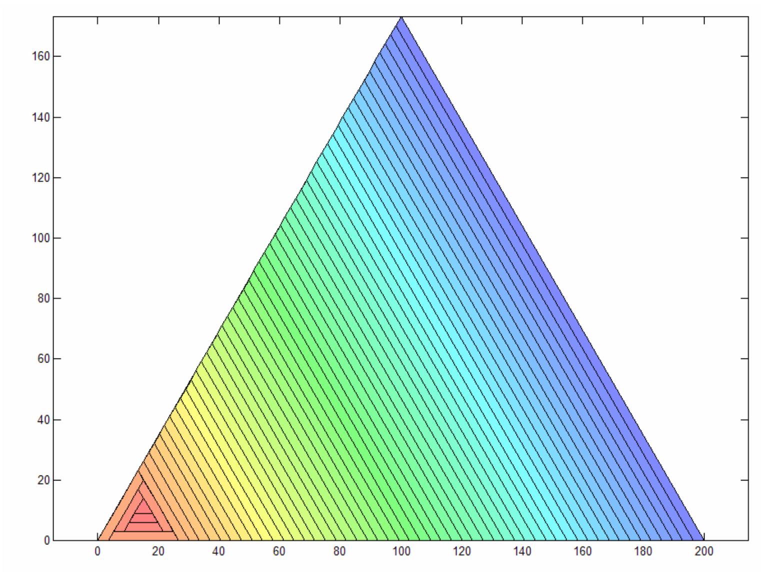


Figure 55. Triangular field with sides 1:1:1

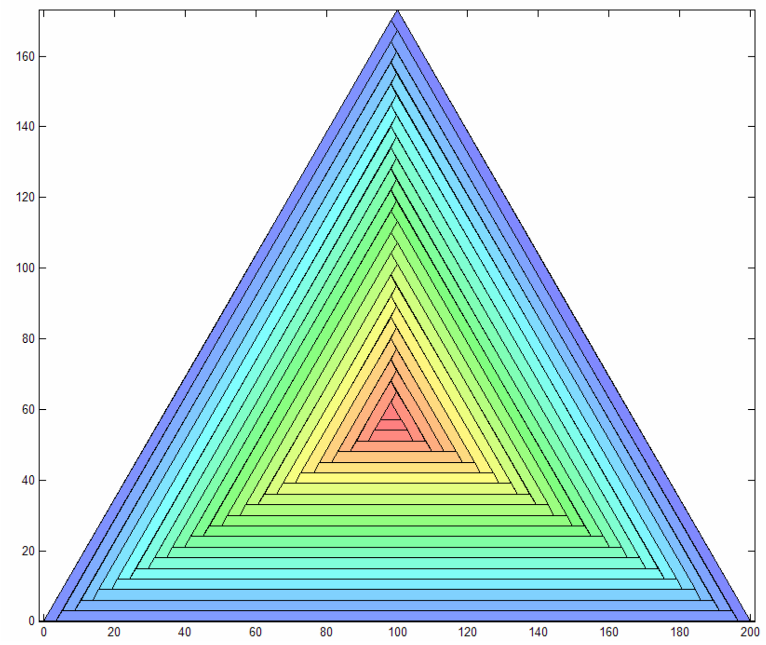


Figure 56. Triangular field with sides 1:1:1, turning radius 12 m

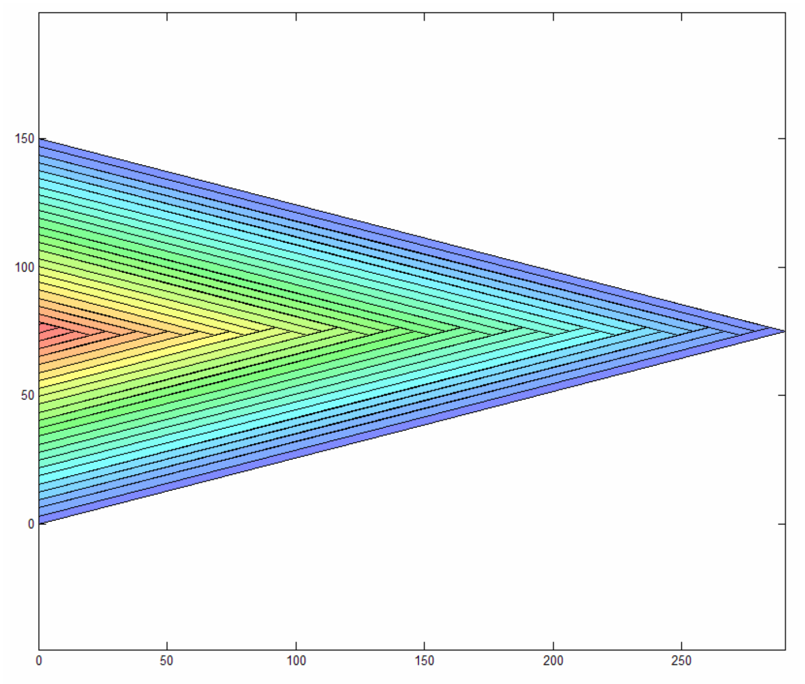


Figure 57. Triangular field with sides 2:2:1

With these examples, the basic behavior of the algorithm was understood to be reasonable and the effect of turning time was learned.

L-shaped field is a good example of the power of this algorithm. In Figure 58, a field with L-shape is presented. As it can be seen in the solution, at first the algorithm starts to drive both "wings", and switching a wing in a corner. This behavior continues until the sixth swath in both wings. Then the algorithm "detects" that it is more efficient to drive only the longest wing and make 180° turns instead of driving to wings and making 180°+270°+180° turnings. After about 17 swaths, the algorithm again changes the behavior so that both sides of the longest wing are driven at the same time. The rest of the field, the shortest wing, is operated with back-and-forth, and the last swaths with using both sides. This solution can be said to be very good in the sense of work efficiency.

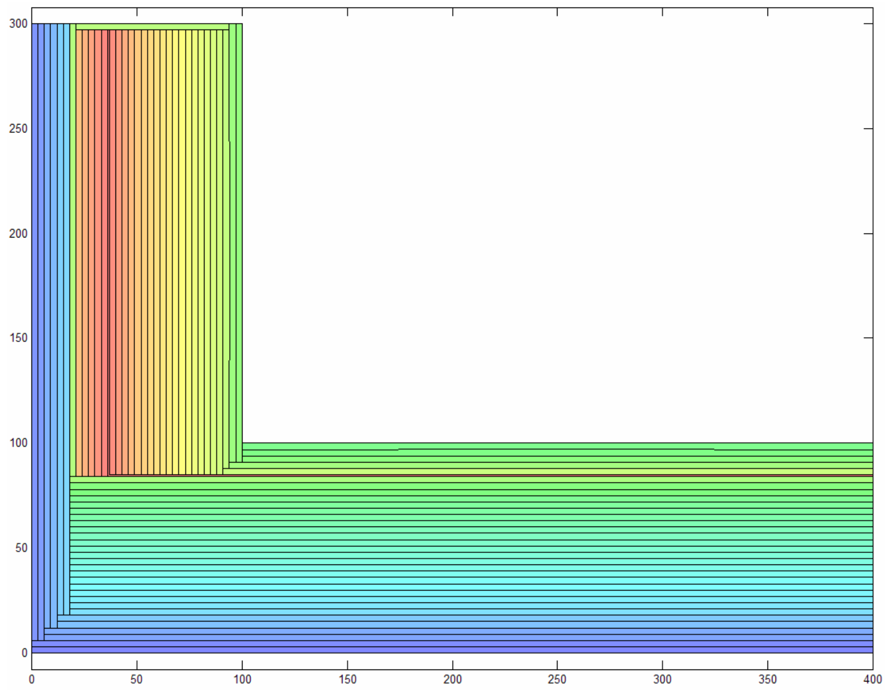


Figure 58. L-shaped field 4:3

5.4.3 Fields with curved edges

The solutions for some fields with curved edges are shown in the next set of figures. In Figure 59, a D-shaped field is presented. The most efficient continuous driving line is found on the curved edge and that is followed until the curvature becomes too big and turning is needed.

In Figure 60, a field with no straight edge and the solution for it is shown. Luckily, one long edge has so small curvature that it can be driven in a continuous line. As the vertices of the curved edge are reflex, the curvature decreases or remains the same and therefore the solution follows the same curved edge for as long as possible. The solution is practical. The problem of an algorithm with multiple regions is visible in this solution; jumping between regions happens too easily.

In Figure 61, a fictional field with many regular curves is presented. For this example, the toolwidth is changed to 10 m in order to speed up the computation. At the bottom, the continuous "nice" edge is cut by inserting a sharp bay. This is made in order to force the turning to be made in the desired position. In first seven swaths, the field is rounded. Thereafter the curvature becomes too high and turning is needed.

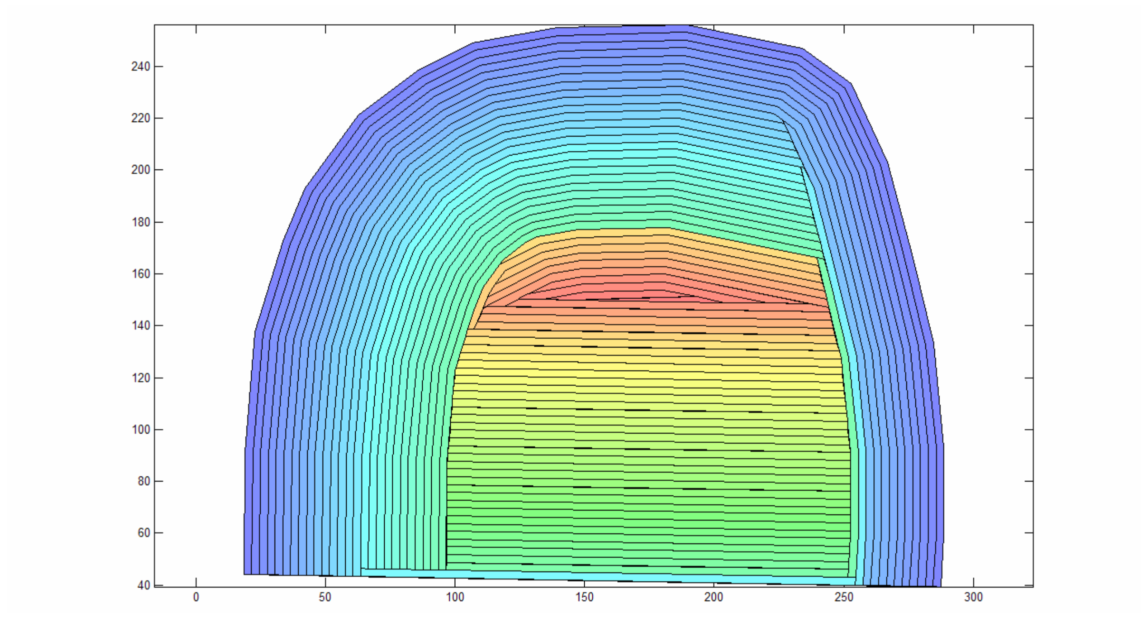


Figure 59. D-shaped field

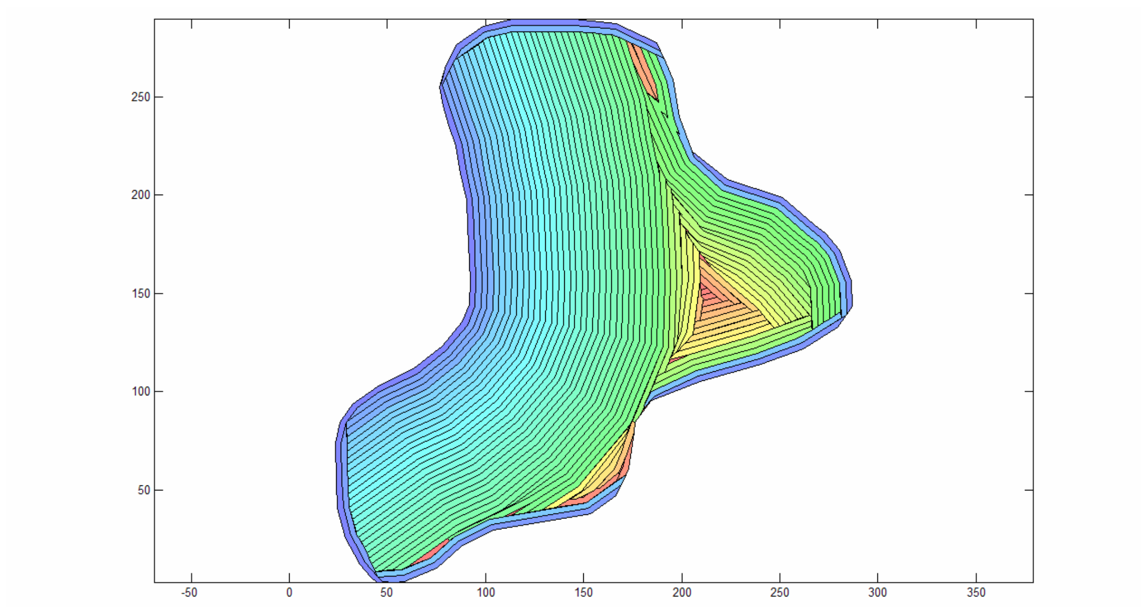


Figure 60. Field with no straight edges

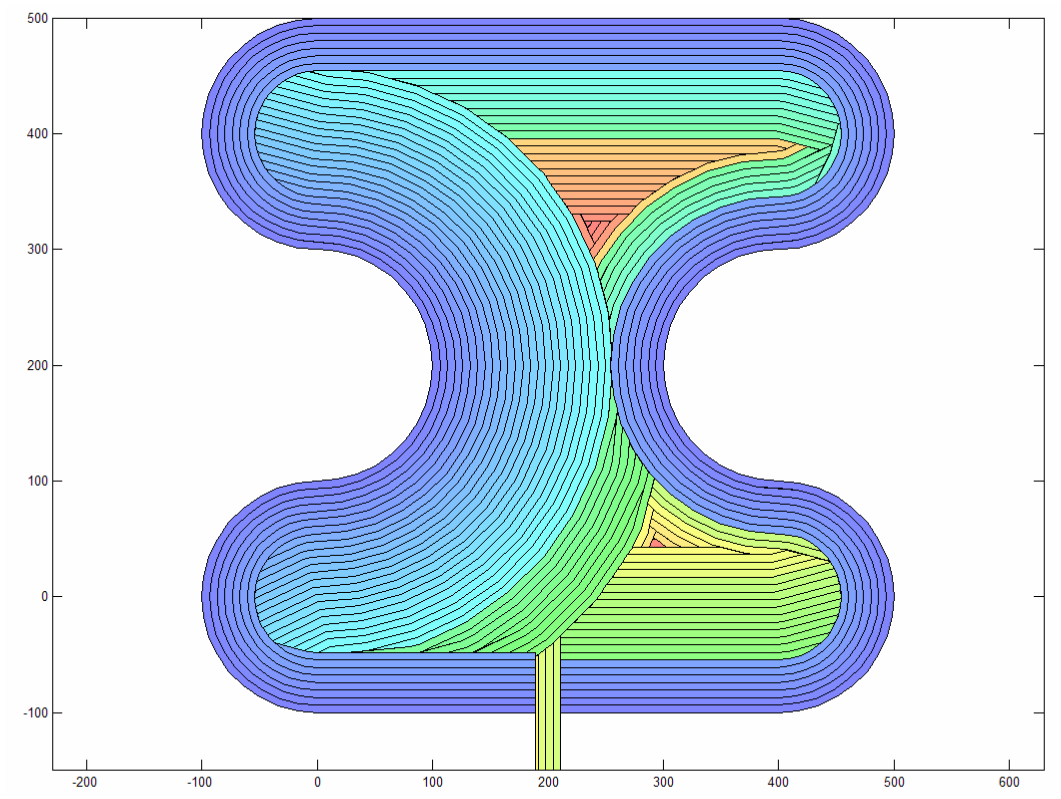


Figure 61. Rounded-H shaped field (toolwidth=10m)

5.4.4 Fields with obstacles

Here are two examples of solutions for fields that contain obstacles. In Figure 62, the case is a rectangular 4:1 field with a rectangular obstacle at the left end. The solution starts normally until the obstacle comes to the way and the algorithm continues following the longest edge. In the end, the "back of obstacle" is operated perpendicular to the main driving direction. In Figure 63, the solution starts following the left edge until the edge comes to the way. After that, the field is rounded twice, with skipping some short segments. In this solution, the edge of obstacle is followed only in two swaths. However, the solution in both cases is reasonable.

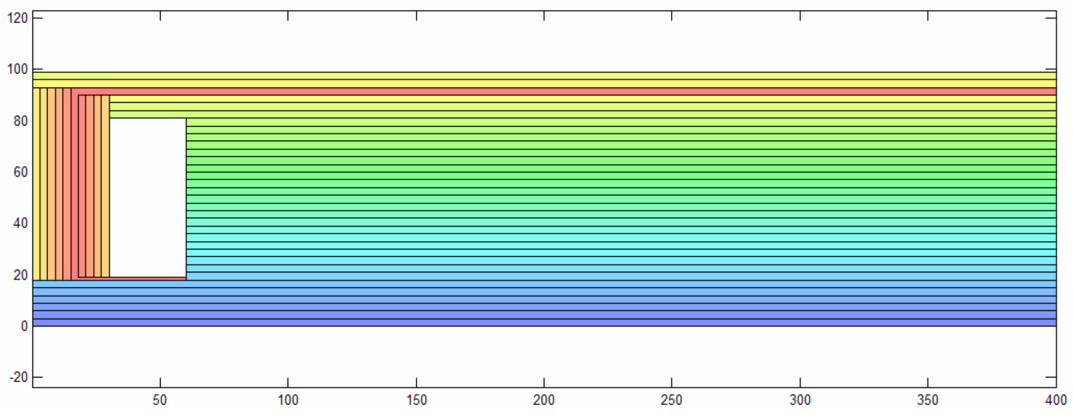


Figure 62. Rectangular field with rectangular obstacle

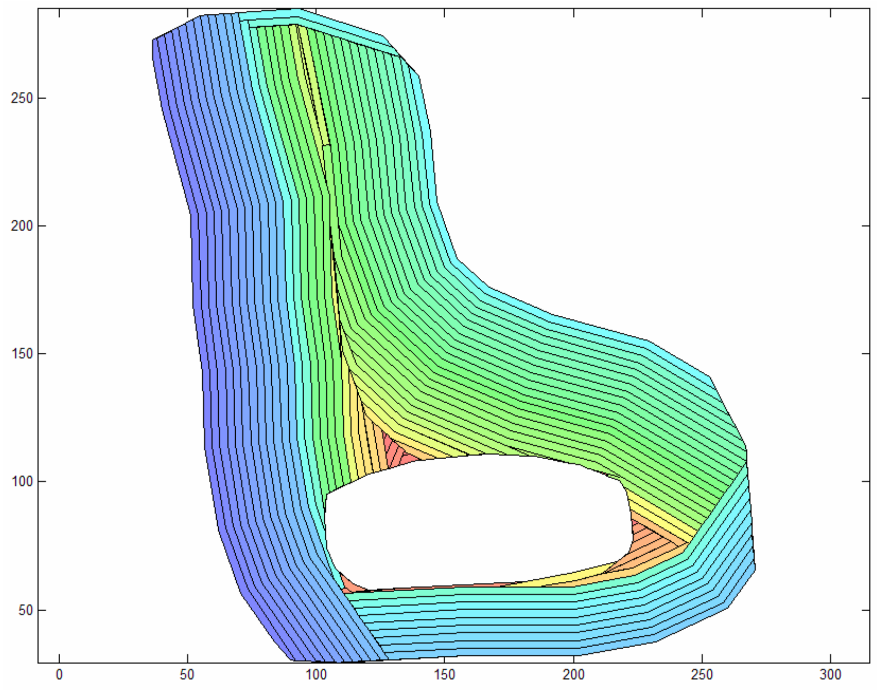


Figure 63. Field with obstacle.

5.4.5 Servicing

The refilling or emptying of the tank of the machine was included in the algorithm. In the following examples, the functionality of the approach is shown.

In the first example, in Figure 64, the field is square, as was also the case in Figure 52. The field is 200 m x 200 m, so the area is 4 ha, the machine tank capacity is 400 and the spreading rate is 500 ha^{-1}. The physical unit for tank capacity may be liters or kilograms, but here only the ratio of spreading rate and tank capacity matters. The service point is marked with *T*, and, in this example, is located "outside" of the field on purpose in order to clarify the visualization. The starting point is at the edge of the field. The center lines of operational swaths are drawn in blue, normal turning trajectories with magenta and visits to the service point are represented with bold black lines. As it can be seen, the solution is reasonable. No trip to the service point is made from the far edge. Six services were completed, and the tank was full at the start.

As described in the algorithm above, it is possible to handle multiple service points as far as they are stationary. The field is the same as in the previous example, but now there are two service points, in two opposite corners, see Figure 65. The route is automatically organized so that the nearest service point is used each time; six services were also needed in this example.

One may start counting that 4×500=2000, and (6+1)×400=2800, and 2000÷400=5, so that four services should be enough. The reason why more than the ideal number of services is needed is the fact that the tank can never be operated completely empty, as only full swaths are considered in this algorithm. So every time the machine comes to the service point, there is still some amount of material left in the tank, and the amount added is less than the full capacity of the tank.

The number of services depends partly on the location(s) of service point(s). If the service points are available so that, when the service is required, the tank level is so low (or high) that it is not anymore possible to operate any of the possible segments, the number of services needed should decrease or remain the same. In Figure 66, four service points are supplied, and, in practice, there is always the possibility of making a visit to service point at the end of every swath. In this case, the number of services needed is only five, compared to the six in the previous two examples.

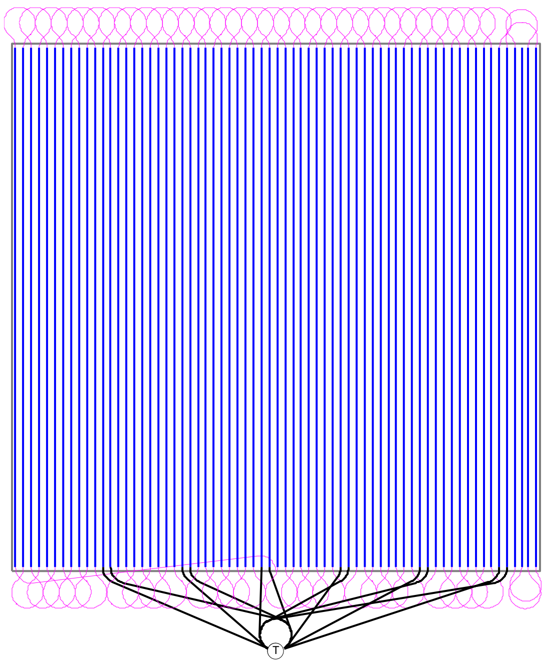


Figure 64. Square field with servicing

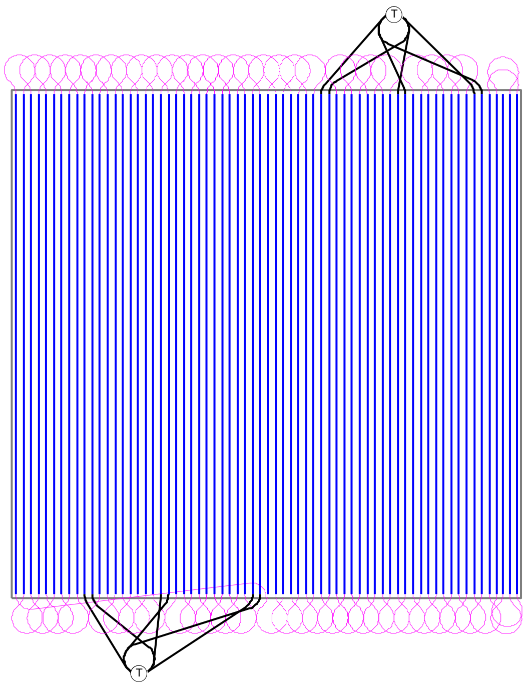


Figure 65. Square field with two service points

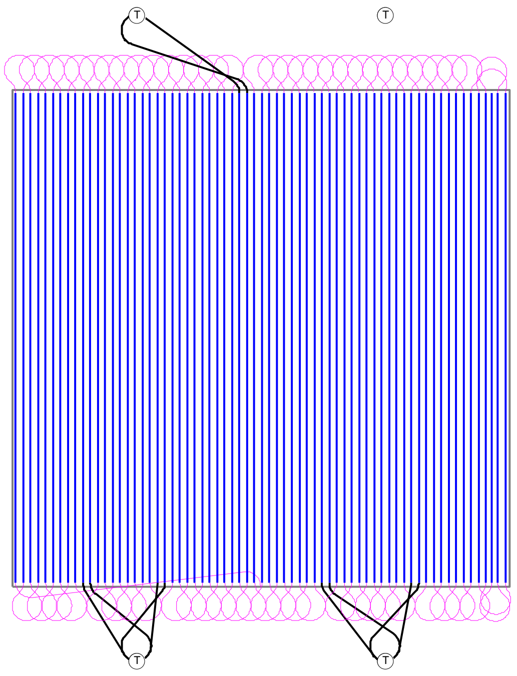


Figure 66. Square field with four service points

Another example is shown in Figure 67. The purpose of this example is to demonstrate that servicing also changes the route compared to solution where no service is needed. If no service was needed, the route would mainly follow the top edge of the field. However, in this case after eight swaths along the top edge the algorithm "sees" that the service is needed and the best way to make it is to change the driving direction so that it first drives the line parallel to the left edge of the field (in order to get near to the service point without useless driving), then makes the service and returns along the left edge back to the longest edge (top) and continues. First two services are made using this strategy, and in the third time it is more efficient to drive the tank all empty along the top edge and after that directly go to service point. In its entirety the solution is very nice and reasonable.

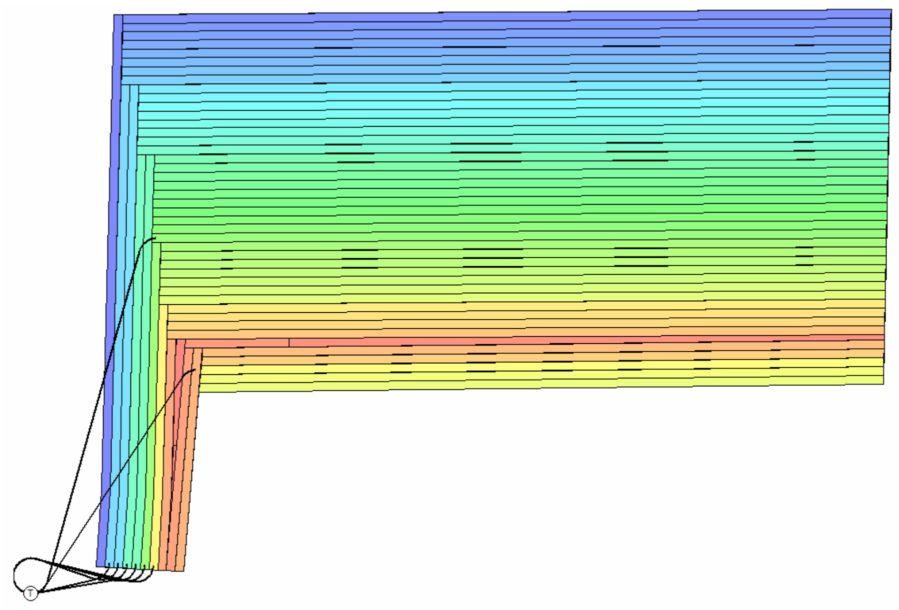


Figure 67. L-shaped field with servicing

5.4.6 Complete examples

How about "very hard" real fields which were presented in Chapter 2.7? Figure 68 shows the solution for field in Figure 25. Servicing was not considered. The solution is found and there is some sense in that, at least the left end is operated well and maybe also the right end. The only thing that clearly decreases the overall optimality is the bay in the center which seems to split the nice long swath, and "personally I would" drive that part differently. But it should be reminded that this kind of solution that can be generated by common sense was excluded from the search space in the algorithm.

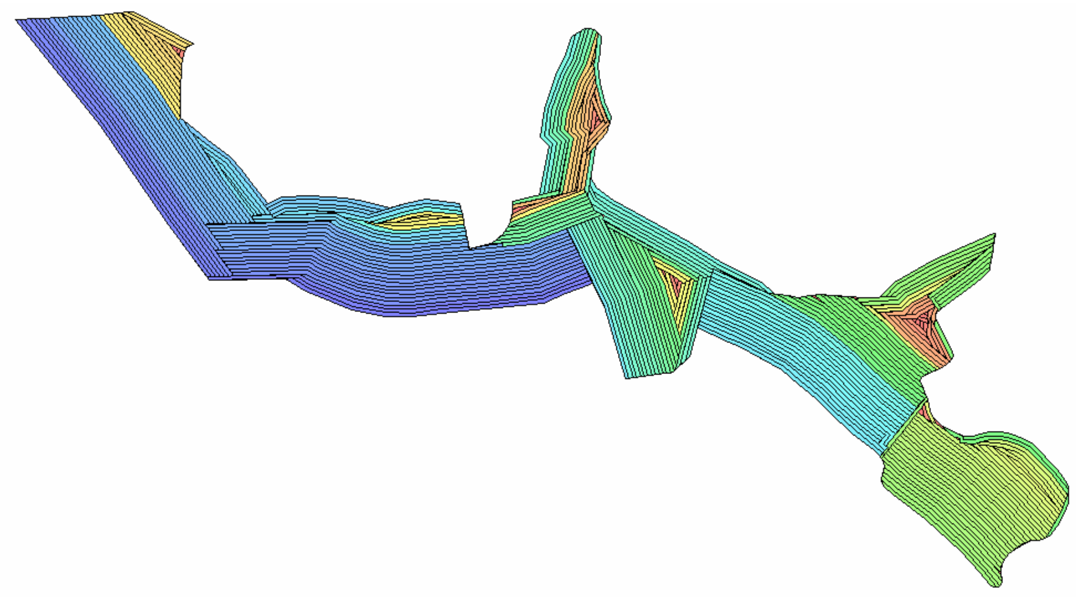


Figure 68. Complete example.

Other complete examples with real fields and realistic situations are presented in Appendix A. Those examples are provided as such without explanation. All of them are reasonable and feasible for operational instructions.

5.4.7 About computational speed of algorithm

With a modern office computer (P4@3.2GHz), the whole calculation of an L-shaped field (Figure 58, with 6 critical vertices) takes less than four minutes, when the field area is 6,0 hectares and the simulated driving in that field takes 133 minutes. With this complexity, the algorithm works well in real time. But for some more complex shapes the computation takes much longer and the real-timeliness of this cannot be guaranteed. For example, for the field plot in Figure 68, the computation took over two hours.

The algorithm is developed with Matlab, and was more concentrated on functionality than computational optimality of the code, therefore the speed of the algorithm could be much better if programmed more carefully. For example, some subalgorithms use dynamic matrix creation (pile up rows) which is known to be quite slow in Matlab.

If this algorithm is applied online in some kind of mobile driving assisting system, the edge being driven currently should be simulated to the end, when the driver decides, and the search for the next corner can be started. The time to make the calculation decreases as the field is operated and the length of swaths becomes shorter, at least if rounding the field. For such a real-time assisting system, it must be noted that there is no guarantee that the calculation is finished in a certain time. Therefore some heuristics must be used to either calculate first a rough solution or sort the possible routes based on some preknown criteria (for example, from earlier calculations) and after that calculate as many solution possibilities as there is time to do so.

5.5 Conclusion

In this chapter, a new algorithm for coverage path planning, especially for field operations, is presented. Due to the limitations set, the algorithm is not able to find the global absolute optimal solution, which always remains suboptimal. This is analogous to model predictive control strategy: the strategy is a compromise between optimality and computational performance. Because the optimal solution is not available, it is difficult to see how near to optimal the solutions of various algorithms are.

The headlands or the turning area are not taken into consideration in the current phase of development. The areal limitation inside which to make the turning should be included in the time estimation of each turning. This is omitted for the sake of speeding up the computation. It is possible to use the method presented in Chapter 3 to solve the turning in each turning, but then the time to solve one step of this algorithm would be much higher.

The refilling or emptying (servicing) subalgorithm seems to work well; its behavior is similar to common practice. The only problematic situation where this algorithm is not leading to an optimal solution is when the field has a long edge and the service point is in the middle of the longest edge. In this algorithm, the service is always made at the end of the segment, and this leads to some unnecessary driving. This could be solved by, for example, splitting the segment on the point which is nearest to the service point, or by optimizing the split point in the segment, but this was not implemented in this work due to the time constraints of the project.

6 Discussion

In the second chapter, the shapes of Finnish field plots were analyzed. Based on the literature review, it was noted that shapes of field plots have hardly been researched at all. One problem may be that evidence was searched only in scientific material in English, and it may be possible that some similar research has been undertaken somewhere, but not published in English. However, six different shape measure indexes were calculated, all of which were found in the literature. Three of them (convexity, compactness, ratio of principal moments) are widely used for shape classification and the other three (triangularity, ellipticity, rectangularity) are more recent methods and somehow still experimental, and there are different algorithms to calculate these indexes. The algorithms to calculate the last three were selected based on earlier comparison results. One of the missing and tempting indexes that should have been used is trapezoidality, but unfortunately there is no such algorithm developed yet, or it was not found. The classification based on the shape indexes was tried, but the results were not very promising. About 25% of field plots were classified into some classes, but the rest were not classified. With less strict thresholding, it could be possible to reach a classification level of 30% or so, but still the problem is that there are so many such odd shapes that they cannot be put into any class, unless the total number of classes would be 20 to 50. The problem may not be in the process of classification itself, but in the Finnish field plots, as there really are so many kinds of shapes that they cannot be put into clear classes. However, this part of research showed the variation of field shapes in Finland and the proportions of some simple shapes. It also gave a perspective to the coverage path planning problem and its demands.

At the beginning of this research, I had no idea how hard problem the coverage path planning for agricultural machines would be. In the research plan, the idea was to formulate the path planning as an optimization problem and solve it using some standard method. One way to formulate the problem could be OCP (utilized in chapter 3); the state variable could be (*x, y, heading*) in 2D plane. This could be achieved, but such a problem could not be solved in a reasonable time with any approximation to the case of a real field. Some other conversion from a general 2D route to static optimization problem could also be used, but always the situation is the same: the problem is too complex to be solved accurately in a finite period of time.

So, after a while, I noticed that it is impossible to formulate the path planning problem as an optimization problem, as in practice this would lead into an infinite dimensional optimization, and so nonlinear, problem that could not be solved with present computers and numerical algorithms. The literature review confirmed that the real challenge lies in the matter of optimality and therefore the goal of this research was changed from "optimal path" to "feasible and reasonable path".

So something else had to be tried.

As the problem cannot be solved as a general formulation (here is a field boundary, this is a minimum turning radius, find the most efficient route), it has to be approximated in some way. In approximation, something is always lost (usually accuracy). By limiting the search space, it is possible to find a solution in finite time, but this also means that the most optimal route may not be found. But this approximation is the only way to find some solution.

It depends on the approximation of what you will lose. I tried two approximations; they are very different from each other. One could be called *top-down* and the other *bottom-up*. Both of these approximations clearly lose sight of the optimal solution in the case of the random field: for certain type of fields, such as convex fields or rectangular, the found solution may be very near to optimal, but only in some special types.

Why were these two approximations selected? Naturally, there are many reasons, but maybe one of the most significant was my personal background in practical path planning in fields, my personal experience. The approach of the proposed algorithms has some similarity to that of how humans think and try to solve the problem, or at least how I think. The split-and-merge approach resembles the way a human tries to solve the problem when he/she sees the field as a 2D boundary line printed in a paper. The other algorithm mimics somehow the way a driver thinks when he/she is in a tractor or combine harvester, sees the field from the cabin, and thinks how it could be driven.

In the current state of activity in this field of research, the automatized planning of a suitable route is more important than absolute optimality. Therefore I haven't discussed much of the absolute optimality of the routes coming out of the developed algorithms. The other problem with measuring the optimality of the routes is that you have to know the optimal route if you want to measure how near or far from the optimal solution some other route is. Furthermore, as the optimal routes are not available to me as there is no all-inclusive perfect algorithm that could do it, the measure cannot be calculated. As the optimality measure is beyond the scope of this thesis, the sensibility of the solution was shown through examples.

Both algorithms have the same main property, which can be said to be some kind of drawback: at first, the most efficient part of the field plot is searched (search space is differently set in the two algorithms) and the most efficient local area is operated first despite global optimality. Both algorithms try to extend the search space from very local to more global (block instead of single swath - round the field once instead of single swath etc.) and in that way get a better solution in the sense of global optimality or total efficiency.

As a suggestion for future research, a tempting idea would be to somehow combine the two algorithms presented in this thesis (Chapters 4 and 5). By mixing top-down and bottom-up algorithms it might be possible to minimize the drawbacks existing in each algorithm.

In practice, the coverage path planning for fields is not only looking at a field as a 2D homogenous area. In this thesis, this aspect was only briefly touched upon in Chapter 4 as the regions with prohibited driving directions were introduced. It can be discussed if the formulation is more continuous, i.e. not so strict. The reason for strict prohibited driving directions came from a culture where the ploughing driving direction is not the same as the direction of the underdrains, as in this case the drainage system would not work. I think (without proving it) that, in this case, the directions must not be exactly perpendicular to each other, and it is enough that the angle between the directions is over some limit. The efficiency of underdrainage is practically the same whether the angle is 60° or 90° degrees. If the variability of the field (varying quality of operation, varying cost to drive) should be taken into consideration in path planning as a continuous variable, the formulation should be made through cost function, not prohibited regions.

Now some remarks about the cost function in coverage path planning. In this thesis, a cost function to measure the quality of some route is simple: efficient working distance per consumed time. It could be said that time efficiency may not be the only thing to be minimized. The other properties that could be taken account of in the formula are fuel consumption, amount of compaction of soil and some needs of the driver, to mention just a few. An example of the latter would be the difficulty a driver would have in applying the route planned. A badly curved route is much harder than a straight line for a human driver to drive. In some operations, the machine is designed only for straight driving and all curving may lead to quality losses in operation, in the case of some harrows, for example. If advanced coverage path planning is to be completed, the cost function should be carefully designed. A thing that makes it difficult to utilize fuel consumption or amount of compaction in the cost function is that those measures require a model with which the cost can be predicted in a simulator. It may be hard to develop such a generic model. However, a simple cost function as used here is usually adequate.

All the algorithms presented here were implemented using Matlab. Matlab is a great tool for numerical computation and a nice environment for developing numerical algorithms, but it is also less than satisfactory, as most of the calculations are performed using floating point numbers. And when the floating point variable is used to store a real number, something is lost. Small inaccuracies are not usually problematic in engineering, but in the case of computational geometry a small inaccuracy tends to cumulate in iterations. One of the most common simple examples (for example in Matlab) is `1-0.2-0.2-0.2-0.2-0.2`; the result of that count is not zero. This inaccuracy of calculation leads to using some tolerances.

With tolerances, it is possible to solve some problems, but, on the other hand, using the tolerances leads to the world of misunderstood pairing. Sometimes the results of calculations vary only slightly, but if this goes under the tolerance, an error occurs in the algorithm. This is a very tricky problem to struggle with and there are no good solutions. In the algorithm presented in Chapter 4, this was mainly solved by developing a special programming code to solve the special cases, but that leads to nothing but solving more special cases and the outcome is that there is ten times more code for handling the special cases than for the actual algorithm.

In the algorithm presented in Chapter 5, the first idea was that the vertices should be presented in a list and the edges in another list and so on. The actions should be performed according to these lists and new vertices or edges added if needed. This works well until the need to use two vertices in the same position arises, as there are links in some vertices to some other lists. If two vertices have to be put in the same position, this will make life difficult in some other algorithm. Also, the value of the tolerance constant is not evident. In some cases, the error is cumulated over many calculations; the error depends on the number of calculations and it is hard to find a good value, as in some cases (low number of iterations) a small value would be better and in some cases a larger one is needed. And if the tolerance is too big, then it leads to a miscoupling of points. If one value has been used in some algorithm for tolerance, it does not mean that the same tolerance should be used in the other algorithm if the algorithms are applied in parallel or sequentially. This leads to a set of tolerance constants for subalgorithms. If the relations of tolerances are not correct, the whole algorithm will fail. It is not easy to tune this instrument to play perfectly.

7 Conclusions

Coverage path planning is an important part of an intelligent agricultural field machine. The machine can be a traditional tractor driven by a human with a navigation system or an autonomous vehicle, a mobile robot. As has been noted in this thesis, in reviews and in results, this part is very challenging and no universal coverage path planning algorithm to support all kind of fields and machines has yet been developed. The general problem had been proved to be NP-hard (as the traveling salesman problem) or infinite-dimensional (as an optimization problem) and approximation has to be performed in order to find a solution in a reasonable computational time.

In this thesis, two new algorithms for coverage path planning for agricultural fields are presented. The two algorithms are very different from each other. The first one uses a top-down approach to split the complex-shaped fields into simple ones, and the other one uses a bottom-up approach to cover the field using prediction and brute force methods. The former can be classified into offline and the latter into online coverage path planning algorithms. Both algorithms have advantages and drawbacks; these are reported in this thesis. Neither of them will solve the universal problem optimally. It is possible to find cases in which the algorithms are not giving reasonable solutions. Nevertheless, the developed algorithms are remarkable steps along the way to finding more optimal algorithms to solve the routing problem for agricultural operations.

As a side-result, this thesis presents six different shape-measuring indexes that can be used to describe the shape of a field plot. The shape classification was investigated, but only 25% of field plots were classified into clear classes; the rest were undetermined shapes. The study was adequate in this context, but if classification of field shapes was needed for some other purpose, at least a trapezoidality index should be developed. For 25% of Finnish fields, the coverage path planning problem is quite simple to solve, but the rest are challenging.

As another side-result, a single turning of a tractor-trailer system was optimized in various headland scenarios. By formulating the problem as an optimal control problem, it was possible to find the optimal trajectory in the minimum time with areal constraints. With these results it is possible to estimate the turning times, but also these trajectories are needed for an autonomous field machine to perform successful turning in a headland.

References

Acar, E.U, Choset, H., Rizzi, A.A., Atkar, P.N. and Hull, D. 2002. Morse decompositions for coverage tasks. *The International Journal of Robotics Research.* Vol. 21(4): 331-344.

Acar, E.U, Choset, H., Zhang, Y. and Schervish, M. 2003. Path planning for Robotic Demining: Robust Sensor-based Coverage of Unstructured Environments and Probabilistic Methods. *International Journal of Robotics Research*. Vol. 22(7-8): 441-466.

Aichholzer, O., Aurenhammer, F., Alberts, D. and Gärtner, B.. 1995. A Novel Type of Skeleton for Polygons. *Journal of Universal Computer Science*, Vol. 1(12):752-761.

Alhoniemi, E., Himberg, J., Parhankangas, J. and Vesanto, J. 2005. *Homepage of SOM Toolbox.* Laboratory of Information and Computer Science. Helsinki University of Technology. http://www.cis.hut.fi/somtoolbox/. Accessed 15th August 2007.

Arkin E.M. and Refael H. 1994. Approximation algorithms for the geometric covering salesman problem, *Discrete Applied Mathematics*. Vol. 55(3):197-218.

Arkin, E. M., Fekete, S. and Mitchell J. 1993. The lawnmower problem. *In Proceedings of 5th Canadian Conference Computational Geometry*, pp. 461–466.

Balch, T. 2000. *The case for randomized search.* Workshop on Sensors and Motion, IEEE International Conference on Robotics and Automation, San Francisco, CA. May 2000.

Bochtis D., Vougioukas, S., Tsatsarelis C. and Ampatzidis. Y. 2006. Optimal Dynamic Motion Sequence Generation for Multiple Harvesters. *Proceedings of the Automation Technology for Off-road Equipment (ATOE) 2006 Conference*, Bonn, Germany, pp. 33-40.

Bryson Jr. A. E. 1996. Optimal control 1950 to 1985. *IEEE Control Systems*, Vol. 16(3):26-33.

Cao, Z.L, Huang, Y. and Hall, E.L. 1988. Region filling operations with random obstacle avoidance for mobile robotics. *Journal of Robotic Systems*, Vol. 5(2):87-102.

Choi, H.I., Han, C.Y., Choi, S.W, Moon, H.P, Roh, K.H. and Wee N.-S. 2001a. *Two-dimensional Offsets via Medial Axis Transform I: Mathematical Theory. Preprint.* Available at: http://newton.kias.re.kr/~swchoi/homepage/offset_theory.pdf. Accessed 9th August 2007.

Choi, H.I., Han, C.Y., Choi, S.W, Moon, H.P, Roh, K.H. and Wee N.-S. 2001b. *Two-dimensional Offsets via Medial Axis Transform II: Algorithm. Preprint.* Available at:

http://newton.kias.re.kr/~swchoi/homepage/offset_alg.pdf. Accessed 9th August 2007.

Choset, H. 2001. Coverage for robotics - A survey of recent results. *Annals of Mathematics and Artificial Intelligence*. Vol. 31:113-126.

Choset, H. and Pignon, P. 1997. *Coverage Path Planning: The Boustrophedon Cellular Decomposition*. In International Conference on Field and Service Robotics, Canberra, Australia.

Felkel, P. and Obdrzalek, S. 1998. Straight Skeleton Implementation. *Proceedings of Spring Conference on Computer Graphics*, Budmerice, Slovakia. pp. 210-218.

Gill, P.E., Murray, W., and Saunders, M.A.. 2002. SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization. *SIAM journal on optimization*. Vol. 12(4):979-1006.

Gray, S.A. 2001. *Planning and replanning events for autonomous orchard tractors*. Master's thesis, Utah State University, Logan, Utah, USA.

Hansen, A.C., Zhang, Q. and Wilcox, T.A. 2007. Modeling and Analysis of Row Crop Harvesting Patterns by Combines. *Transactions of the ASABE*, Vol. 50(1):5-12.

Hunt, D. 2001. *Farm Power and Management*. 10th ed. Ames, Iowa: Iowa State University Press.

Jin, J. and Tang, L. 2006. *Optimal Path Planning for Arable Farming*. In 2006 ASABE Annual International Meeting. ASABE Paper No. 061158.

Klemola, E., Karttunen, J., Kaila, E., Laaksonen, K. and Kirkkari, A.-M. 2002. *Lohkon koon ja muodon taloudelliset vaikutukset* [The economic effects of field size and shape]. TTS Institutes Publication Series: 386. Helsinki, Finland. In Finnish.

Latombe, J.C. 1991. *Robot Motion Planning.*Boston, MA.: Kluwer Academic Publishers.

Liu, G. and Palmer, R. 1989. Efficient field courses around an obstacle. *Journal of agricultural engineering*. Vol. 44:87-95.

Maciejowski, J.M. 2002. *Predictive Control with Constraints*, Prentice Hall, England.

Murphy, R.R. 2000. *Introduction to AI Robotics*. A Bradford Book, The MIT Press.

Myyrä, S. 2001. *Tilusrakenteen taloudelliset vaikutukset* [Economic impacts of arable land structure]. MTT Economics publications 1/2001. Helsinki, Finland. In Finnish.

Noguchi, N. and Terao, H. 1997. Path planning of an agricultural mobile robot by neural network and genetic algorithm. *Computers and Electronics in Agriculture.* Vol. 18(3):187-204.

Noguchi, N., Reid, J.F., Zhang Q. and Will, J.D.. 2001. *Turning Function for Robot Tractor Based on Spline Function.* ASAE Annual Meeting 2001. ASAE Paper No. 011196.

Oksanen, T. and Visala, A. 2004. Optimal control of tractor-trailer system in headlands. *Proceedings of the Automation Technology for Off-road Equipment (ATOE) 2006 Conference*, Kyoto, Japan, pp. 255-263.

Oksanen, T., Kosonen, S. and Visala, A. 2005. *Path planning algorithm for field traffic.* 2005 ASAE Annual International Meeting. ASAE Paper No. 053087.

Oksanen, T. and Visala, A. 2006a. Split and Merge Based Path Planning for Agricultural Machines. *Proceedings of the Automation Technology for Off-road Equipment (ATOE) 2006 Conference*, Bonn, Germany, pp. 151-159.

Oksanen, T. and Visala A. 2006b. Recursive Online Path Planning for Agricultural Machines. *Proceedings of the Automation Technology for Off-road Equipment (ATOE) 2006 Conference*, Bonn, Germany, pp. 161-169.

Oksanen, T. and Visala, A. 2007. Path Planning Algorithms For Agricultural Machines. *Agricultural Engineering International: the CIGR Ejournal.* Manuscript ATOE 07 009. Vol IX. 19p.

Palmer, R. 1984. Optimisation of farm field operations energy developments: new forms, renewable, conservation. *Energex Proceedings.* Peragon Press, Regina.

Palmer, R., Wild, D., Runtz, K. 1988. *Efficient path generation for field operations.* Report. Dept. of Computer Science, University of Regina.

Palmer, R., Wild, D., Runtz, K. 2003. Improving the efficiency of field operations. *Biosystems engineering*, Vol. 84(3):283-288.

Pandey, M. M., and R. S. Devnani. 1987. Analytical determination of an optimum mechanical harvesting pattern for high field efficiency and low-cost of operation. *Journal of Agricultural Engineering Research.* Vol. 36(4):261-274.

Peltola, R., Mattila, P. and Kasteenpohja, E. 2006. *Pellon arvo* [Value of agricultural land]. Maanmittauslaitoksen julkaisuja 102. In Finnish.

Reid, J.F. 2004. Mobile Intelligent Equipment for Off-road Environments. *Proceedings of the Automation Technology for Off-road Equipment (ATOE) 2006 Conference*, Kyoto, Japan, pp. 1-9.

Ross, I.M., and Fahroo, F. 2002. A direct method for solving nonsmooth optimal control problems. *Proceedings of the 2002 IFAC World Congress*, Barcelona, Spain. July 2002.

Ross, I.M. 2004. User manual for DIDO.

Rosin, P. 2003. Measuring shape: ellipticity, rectangularity, and triangularity. *Machine Vision and Applications*. Vol. 14(3):172-184.

Ryerson, A.E. and Zhang, Q. 2006. Vehicle path planning for complete field coverage using genetic algorithms. *Proceedings of the Automation Technology for Off-road Equipment (ATOE) 2006 Conference*, Bonn, Germany, pp. 309-317.

Sipilä, M., Kantola, M., Uotila, P., Levander, A. and Valjakka, M. 1954. *Maamiehen taitokirja* [Farmer's knowledge book]. Työtehoseuran julkaisuja 69, Pellervo-seura, Helsinki. In Finnish.

Sorensen, C.G., Bak, T., Jørgensen, R.N. 2004. Mission planner for agricultural robotics. *AgEng 2004*, Leuven, Belgium. 8p.

Stryk, O. von, and Bulirsch, R. 1992. Direct and indirect methods for trajectory optimization. *Annals of Operation Research*. Vol. 37(1992): 357-373.

Stoll, A. 2003. Automatic operation planning for GPS-guided machinery. *Proceedings of 4th European Conference on Precision Agriculture*, pp. 657-664.

Toussaint G.T. 1983. Solving geometric problems with the rotating calipers. *Proceedings of MELECON '83*.

Witney, B. 1996. *Choosing and Using Farm Machines*. Edinburgh, U.K.: Publ. Land Technology.

Vougioukas, S., Blackmore, S., Nielsen J. and Fountas S. 2005. A two-stage route planning system for autonomous agricultural vehicles. *Proceedings of 5th European Conference on Precision Agriculture*. pp. 597-604.

Yang, S.-N. and Huang, M.-L. 1993. A new offsetting algorithm based on tracing technique. *Proceedings on the second ACM symposium on Solid modeling and applications*. ACM Press. pp. 201-210.

Yang, S.X. and Luo, C. 2004. A neural network approach to complete coverage path planning, *IEEE Transactions on Systems, Man and Cybernetics, Part B*, Vol. 34(1): 718- 724.

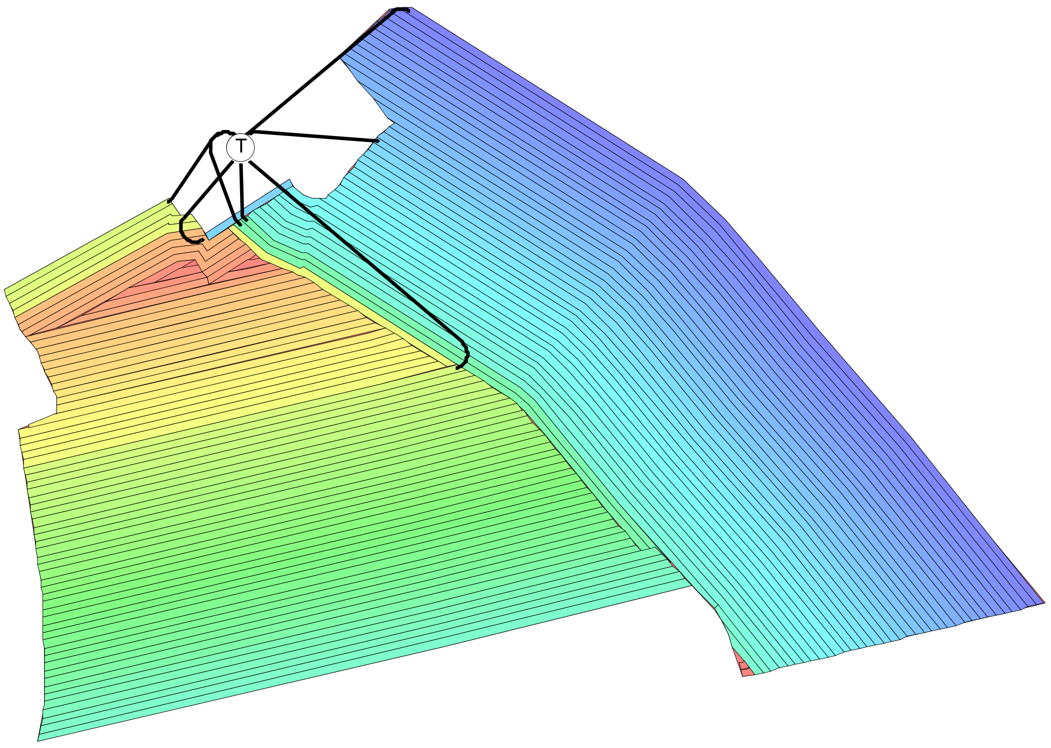
Zelinsky, A., Jarvis, R.A., Byrne, J.C. and Yuta, S. 1993. Planning paths of complete coverage of an unstructured environment by a mobile robot. *Proceedings of International Conference on Advanced Robotics*, Tokyo, Japan. pp. 533-538.

Appendix A: Complete examples

In the following figures, some complete examples with predictive recursive online algorithms for real fields are presented. With these results, the reader may take a visual survey and consider the functionality of the algorithm. In each example, a solution is projected onto the field using color coding. In the related table some key figures are taken from the route. The driving speed is 10 km/h in working phase and 6 km/h in turnings. The analysis of suitability or sensibility or optimality is left to the reader.

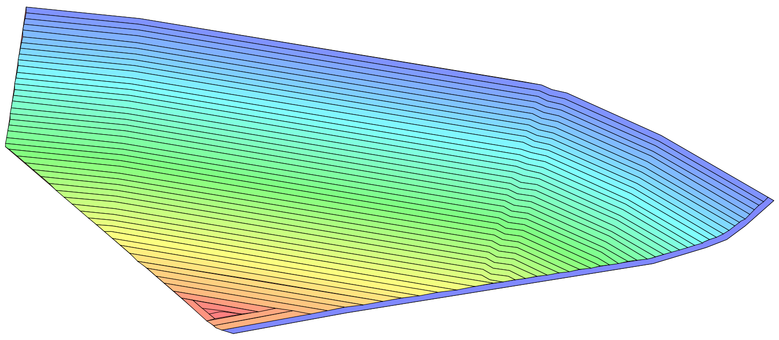
Example A

Field area	*6.8 ha*
Number of turnings	*109*
Work time	*2h 01m 20s*
Total time	*2h 38m 45s*
Work distance / total distance	*22.2 km / 27.4 km*
Overhead of time (%)	*31%*
Notes	*Toolwidth=3m, Tank capacity=1000* *Application rate = –500 ha^{-1}*



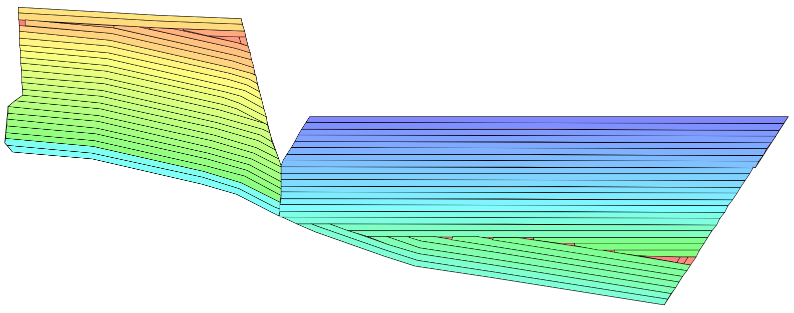
Example B

Field area	*4.1 ha*
Number of turnings	*51*
Work time	*1h 08m 34s*
Total time	*1h 26m 15s*
Work distance / total distance	*12.6 km / 14.9 km*
Overhead of time (%)	*26%*
Notes	*Toolwidth=3m*



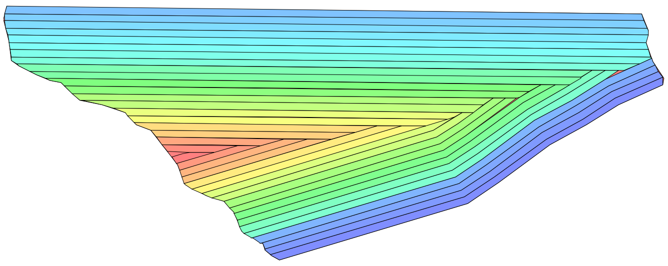
Example C

Field area	*2.5 ha*
Number of turnings	*61*
Work time	*0h 43m 22s*
Total time	*1h 06m 25s*
Work distance / total distance	*8.0 km / 11.0 km*
Overhead of time (%)	*53%*
Notes	*Toolwidth=3m*



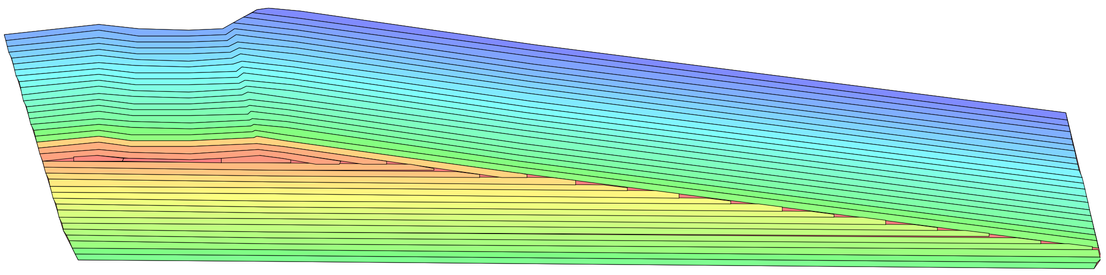
Example D

Field area	*1.8 ha*
Number of turnings	*38*
Work time	*0h 31m 36s*
Total time	*0h 55m 56s*
Work distance / total distance	*5.8 km / 9.0 km*
Overhead of time (%)	*77%*
Notes	*Toolwidth=3m*



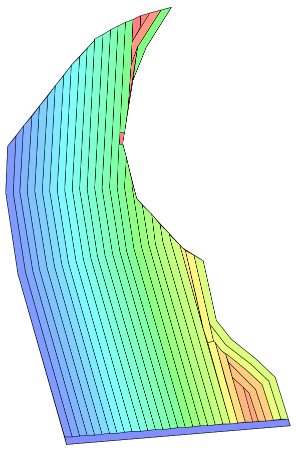
Example E

Field area	*5.2 ha*
Number of turnings	*43*
Work time	*1h 32m 07s*
Total time	*1h 47m 17s*
Work distance / total distance	*16.9 km / 18.9 km*
Overhead of time (%)	*16%*
Notes	*Toolwidth=3m*



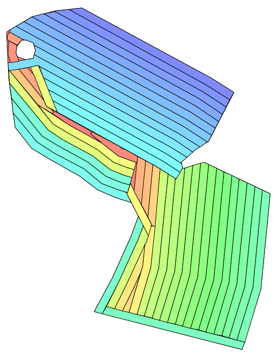
Example F

Field area	*1.0 ha*
Number of turnings	*38*
Work time	*0h 17m 47s*
Total time	*0h 31m 53s*
Work distance / total distance	*3.3 km / 5.1 km*
Overhead of time (%)	*79%*
Notes	*Toolwidth=3m*



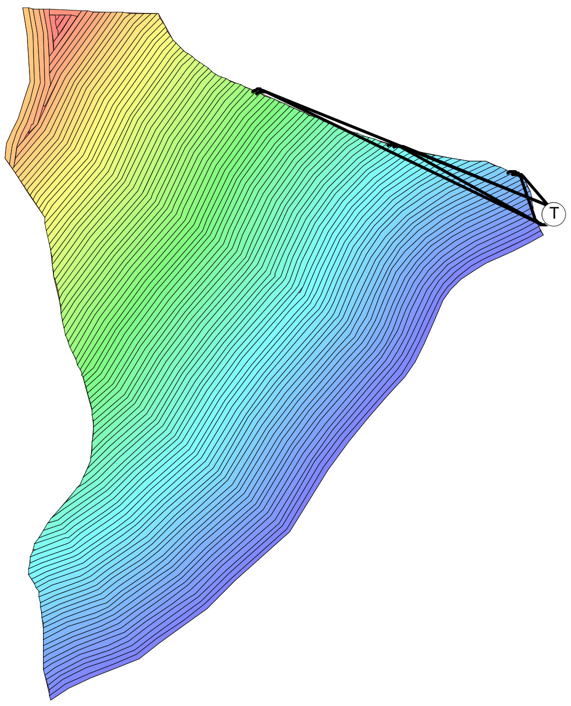
Example G

Field area	*0.71 ha*
Number of turnings	*49*
Work time	*0h 12m 02s*
Total time	*0h 27m 24s*
Work distance / total distance	*2.2 km / 4.3 km*
Overhead of time (%)	*128%*
Notes	*Toolwidth=3m*



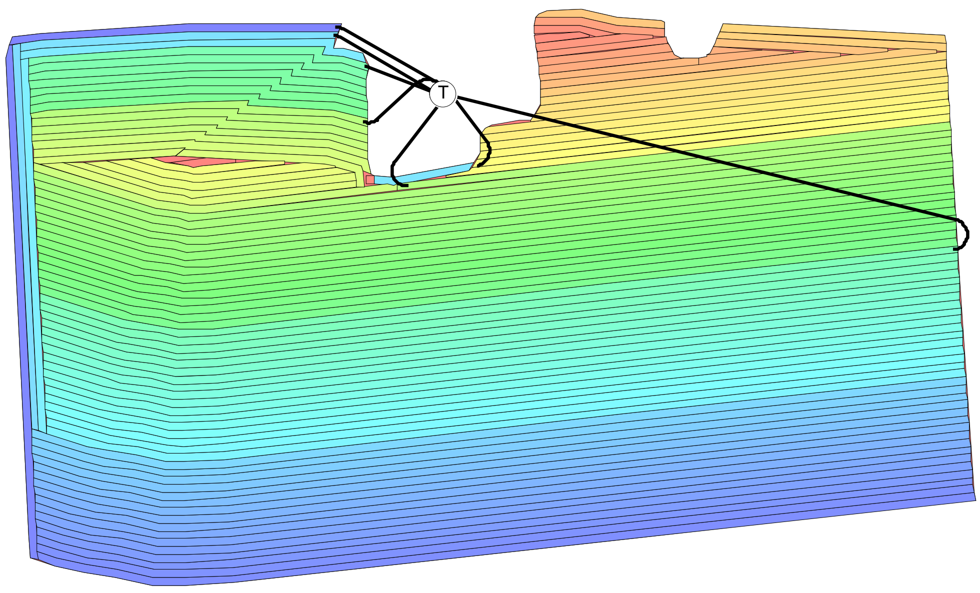
Example H

Field area	*7.0 ha*
Number of turnings	*100*
Work time	*2h 05m 39s*
Total time	*2h 37m 29s*
Work distance / total distance	*23.0 km / 27.3 km*
Overhead of time (%)	*25%*
Notes	*Toolwidth=3m, Tank capacity=1500* *Application rate = –500 ha^{-1}*



Example I

Field area	*7.1 ha*
Number of turnings	*100*
Work time	*2h 06m 33s*
Total time	*2h 42m 24s*
Work distance / total distance	*23.2 km / 28.2 km*
Overhead of time (%)	*28%*
Notes	*Toolwidth=3m, Tank capacity=1000* *Application rate = –500 ha^{-1}*



Example J

Field area	*2.8 ha*
Number of turnings	*68*
Work time	*0h 49m 14s*
Total time	*1h 13m 38s*
Work distance / total distance	*9.0 km / 12.5 km*
Overhead of time (%)	*50%*
Notes	*Toolwidth=3m, Tank capacity=2700* *Application rate = 4000 ha^{-1}*